Vehicle Routing Based on Discrete Particle Swarm Optimization and Google Maps API

Fu-Shiung Hsieh

Hao wei Huang

VEHICLE ROUTING BASED ON DISCRETE PARTICLE SWARM OPTIMIZATION AND GOOGLE MAPS API

Fu-Shiung Hsieh, Chaoyang University of Technology, Taiwan, fshsieh@cyut.edu.tw
Hao wei Huang, Chaoyang University of Technology, Taiwan, s10127629@gm.cyut.edu.tw

ABSTRACT

Transportation imposes considerable cost on goods and has a significant influence on competitive advantage of a company. How to reduce the costs and improve the profit of a company is an important issue. Vehicle routing is a critical factor in reducing transportation costs. Finding optimal vehicle routes offers great potential to efficiently manage fleets, reduce costs and improve service quality. An effective scheme to manage fleets and determine vehicle routes for delivering goods is important for carriers to survive. In the existing literature, a variety of vehicle routing problems (VRP) have been studied. However, most papers do not integrate with GIS. In this paper, we consider a variant of VRP called Vehicle Routing Problem with Arbitrary Pickup and Delivery points (VRPAPD). The goal of this paper is to develop an algorithm for VRPAPD based on Google Maps API. To achieve this goal, we propose an operation model and formulate an optimization problem. In our problem formulation, we consider a set of goods to be picked up and delivered. Each goods has a source address and a destination address. The vehicles to transport the goods have associated capacities, including the maximal weight a vehicle can be carried and the maximal distance a vehicle can travel. The problem is to minimize the routes for picking up and delivering goods. The emerging Google Maps API provides a convenient package to develop an effective vehicle routing system. In this paper, we develop a vehicle routing algorithm by combining a discrete particle swarm optimization (DPSO) method with Google Maps API. We illustrate the effectiveness of our algorithm by an example.

Keywords: Vehicle routing problem, decision support system, discrete particle swarm, Google Maps API.

INTRODUCTION

Transportation imposes considerable costs of goods and has a significant influence on competitive advantage of a company. How to reduce the costs and improve the profit of a company is an important issue. Finding optimal vehicle routes offers great potential to efficiently manage fleets, reduce costs and improve service quality. An effective scheme to manage fleets and determine vehicle routes for delivering goods is important for carriers to survive. Development of a decision support system becomes an important issue for carriers and logistics companies. Development of an effective decision support system for vehicle routing relies on a seamless integration between the routing algorithms and an appropriate GIS system. However, most GIS-based systems have a drawback in availability, up-to-date cartography and high price. Google Maps API may overcome those limitations by providing a world wide access to cartography and to road/street network structures. In addition, a lot of important real data associated with roads and traffic restrictions such as one-way streets, prohibited left and U-turns, etc., is also provided. The capabilities provided by the Internet and Google Maps API open a promising avenue for developing web based decision support system for routing optimization problems. How to develop an effective vehicle routing algorithm and combine it with Google Maps API to seamless integrate with real GIS is an important issue.

In the existing literature, a variety of vehicle routing problems (VRP) have been studied. The classical vehicle routing problem (VRP) can be defined as the determination of an optimal set of routes for a fleet of vehicles to serve a set of customers [1]. The VRP is NP-hard, therefore, heuristic algorithms are required for tackling real-life instances [2]. The simplest and fastest methods are simple constructive heuristics like the classics proposed by Clarke and Wright, Mole and Jameson, and Gillett and Miller: see Laporte et al. [3] for a recent survey extended to tabu search algorithms, and Christofides et al. [4], for an earlier review. Metaheuristics provide much better solutions, especially on large-scale instances. Two excellent surveys for this very active research area are provided by Gendreau et al. [5] and by Golden et al. [6]. They show that the best metaheuristics for the VRP are powerful tabu search (TS) algorithms that easily outperform other metaheuristics like simulated annealing (SA), genetic algorithms (GA) and ant algorithms.

Several variants of VRP have been studied. The vehicle routing problem with backhauls (VRPB) is an extension of VRP. In the VRPB a set of capacitated vehicles are to be routed from a central depot to deliver and pickup goods at different customer locations [7]. For the linehauls, goods are loaded onto a vehicle at the depot and are dropped at the delivery points. Then, for the backhauls, new goods are collected at the pickup points and brought back to the depot. From a practical consideration, since it is often inconvenient to rearrange the delivery loads onboard to accommodate new pickup loads, the backhauls must necessarily be serviced after the linehauls in each route. The objective of VRPB is to minimize the number of vehicles and/or the total distance to service all customers, without violating the capacity and the maximum route time constraints of each vehicle. When all customers are linehauls (or backhauls), this problem reduces to the classical VRP. Gendreau et al. [8] applied a special insertion procedure that includes a local re-optimization of the route after the insertion of each customer. Goetschalckx and Jacobs-Blecha [9] used the space-filling curve heuristic to obtain an initial solution to the problem which is then improved using a number of local search heuristics. Exact algorithms for the VRPB are also proposed in [10, 11].

