Hittite Journal of Science and Engineering, 2022, 9 (4) 263–273
ISSN NUMBER: 2148–4171
DOI: 10.17350/HJSE19030000279

An Ant Colony Optimization Based Real-time Mobile Application for the Capacitated Vehicle Routing Problem

Emrehan Yavsan **Ilhan Ilhan**
Necmettin Erbakan University, Department of Mechatronics Engineering, Konya, Turkey

ABSTRACT

This work focuses on the capacitated vehicle routing problem. In this work, a real-time application is developed using online real-world data for mobile devices have IOS and Android operating systems. The fuzzy c-means clustering algorithm is used to group the demand points and the ant colony optimization algorithm is employed to determine the best route within each group. The customer demand points and distances between these points are obtained via Google Places and Google Directions APIs. The deviations in the route that result from the environmental and road conditions are identified immediately with the help of global positioning system technology allowing the route suggestions to be made. The developed application was evaluated on two datasets for testing. The test results showed that this real-time application can be used to find the optimum route for the capacitated vehicle routing problem and follow the route optimally.

Article History:
Received: 2022/07/14
Accepted: 2022/11/16
Online: 2022/12/31

***Correspondence to**: Emrehan Yavsan, Necmettin Erbakan University, Mechatronic Engineering, 42090, Konya, TURKEY*
E-Mail: eyavsan@erbakan.edu.tr
Phone: +90 332 325 20 34 (4000)

Keywords:
Ant colony optimization; Fuzzy C-Means; Global positioning system; Mobile devices; Real time application; Vehicle routing problem.

INTRODUCTION

The process of transportation of any goods or services from supply points to multiple demand locations is called logistics. A complete logistics system starts with the transfer of the raw material from the sellers or the suppliers to the factories for production. It goes on with the transfer of the goods produced in the factories to the depots or distribution centers. Distribution to the customers is the final step [1]. Both the distribution and the supply procedures need an effective transportation method, because about 50% of the logistics cost for any business is due to distribution activities [2]. Therefore, effective and efficient utilization of distribution equipment and staff is of great importance for business managers.

The vehicle routing problem (VRP) was defined by Dantzig and Ramser [3]. In accordance with their definition, a fleet consisting of vehicles with identical or different capacities needs to serve a customer group located at different places with different demands from a central store. The objective is to select the most convenient route, time and cost. Since its initial definition, new constraints have been included to VRP, and various types have been proposed. For each type of problem, various models were developed and many algorithms have been proposed [4,5]. Some of those algorithms use the exact solution methods, while others use heuristic methods [6–10]. In the first of these studies, a heuristic method for the capacitated vehicle routing problem (CVRP) solution was suggested, which depends on the principle of "cluster first, route thereafter" [6]. In the clustering step, the fuzzy c-means (FCM) clustering algorithm was used and membership values between 0 and 1 were assigned to each demand point. In the routing step, the route was enhanced using the tabu search algorithm. The suggested method was tested on benchmark datasets presented in the literature. In a later study, a web-based spatial decision support system (wSDSS) was proposed that depends on ant colony optimization (ACO) [7]. In this system only a web browser was needed; therefore, it had the flexibility to be used in various real-world conditions. The proposed method was tested on the city garbage collection network in Coimbra, Portugal. In another study, an advanced ACO method was proposed that had a novel strategy and a mutation process to update increasing pheromone levels [8]. This method, which used an ant-weight strategy, was tested on 14 different benchmark problems. The results obtained were compared with other methods proposed for CVRPs. In contrast, Harmanani et al. [9] developed a new method

for the solution of the same problem based on the simulated annealing algorithm. This method included a combination of random and deterministic operators based on the problem information. To test the suggested method, 24 benchmark datasets were used and comparisons with other methods were made. Additionally, a new method, based on a genetic algorithm that used an optimized crossover operator, was suggested by Nazif and Lee [10]. For the crossover process in this method, two parents were chosen first and then two offspring were produced with the help of a mechanism constructed by a complete, undirected, bipartite graph. This method, called optimized crossover genetic algorithm (OCGA) was tested on existing benchmark datasets.

In addition to the above studies, new studies have also been carried out, especially using heuristic methods [11–16]. For instance, Zhang and Lee [11] proposed an algorithm that includes numerous improvements to enhance the capabilities of the diversified and intensified search of the conventional artificial bee colony (ABC) algorithm. They called it RABC. The RABC algorithm was examined with different benchmark instances. Teoh et al. [12] suggested an improved differential evolution algorithm with the local search (DELS). They used three standard local search operators, which are drop one point, swap two points and flip method. The DELS algorithm was tested on benchmark instances. Mohammed et al. [13] proposed a new method using the k-nearest neighbor algorithm called KNNA. The KNNA algorithm was designed not to require a large database to record the population and it was examined with well-known instances. Rabbouch et al. [14] suggested a new dynamic version of the simulated annealing algorithm, which was called the empirical-type simulated annealing (ETSA). The ETSA method incrementally exploits the last portion of worse feasible solutions and updates the Boltzmann acceptance criterion of the simulated annealing. The method was tested on benchmark instances. Altabeeb et al. [15] proposed a new hybrid firefly algorithm (CVRP-FA). They combined improved 2-opt and 2h-opt algorithms. Besides, they used two mutation operators and a partially mapped crossover. Thus, they improved the solutions and obtained a higher convergence rate. The CVRP-FA method was examined with different scale instances. İlhan [16] suggested an improved simulated annealing algorithm with crossover operator (ISA-CO). In the ISA-CO method, a population based simulated annealing algorithm was used. The solutions in the population were improved through the reversion, insertion, scramble and swap operators. The routes of the solution were developed the improved 2-opt algorithm. The order crossover and partially mapped crossover operators were applied to the solutions to obtain a higher convergence rate. The ISA-CO algorithm was tested on 91 well-known instances.

As seen in the literature review, throughout the studies of the solution to CVRP, there is no cross-platform support except for the web based ones [6,8–16]. None of these has a special interface designed for mobile devices. Only one of them was tested on real-world data [7], while the others were tested on widespread benchmark datasets [6,8–16]. Again, none of them utilized global positioning system (GPS) [6-16]. In this work however, a real-time application for CVRP was developed that does use real-world online data and that can run on Android and IOS devices. In this application, the FCM clustering method was used to group the demand points and the ACO algorithm was preferred to determine the best route within a group. Customer demand points and distances between these points were obtained instantaneously via Google Places and Google Directions APIs. The deviations in the routes, due to environmental and road conditions, were identified immediately with the help of GPS technology, and new route suggestions could be made. The developed applications were experimented on two different datasets; 1) against a benchmark and 2) against real-world data.

MATERIAL AND METHODS

Vehicle Routing Problem

In the Travelling Salesman Problem (TSP), a vehicle or a salesman begins from a specific location, stops at every other location in the system once and comes back to the beginning [17,18]. The aim is to minimize the total tour distance. VRP, however, is a more complicated derivative of m-TSP, which includes more than one salesman. In addition to the necessity for every salesman to have a different route, which is present in m-TSP, VRP was obtained by introducing the constraint that salesman can carry a certain amounts of load and every point can have different demands [1]. VRP can be static or dynamic depending on the environment and open- or closed-ended, depending on the route. More subcategories are defined depending on the number of constraints that are present in the problem, such as time, cost, the distances between the demand points, or the capacity of the vehicles [19,20]. The mathematical model for CVRP, which is commonly studied in literature and is considered in this study, is given below [21]:

$$min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ijk} \quad (1)$$

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1, \ \forall i \in N \quad (2)$$

$$\sum_{i \in N} m_i \sum_{j \in N} x_{ijk} \leq q_k, \ \forall k \in V \quad (3)$$

$$\sum_{j\in N} X_{0jk} = 1,\ \forall k \in V \tag{4}$$

$$\sum_{i\in N} x_{i0k} = 1,\ \forall k \in V \tag{5}$$

$$\sum_{i\in N} x_{ihk} - \sum_{j\in N} x_{hjk} = 0,\ \ \forall h \in N, \forall k \in V \tag{6}$$

$$x_{ijk} \in \{0,1\},\ i \neq j,\ \ \forall i, j \in N,\ \forall k \in V \tag{7}$$

The equalities above were obtained for a system with a number of vehicles, V, and a number of demand points, N. Here, x_{ijk} is a binary variable stating that the connection (i, j) was established with the vehicle k, and mi is the demand of point i. The aim function of the model is given by (1). The costs for the connections (i, j) with dij (distance from point i to point j) are to be minimized with a number of vehicles, k. It is expected that every point is served by one vehicle only (2). The necessity for the routes created not to exceed the vehicle capacity qk, is included in the model by (3). Connections for every vehicle k to come in and go out of the depot is provided by (4) and (5). The constraint that regulates the flux is given by (6). A vehicle that arrives at any demand point has to leave for another demand point.

In this work, demand points and the distances between them are took place using real-world online data. Google Directions and Google Places APIs are used for this task. In Asymmetric CVRP (because the journeys are made by vehicles) $d_{ij} \neq d_{ji}$, therefore, the distances from point i to point j and from point j to point i are determined [22]. Additionally, if there are two or more alternative routes from point i to point j, the shortest one is chosen while calculating the dij cost value.

Methods

The proposed algorithm in this work consists of three steps; the grouping of demand points, the determination of the best route within each group and the implementation of route enhancing operators. The first step is achieved by using the FCM clustering method, as suggested in [6]. The second step is achieved by using the ACO algorithm. The third step is performed using the Swap and Insertion operators. The flowchart for the proposed algorithm, and the related steps are given in Fig. 1. As seen in this figure, algorithm is supplied with the necessary initial parameters for the FCM and ACO. Geographical locations of the customers and the depot (latitude and longitude), and the demands of the customers, are retrieved from a prerecorded database. Using these data, the distances between the customers themselves and the distances to the depot are calculated with the help of the

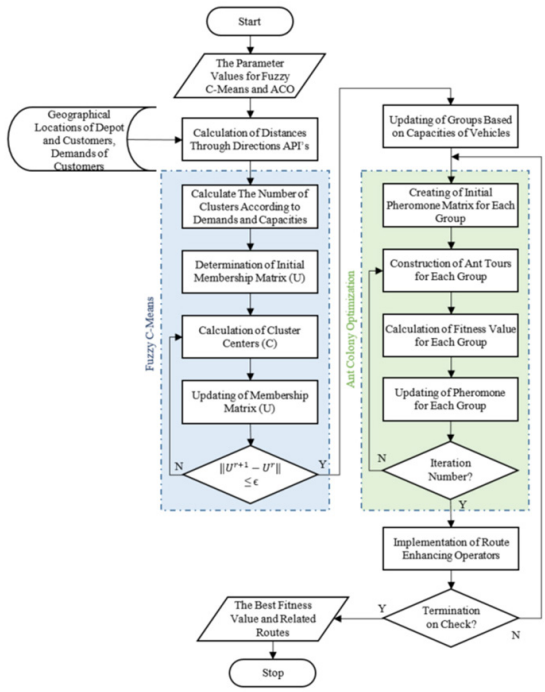


Figure 1. The proposed algorithm flowchart.