In [12], a variant of the VRPB called the vehicle routing problem with backhauls and time windows (VRPBTW) was studied.

In VRPBTW, each customer must be serviced within a specified time interval. In [13], it is shown that finding a feasible solution to the traveling salesman problem with time windows (TSPTW) is NP complete. Therefore, finding a feasible solution to the VRPBTW with a fixed number of vehicles is also NP-complete, since this problem is even more complex than the TSPTW. Another variant of the VRP is the vehicle routing problem with simultaneous pickup and delivery (VRPSPD) [14, 15]. In the VRPSPD, the vehicles are not only required to deliver goods to customer, but also pickup goods at customer locations. A general assumption in the VRPSPD is that all delivered goods must be originated from the depot and all pickup goods must be transported back to the depot. Delivery and pickup goods must be met simultaneously when each customer is visited only once by a vehicle and unloading is carried out before loading at the customers. The VRPSPD is an NP-hard problem [19], because it can be considered as the VRP, which is well-known NP-hard problem, when only delivery goods or pickup goods are considered.

The goal of this paper is to develop a solution algorithm for a class of VRP based on Google Maps API. In this paper, we consider a variant of VRP called Vehicle Routing Problem with Arbitrary Pickup and Delivery points (VRPAPD). The class of VRPAPD we study is more general than VRPSPD. In VRPAPD, we consider a set of goods to be picked up and delivered. Each vehicle that transports the goods has its associated capacity, including the maximal weight a vehicle can be carried and the maximal distance a vehicle can travel. Each goods has a source address and a destination address. The goods to be delivered may not be located in the depot originally. The destination of goods is a node other than the depot. Therefore, the above assumption is different from VRPSPD. To achieve this goal, we formulate an optimization problem for this class of VRP and propose a solution algorithm for it.

Our solution method consists of four parts: (1) computation of distance matrix for the transportation network, (2) representation of vehicle routes, (3) an algorithm to evaluate a solution and (4) a mechanism to evolve a solution. The first part is developed based on Google Maps API to compute the distance matrix for the transportation network. For the second part, we adopt the concept of permutation encoding to represent a solution. For each permutation, there is a corresponding giant tour, which is defined as the sequence of nodes that starts and ends with a depot. For the third part, to evaluate the performance of a solution (giant tour), we adopt the Split procedure proposed in [16]. For the fourth part, we adopt the discrete particle swarm optimization (DPSO) method to evolve and optimize the solution. We implement a system to support the decision of pickup and delivery routes. Our system generates the pickup and delivery routes based on the output of our algorithm to support the operations of carriers. By comparing the performance of our algorithm with that of VRPSPD, we demonstrate that our algorithm outperforms the existing algorithm.

The remainder of this paper is organized as follows. In Section 2, we describe the problem and propose our problem formulation. In Section 3, we present solution algorithms. In Section 4, we conduct experiments based on numerical examples. We conclude this paper in Section 5.

PROBLEM FORMULATION

To reduce transportation costs and improve profit, one way is to optimize vehicle routes. A mathematical model needs to be proposed to develop an algorithm for solving the optimization problem. In this section, we first define the notations required to formulate the optimization problem. Then we propose a solution methodology for solving it.

The vehicle routing problem considered in this paper is defined on a complete undirected network $G(N, E)$ with a node set N and an edge set E . Let $\boldsymbol{N} = \{0,1,2,..,N\}$. Node 0 is a depot with h identical vehicles of capacity Q . Each other node in $\boldsymbol{N} \setminus \{0\}$ may either represent a source node, a destination node or both depending on the goods to be delivered. The distance from node i to node j each is represented by c_{ij} . The distance between all pair of nodes is described by a distance matrix C . Let $\boldsymbol{M} = \{1,2,\ldots,M\}$ denote the set of all goods to be delivered. Each goods $m \in \boldsymbol{M}$ placed by a customer has a source node and a destination node. Each goods $m \in \boldsymbol{M}$ is described by several attributes, including v_m : the size of the goods, w_m : the weight of the goods, the source node of the goods, cs_m , and the destination node of the goods, cd_m .

Let $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ be the set of all routes for delivering goods. Let $\ell(\pi_k)$ be the cost for delivering goods along the route π_k . The total cost for traveling the set of all routes $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ to deliver goods is $f = \sum_{k=1}^{K} \ell(\pi_k)$.

The problem is to find a set of routes such that each route starts and ends at the depot, each customer is visited exactly once by one vehicle, the total vehicle load in any arc does not exceed the capacity of the vehicle and the total profit is maximized.

Several constraints need to be satisfied:

(1) Each route $\pi_k \in \Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ starts and ends at the depot.

(2) Each source node cs_m and destination node cd_m of goods $m \in \boldsymbol{M}$ is visited exactly once by one vehicle.
(3) The total vehicle load in any arc does not exceed the capacity of the vehicle.

Formally, the problem is to determine is a collection of routes $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ such that each node will be covered by exactly one route in Π and the total distance of the routes is minimized is formulated as follows:

Vehicle Routing Problem with Arbitrary Pickup and Delivery points (VRPAPD)

$$\min_{\pi_1,\pi_2,\ldots,\pi_K} \quad f = \left(\sum_{k=1}^{K} \ell(\pi_k)\right)$$

subject to Constraint (1), (2) *and* (3)

$\boldsymbol{M} = \{1,2,\ldots,M\}$: The set of all goods to be delivered.
m :The ID of a goods, where $m \in \boldsymbol{M}$.
$\boldsymbol{N} = \{0,1,2,..,N\}$: The set of all source and destination nodes.
n :The ID of a source/destination, where $n \in \boldsymbol{N}$.
C :A $(N+1)x(N+1)$ matrix, c_{ij} denotes the distance from node i to node j .
Q : The total load a vehicle can carry.
L : The total travel distance a vehicle can travel.
v_m : The size of goods m .
w_m : The weight of goods m .
cs_m : The source of goods m .
cd_m : The destination of goods m .
$\boldsymbol{K} = \{1,2,\ldots,K\}$:The set of IDs of all routes for delivering goods.
k :The ID of a route, $k \in \boldsymbol{K}$.
$\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$:The set of all routes for delivering goods.
$\ell(\pi_k)$:The cost for delivering goods along the route π_k .

DISCRETE PARTICLE SWARM OPTIMIZATION

Representation is an important issue in the development of meta-heuristics as it may affect the performance of the algorithms. Different problems need different data structures or representations to improve performance and efficiency. For VRP, a common representation method is permutation encoding, which represents the sequences of nodes to be visited. The tour associated with an instance of permutation encoding is called a giant tour. We use giant tours, i.e. sequences of nodes without trip delimiters, to represent solutions for our problem. For a problem with a set of nodes $\boldsymbol{N} = \{0,1,2,..,7\}$, Fig. 1 shows an example of giant tour is 0,1,2,3,4,5,6,7. As the overall distance of a giant tour may exceed the maximal distance a vehicle can travel or the capacity a vehicle can carry, it is usually necessary to divide a giant tour into a number of routes such that each route can be visited by a vehicle without exceeding its travel distance limit or capacity. One well-known method to divide a number of routes is the Split procedure proposed by Prins [16]. We adapt Split procedure, as a decoding scheme, developed by Prins [16] for the capacitated VRP. In the following, the adapted Split procedure is briefly explained. Fig. 2 shows how Split procedure works. In Fig. 2, three routes are obtained by applying Split procedure to Fig. 1. Each route is assigned to a single vehicle to deliver and pickup goods.