Google Directions APIs, online. Then, the FCM clustering, ACO algorithms and route enhancing operators are operated respectively.

Fuzzy C-Means Clustering

The FCM was first proposed by Bezdek in 1981 [23]. When grouping the customers, membership values are calculated for every customer in every cluster, but related customers are included in the cluster with the highest membership values. The number of clusters is calculated at the beginning of the FCM clustering algorithm with regard to the sum of the customer demands and the vehicle capacities. Then the initial membership matrix is created randomly. Calculations of cluster centers and the updates to the membership matrix are repeated until the termination criterion is reached. These processes are performed by (8) and (9), in order, respectively. For the termination criterion, the inequality (10) is used.

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \cdot X_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{8}$$

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\left\|x_i - c_j\right\|}{\left\|x_i - c_k\right\|}\right)^{\frac{2}{m-1}}} \tag{9}$$

$$\left\|U^{(r+1)} - U^r\right\| \leq \epsilon \tag{10}$$

Here, X = $\{x_1, x_2, \ldots x_N\}$ are the demand points, C = $\{c_1, c_2, \ldots c_c\}$ are the cluster centers and U = $u_{ij} \in [0, 1]$ (i = 1, 2, …, N; j = 1, 2, …, c) is the membership matrix. Element uij gives the membership degree of customer xi to the cluster cj. The parameter m is the weighted exponent that is bigger than 1, and ϵ is the termination criterion between 0 and 1. The r parameter shows the number of iterations.

The expression $\|x_i - c_j\|$ in (9) shows the distance between demand point xi and the cluster center cj. In this work, xi and cj are two geographical points on Earth and they symbolize the latitude and the longitude, respectively. The distance between these two points is not calculated as distance travelled by land but distance travelled by air. This is due to Google Directions APIs that could not always determine a route between a cluster center (cj) and a demand point (xi) for vehicle travel. For the calculations of the air distances, the Haversine formula given by (11) is used [24].

$$\begin{aligned} a &= \sin^2\left(\frac{\Delta_{lat}}{2}\right) + \cos(lat_1).\cos(lat_2).\sin^2\left(\frac{\Delta_{long}}{2}\right) \\ c &= 2\arctan(\sqrt{a}, \sqrt{1-a}) \\ d &= R.c \end{aligned} \tag{11}$$

In (11), the first point is (lat_1, $long_1$) and the second point is (lat_2, $long_2$); the distances, $\Delta_{lat} = lat_2 - lat_1$ and $\Delta_{long} = long_2 - long_1$, are then calculated. The parameter R represents the Earth radius and is assumed 6371 km in this work. After calculating the membership values of the demand points for every cluster between 0 and 1, the groups are determined by considering the vehicle capacities. For this, the demand points with the highest degree of membership are first assigned to the related group. Groups created are then checked against the vehicle capacities. If the sum of demands in any group exceeds the vehicle capacity, then the customer with the lowest degree of membership is excluded from that group. After that, the customer is assigned to the second group in terms of the degree of membership, again checking against the vehicle capacities.

Ant Colony Optimization Algorithm

The ACO algorithm was first proposed by Dorigo [25] for solutions to combinational optimization problems. In this work however, it is employed to determine the best routes after the group updates. Therefore, the operational steps explained above are repeated for every group separately, and the best fitness values and corresponding routes are determined. The ACO algorithm starts with the formation of an initial pheromone matrix. In this work, the constant value of 10−4 is assigned to every element of the initial pheromone matrix. After that, ant tours are created according to the initial population size given to the algorithm. For this process, every ant is initially placed in the depot. Then, the destination of the ant k at the depot (location i) is calculated with (12) from a number of alternative locations, u.

$$j = \begin{Bmatrix} \max_{u \in Jk(i)}[\tau(i,u)]^a * [\eta(i,u)]^\beta & if\ q \le q_0 \\ use\ Equation\ (8) & \text{Otherwise} \end{Bmatrix} \tag{12}$$

The τ(i, u) is the pheromone path on the (i, u) line. The expression η(i, u) = 1/δ(i, u) is the inverse of the distance between the points i and u. The J_k(i) symbolizes the points that have not yet been visited by the ant k at the point i. α is the importance of the pheromone amount to the related route and β is the effect of the route lengths on the choice of the next point. These parameters are greater than 0 and given to the algorithm initially by the user. q_0 ($0 \le q_0 \le 1$) gives the relative importance of studying the solution space. If $q > q_0$, then the second rule for the transition is applied and the possibilities of choice between the available routes are calculated with (13).

$$P_k(i,j) = \begin{Bmatrix} \dfrac{[\tau(i,j)]^a * [\eta(i,j)]^\beta}{\sum_{u \in J_k(i)}[\tau(i,u)]^a * [\eta(i,u)]^\beta} & If\ j \in J_k(i) \\ 0 & Otherwise \end{Bmatrix} \tag{13}$$

After all the ants' tours are completed, fitness values are calculated. The fitness value is calculated on the assumption that the ant leaving the depot would return after stopping at the demand points. The minimum values obtained and the routes providing these values are recorded as the best values for the related iteration. The local pheromone value is updated using (14) and (15).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k(t+1) \tag{14}$$

$$\Delta\tau_{ij}^k(t+1) = \begin{Bmatrix} 1/L^k(t+1) & If\ ant\ k\ used\ the\ path\ ij \\ 0 & Otherwise \end{Bmatrix} \tag{15}$$

The τ_{ij} (t) is the initial pheromone level and ρ ($0 \le \rho \le 1$) is the pheromone evaporation parameter, whose value is assigned by the user. The term L^k(t+1) represents the total tour length of the ant k, within the associated group. After all the ants complete their tours, a global pheromone update is carried out. The pheromone levels on the shortest routes preferred by the ants are increased according to (16) and (17).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^k(t+1) \tag{16}$$

$$\Delta\tau_{ij}^k(t+1) = \begin{Bmatrix} 1/L_{best}(t+1) & If\ ij\ belongs\ to \\ & the\ best\ tour \\ 0 & Otherwise \end{Bmatrix} \tag{17}$$

The term L_{best}(t+1) is the best tour length took place in the related group. These operations are rerun for the iteration count determined by the user.

Route Enhancing Operators

After the work of the ACO algorithm ends, the route enhancing operator is applied the corresponding solution. Two different route enhancing operators, the Swap and the Insertion, were used in this work. Which operator to use was decided according to the random number generated between 0 and 1. If the generated number was equal to or bigger than 0.5, the Swap operator was used, otherwise the Insertion operator was used. Both the Swap operator and the Insertion operator only operate for customers in the route enhancing pool. The customers to be added to the route enhancing pool for each cluster is determined by the threshold value. The threshold value is compared with the membership values calculated by the FCM clustering algorithm for each customer. Customers whose membership value is equal to or smaller than the threshold value are added to the route enhancing pool of the relevant cluster. All customers are naturally added to the route enhancing pool if 1.0 value is entered in the algorithm as the threshold value.

Fig. 2 and Fig. 3 show the implementation of the Swap and the Insertion operators on a current solution, respectively. First, two different clusters are determined at random. It is assumed that clusters 2 and 5 in Fig. 2b and clusters 3 and 5 in Fig. 3b are randomly determined. Then, swap and insertion points are determined randomly for these determined clusters. These points are 2 and 3 for clusters 2 and 5 in Fig. 2b, respectively. These points are 4 and 2 for clusters 3 and 5 in Fig. 3b, respectively. For the swap operator, the customers at the swap points are swapped between the two clusters (Fig. 2c). For the Insertion operator, the customer at the insertion point of the first determined cluster is moved to the insertion point of the second determined cluster (Fig. 3c). The Swap and the Insertion operators can work for one or two customers and this number is randomly determined. For the implementation of these operators, the determined customers must be in the route enhancing pool.

Interface Development

Google Android and Apple IOS dominate the mobile devices market. Therefore, the developed application was aimed at those two platforms. The programming interface was created in Embarcadero Delphi 10 Seattle which provides cross-platform support. The information from Google Directions and Google Places APIs was integrated through the TMS WebGMaps for FireMonkey component. Additionally, SQLite was preferred to keep the data that got into the application and the results.

The developed interface consists of three basic windows: Data Input, Run the Algorithm and Follow the Route.

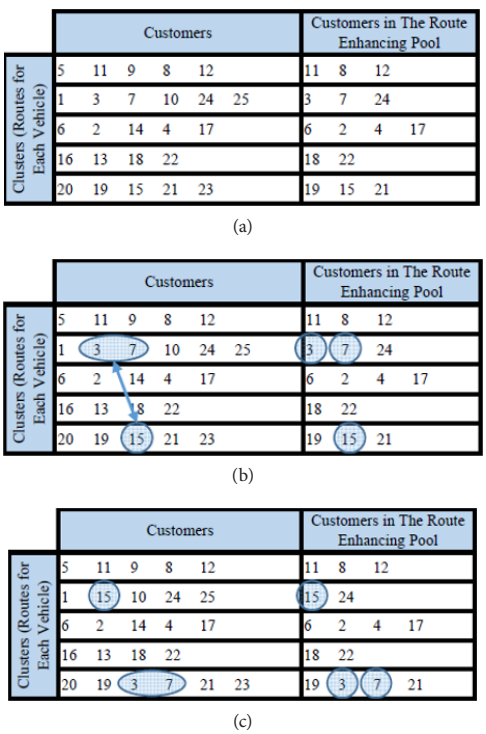


Figure 2. (a) Current Solution (b) Customer choice for Swap operator (c) New Solution.

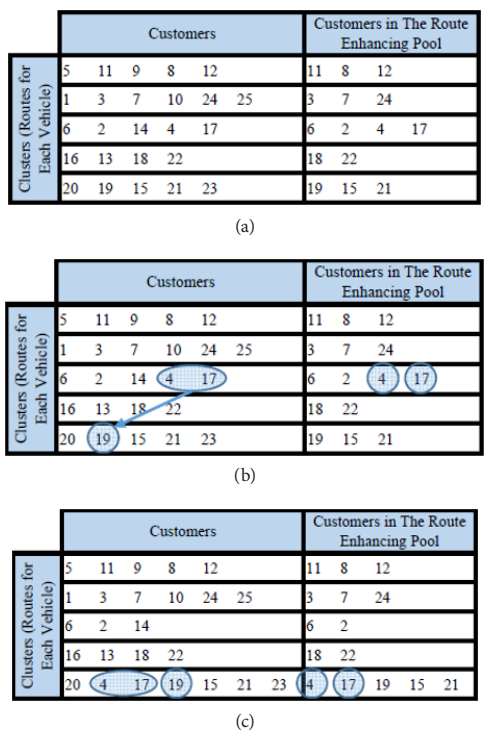
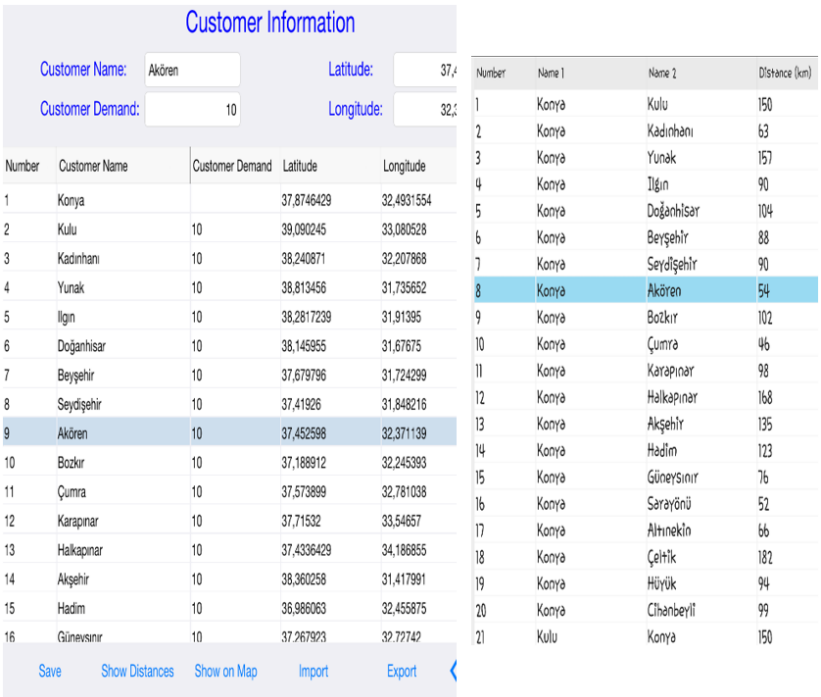


Figure 3. (a) Current Solution (b) Customer choice for Insertion operator (c) New Solution.



(a) (b)

Figure 4. (a) Depot and customer data (b) The distances between locations window.

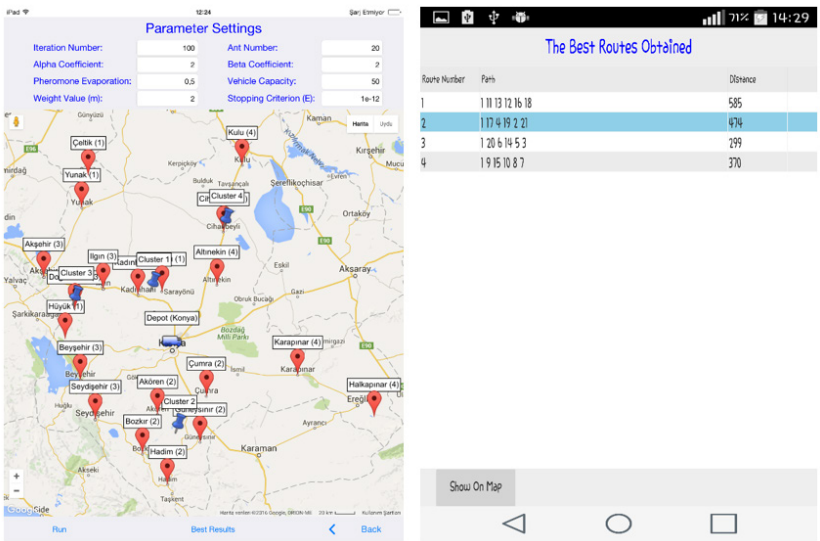
For CVRP, the Data Input window was used for the input of customer locations and demand data, showing their locations on the map and calculating the distances between them. As seen in Fig. 4a, the information input for any customer consists of the name of the customer, the amount of demand, and the latitude and longitude of the location. The first entry in the customer table is the location and the name of the depot. The data recorded in this table can be exported as a .csv extension, a common file format for text. Additionally, customer data saved with a .csv extension can be imported to this table via an associated button. Those data in the customer table can be monitored on the map. Changes can be made on the map using the marker for the related customer via the Google Places API. The distance between the locations is calculated using the Google Directions API. There might be more than one route between two locations. In that case, the shortest route is assigned as the distance between those two locations. The distances are determined both ways, because the distance from point i to point j may be different from the distance from point j to point i, for land travel using a vehicle (Fig. 4b).

The Run the Algorithm window is illustrated in Fig. 5a. parameter adjustments for FCM clustering and ACO algorithms are made in this window. After the parameter values are entered, the algorithm is run with the associated button (Run). The algorithm uses the location, the distance and the demand amount data from the Data Input window. The results are kept along with all the parameter values. The shortest distances and the routes associated with them are listed when the Best Results button is operated (Fig. 5b). Hitting the Show on Map button gives those routes as polylines of different colors on the map. Here, a desired route and its associated distance can be monitored, as well as the routes and the distances for every vehicle altogether (Fig. 5c).

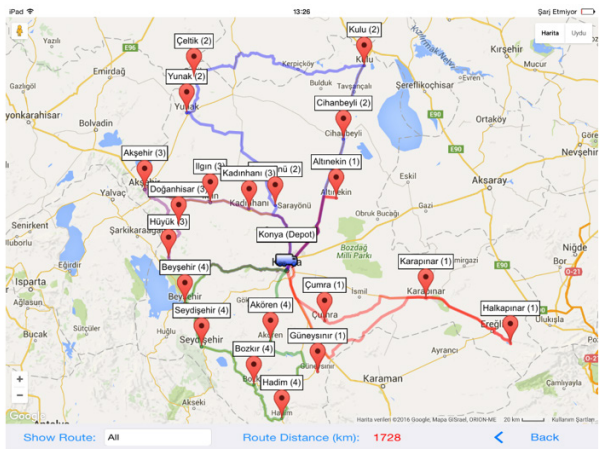
The Follow the Route window can be seen in Fig. 6. With this window, it is possible to monitor a desired route or routes starting from the current location in real time. The current location is retrieved with the help of GPS technology. Thus, the deviations from the routes can be identified immediately and new route suggestions can be made. The planned routes are commonly changed due to environmental and road conditions. Therefore, real-time applications are needed at the present time to avoid these conditions yet suffer minimal loss (time, distance). In this work, the associated window was developed to take consideration of this requirement. The new route suggestions, with regard to the deviations, are identified using the ACO algorithm. The minimum distance from the present location to the depot is determined. Of course, it is assumed that the unvisited demand points would be visited (in the order determined by the first route).

RESULTS AND DISCUSSION

The proposed algorithm was initially tested on benchmark datasets. For this task, 10 datasets were used, 5



(a) (b)



(c)

Figure 5. (a) Parameter Settings window (b) List of best routes obtained (c) Routes as polylines on the map.

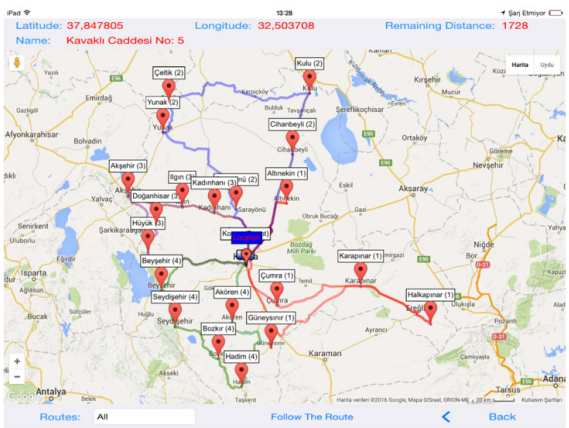


Figure 6. Follow The Route window.

from the suggestions of Augerat et al. [26] and 5 from the suggestions of Christofides and Eilon [27]. Those datasets include 1 depot, between 31 and 50 demand points and between 3 and 6 vehicles. The proposed algorithm was run 10 times for each dataset and the obtained best results were determined. These results are in Table 1. As seen in that chart, the algorithm performance was not greatly affected by the demand points or vehicles numbers. The actual impact is observed in terms of the distances to the cluster centers of the demand points, in other words membership values. In the proposed algorithm, a membership value is calculated for each demand point, these membership values are compared with a threshold value, and the customers to be added to the route enhancing pool are determined. The higher the threshold value, the higher the customer number to be added to the route enhancing pool. The threshold value also varies depending on the dataset used. For example, the datasets A-n32-k5 and E-n33-k4 have almost the same number of demand points. On the other hand, with the threshold value of 0.7, 22 customers for the A-n32-k5 dataset and

Table 1. The obtained results for the benchmark datasets.

Dataset Name	Vehicle Number	Demand Point Number	Threshold Value	Result	Best Known Optimal	Difference From Optimal (%)
A-n32-k5	5	31	0.861	810	784	3.32
A-n36-k5	5	35	0.843	825	799	3.25
A-n38-k5	5	37	0.824	746	730	2.19
A-n39-k6	6	38	0.786	849	831	2.16
A-n45-k6	6	44	0.747	980	944	3.81
E-n22-k4	4	21	0.407	385	375	2.66
E-n23-k3	3	22	0.860	576	569	1.23
E-n30-k3	3	29	0.961	549	534	2.81
E-n33-k4	4	32	0.816	851	835	1.91
E-n51-k5	5	50	0.706	539	521	3.45

26 customers for the E-n33-k4 dataset are assigned to the route enhancing pool. In this work, automatic detection method was used to remove this variability depending on the dataset property, to prevent threshold value entry to the algorithm at each run. In this method, the threshold value is automatically set so that half of the customers will be assigned in the route enhancing pool. The threshold values determined for each dataset is given in Table 1.

The mobile application based on the proposed algorithm was also tested on two different datasets formed using real-world data. The first set was formed by choosing 20 provinces, covering all the seven geographical regions of Turkey. For the second set, counties of the Konya province were considered. Twenty counties were chosen, covering the entire province. Konya province was chosen as the location of the depot in both the provinces and the counties. The amount of demand was determined as 10 for every customer. Fig. 7 illustrates the provinces and the counties in the datasets along with their locations.

Testing of the developed application was performed for a varying number of iterations, different population sizes and vehicle capacities. The number of clusters for the FCM clustering algorithm is calculated by considering the sum of the customer demands and vehicle capacities. Weight m was assumed as 2 and the terminating criterion ϵ as

Dataset - 1			Dataset - 2		
Customer Name	**Latitude**	**Longitude**	**Customer Name**	**Latitude**	**Longitude**
Konya (Depot)	37.8746429	32.4931554	Konya (Depot)	37.8746429	32.4931554
Ankara	39.9333635	32.8597419	Kulu	39.090245	33.080528
Antalya	36.8968908	30.7133233	Kadınhanı	38.240871	32.207868
Adana	36.9914194	35.3308285	Yunak	38.813456	31.735652
Kayseri	38.7204890	35.4825970	Ilgın	38.281724	31.913950
Gaziantep	37.0659530	37.3781100	Doğanhisar	38.145955	31.676750
Samsun	41.2797031	36.3360667	Beyşehir	37.679796	31.724299
Uşak	38.6742286	29.4058825	Seydişehir	37.419260	31.848216
Denizli	37.7830159	29.0963328	Akören	37.452598	32.371139
İzmir	38.4237340	27.1428260	Bozkır	37.188912	32.245393
Bursa	40.1885281	29.0609636	Çumra	37.573899	32.781038
İstanbul	41.0082376	28.9783589	Karapınar	37.715320	33.546570
Kastamonu	41.3766250	33.7764970	Halkapınar	37.433643	34.186855
Sivas	39.7505450	37.0150217	Akşehir	38.360258	31.417991
Siirt	37.9274040	41.9419780	Hadim	36.986063	32.455875
Diyarbakır	37.9249733	40.2109826	Güneysınır	37.267923	32.727420
Erzurum	39.9054993	41.2658236	Sarayönü	38.264661	32.407075
Van	38.5012085	43.3729793	Altınekin	38.308274	32.868888
Trabzon	41.0026969	39.7167633	Çeltik	39.023429	31.790552
Kars	40.6013378	43.0974525	Hüyük	37.951279	31.599333
Karaman	37.1810630	33.2222486	Cihanbeyli	38.656883	32.923303

Figure 7. The locations for the provinces and the counties in two datasets. The customer names state the names of the provinces and the counties..

Table 2. The provinces dataset results.

Mobile Device →			The Android Device		The IOS Device	
Iteration	Population	Capacity	Distance	Time	Distance	Time
100	20	40	10855	972.30	10764	463.54
		50	8446	1098.08	8446	541.78
		60	8065	1542.60	7987	731.86
		70	7512	1591.11	7389	747.23
	50	40	10554	2453.18	10554	1162.15
		50	8416	2705.32	8416	1352.60
		60	7945	2911.31	7902	1467.25
		70	7423	3012.13	7336	1519.61

Table 3. The counties dataset results.

Mobile Device →			The Android Device		The IOS Device	
Iteration	Population	Capacity	Distance (km)	Time (ms)	Distance (km)	Time (ms)
100	20	40	2021	950.04	1919	468.11
		50	1904	1143.61	1724	539.57
		60	1713	1481.02	1609	728.52
		70	1572	1465.65	1543	751.48
	50	40	1856	2397.12	1856	1137.43
		50	1734	2705.30	1711	1353.19
		60	1592	2764.32	1592	1408.34
		70	1557	2908.63	1527	1503.17

10^{-12}. While the factor α and the factor β were considered of 2, the pheromone evaporation value ρ as 0.5 for the ACO algorithm. Experiments were performed both on Android smartphone and IOS tablet devices separately. The results obtained by both devices are given in Table 2 and Table 3 in detail. For example, for the dataset of provinces (Table 2), if the iteration count was 100, the population size 20 and the vehicle capacity 40, a result of 10764 km is obtained with the IOS device. If the vehicle capacity was 50, the minimum distance to be travelled was found to be 8446 km. When the runtime for the algorithm is examined for the same values, it was 463.54 ms for a vehicle capacity of 40 and 541.78 ms for a vehicle capacity of 50. For the dataset of counties, if the number of iterations was 100, population size 20 and vehicle capacity 40, a result of 2021 km was obtained with the Android device. If the vehicle capacity was 50, the minimum distance to be travelled was found to be 1904 km. Runtimes for the algorithm using the same values was 950.04 ms for a vehicle capacity of 40 and 1143.61 ms for a vehicle capacity of 50.

As seen in the results for both the datasets for constant vehicle capacity, the iteration number and the population size does not greatly affect the results in terms of distance. But for a constant iteration count and population size, an increase in the vehicle capacity results in a partial decrease of the travel distance. In Fig. 8 (for the dataset of provinces), the best routes for every single vehicle created by the proposed algorithm is given by polylines of different colors - as an example, with a number of iteration of 100, a population size of 20 and a vehicle capacity of 50.

CONCLUSION

In this work, a real-time application was developed for CVRP, which can run on the mobile devices as IOS and Android. The real-world online data was operated in this application. The FCM clustering algorithm was used to group the demand points and the ACO algorithm was utilized for the best route selection in each group. Customer demand points and distances between these points

Figure 8. The best routes.

were obtained instantaneously via Google Places and Google Directions APIs. The deviations in the routes due to environmental and road conditions were identified immediately with the help of GPS technology and new route suggestions were made instantaneously.

The proposed algorithm was tested on 10 different benchmark datasets. Additionally, two different datasets were forearmed from real-world data, and tests were performed with Android and IOS devices separately. Test results showed that this application could be used in real time to identify and follow the optimum route.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest regarding the publication of this article.

AUTHOR CONTRIBUTION

This study were realized equally by the Authors.

References

1. Ballou RH. Business Logistics/supply Chain Management: Planning, Organizing, and Controlling the Supply Chain, Bus. Logist. (2004).
2. Rushton A, Croucher P, Baker P. The handbook of logistics and distribution management, Proj. Manag. J. (2006).
3. Dantzig GB, Ramser JH. The Truck Dispatching Problem, Manage. Sci. (1959).
4. Euchi J, Zidi S, Laouamer L. A Hybrid Approach to Solve the Vehicle Routing Problem with Time Windows and Synchronized Visits In-Home Health Care, Arab. J. Sci. Eng. 45 (2020) 10637–10652.
5. Häll CH, Andersson H, Lundgren JT, Värbrand P, Posada M. Vehicle Routing, Public Transp. 1 (2006) 573–586.
6. Bozyer Z, Alkan A, Fığlalı A. Cluster-First, Then-Route Based Heuristic Algorithm for the Solution of Capacitated Vehicle Routing Problem, Int. J. Informatics Technol. (2014).
7. Santos L, Coutinho-Rodrigues J, Antunes CH. A web spatial decision support system for vehicle routing using Google Maps, Decis. Support Syst. (2011).
8. Yu B, Yang ZZ, Yao B. An improved ant colony optimization for vehicle routing problem, Eur. J. Oper. Res. (2009).
9. Harmanani H, Azar D, Helal N, Keirouz W. A simulated annealing algorithm for the capacitated vehicle routing problem, Proc. ISCA 26th Int. Conf. Comput. Their Appl. CATA 2011, 2011.
10. Nazif H, Lee LS. Optimised crossover genetic algorithm for capacitated vehicle routing problem, Appl. Math. Model. (2012).
11. Zhang SZ, Lee CKM. An Improved Artificial Bee Colony Algorithm for the Capacitated Vehicle Routing Problem, in: Proc. - 2015 IEEE Int. Conf. Syst. Man, Cybern. SMC 2015, 2016.
12. Teoh BE, Ponnambalam SG, Kanagaraj G. Differential evolution algorithm with local search for capacitated vehicle routing problem, Int. J. Bio-Inspired Comput. (2015).
13. Mohammed MA, Ghani MKA, Hamed RI, Mostafa SA, Ibrahim DA, Jameel HK, Alallah AH. Solving vehicle routing problem by using improved K-nearest neighbor algorithm for best solution, J. Comput. Sci. 21 (2017) 232–240.
14. Rabbouch B, Saâdaoui F, Mraihi R. Empirical-type simulated annealing for solving the capacitated vehicle routing problem, J. Exp. Theor. Artif. Intell. 32 (2020) 437–452.
15. Altabeeb AM, Mohsen AM, Ghallab A. An improved hybrid firefly algorithm for capacitated vehicle routing problem, Appl. Soft Comput. J. 84 (2019) 105728.
16. İlhan İ. An improved simulated annealing algorithm with crossover operator for capacitated vehicle routing problem, Swarm Evol. Comput. (2021).
17. Osaba E, Yang XS, Del Ser J. Traveling salesman problem: a perspective review of recent research and new results with bio-inspired metaheuristics, in: Nature-Inspired Comput. Swarm Intell., (2020).
18. İlhan İ. An Application on Mobile Devices with Android and IOS Operating Systems Using Google Maps APIs for the Traveling Salesman Problem, Appl. Artif. Intell. 31 (2017) 332–345.
19. Toth P, Vigo D. Exact Solution of the Vehicle Routing Problem, in: Fleet Manag. Logist., (1998).
20. Dell'Amico M, Righini G, Salani M. A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection, Transp. Sci. (2006).

21. Toth P, Vigo D. An exact algorithm for the vehicle routing problem with backhauls, Transp. Sci. (1997).
22. Vigo D. A heuristic algorithm for the asymmetric capacitated vehicle routing problem, Eur. J. Oper. Res. (1996).
23. Bezdek JC, Ehrlich R, Full W. FCM: The fuzzy c-means clustering algorithm, Comput. Geosci. (1984).
24. Van Brummelen GR. Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry, (2013).
25. Dorigo M. Optimization, Learning and Natural Algorithms, Ph.D. Thesis, Politec. Di Milano, 1992.
26. Augerat P, Belenguer JM, Benavent E, Corberán A, Naddef D, Rinaldi G. Computational results with a branch-and-cut code for the capacitated vehicle routing problem, Inst. Syst. Comput. Anal. (1995).
27. Christofides N, Eilon S. An Algorithm for the Vehicle-Dispatching Problem, OR. (1969).