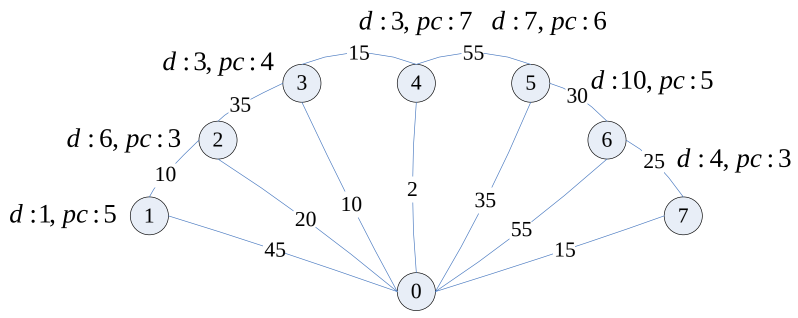

Figure 1 A Giant Tour

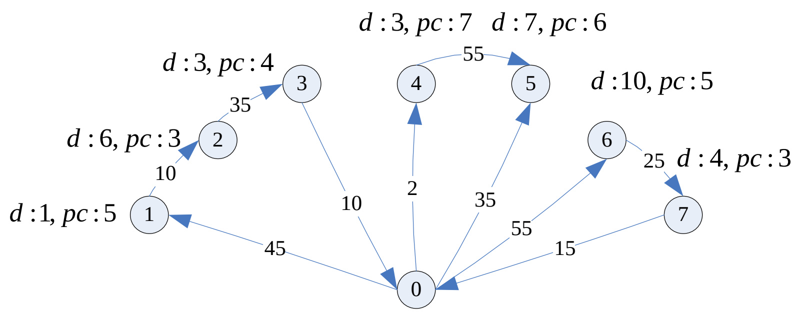

Figure 2 Three routes obtained by applying the Split procedure

Although one may evaluate the value of the fitness function by applying Split procedure to a given giant tour, we need a mechanism to evolve the giant tour to find a good solution. In this paper, we apply a discrete particle swarm optimization (DPSO) method to determine a good giant tour. Let I be the number of particles in the DPSO method. Each $i \in \{1,2,\ldots,I\}$ denotes a particle i. Let Z_i^t denote the giant tour associated with particle i at time t. Applying the Split procedure to Z_i^t will lead to a set of routes $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$. The set of routes $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ obtained by applying the Split procedure to a given giant tour make it possible to evaluate the value of the fitness function f. The inputs of the DPSO algorithm include C, Q, L, M, N, PC, D. The outputs of the DPSO algorithm is $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$.The DPSO algorithm used in this paper is as follows.

```
Discrete Particle Swarm Optimization (DPSO) Algorithm
Input: $C, Q, L, M, N, PC, D$
Output: $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$
Step 1:
$t \leftarrow 0$
Generate initial population of swarm, $Z^t$
Evaluate each particle $Z_i^t$ in $Z^t$
Determine the personal best of each particle in $P^t$
Determine the global best of swarm, $G^t$
While (stopping criteria is not satisfied)
Do
$t \leftarrow t+1$
For each $i \in \{1,2,\ldots,I\}$
Update particle $i$ as follows
$Z_i^{t+1} = c_2 \otimes F_3(c_1 \otimes F_2(w \otimes F_1(Z_i^t), P_i^t), P_g^t)$, where
where $\lambda_i^t = w \otimes F_1(Z_i^t)$ is the velocity of the particle, $F_1$ is the mutation operator applied to $Z_i^t$ with probability of $w$ and $\lambda_i^{t+1}$ is obtained as follows:
$\lambda_i^{t+1} = \begin{cases} F_1(Z_i^t) & if\ u \le w \\ Z_i^t & otherwise \end{cases}$
, where $u$ is a random variable with uniform distribution $U(0,1)$.
$\delta_i^{t+1} = \begin{cases} F_2(\lambda_i^t, P_i^t) & if\ u_1 \le c_1 \\ \lambda_i^t & otherwise \end{cases}$
, where $u_1$ is a random variable with uniform distribution $U(0,1)$.
$Z_i^{t+1} = \begin{cases} F_3(\delta_i^t, P_g^t) & if\ u_2 \le c_2 \\ \delta_i^t & otherwise \end{cases}$
, where $u_2$ is a random variable with uniform distribution $U(0,1)$.
Evaluate each particle in $X^t$
For each particle
    Update personal best $P_i^t$ as follows:
```

$$P_i^t = \begin{cases} Z_i^t \quad if \ f(Z_i^t) \le f(P_i^{t-1}) \\ Z_i^t \quad if \ f(Z_i^t) > f(P_i^{t-1}) \ and \ u \le e^{-\frac{f(Z_i^t)-f(P_i^{t-1})}{f(P_i^{t-1})}} \\ P_i^{t-1} \ otherwise \end{cases}$$

```
Update global best $G^t$
$T \leftarrow \alpha T$
Return $G^t$
```

Different position updating rules for permutation encoding have been proposed in the literature [17], [18]. In this paper, we implement a position updating rule proposed by Pan et al. [17]. According to this rule, the position of a particle at iteration t is updated by considering possible three choices. These are: (i) following its own position, (ii) going towards its personal best position, and (iii) going towards the best position of the particle in the entire swarm population. Thus, the updating equation $X_i^{t+1} = c_2 \otimes F_3(c_1 \otimes F_2(w \otimes F_1(X_i^t), P_i^t), P_g^t)$ is used in DSPO algorithm.

The main algorithm for solving the problem formulated in this paper is as follows:

```
Algorithm for solving Vehicle Routing Problem with Arbitrary Pickup and Delivery points (VRPAPD)
Input: $\boldsymbol{M}, \boldsymbol{N}, Q, L$
Output: $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$
Step 0: Compute $C$ according to $N$ based on Google Maps API
Step 1: Compute $PC$ and $D$ according to $\boldsymbol{M}$
Step 2: Apply DPSO
        $\Pi_h \leftarrow \text{DPSO}(C, PC, D, Q, L, M, N)$
        $\Pi \leftarrow \Pi \cup \Pi_h$
Step 3: Check all delivered goods
     $M_d \leftarrow Deliver(\Pi_h, M, N)$
Step 4: Find
        $M \leftarrow M \setminus M_d$
        If ($M \ne \Phi$)
          $h \leftarrow h+1$
          Go to Step 2
        Else
           Exit
        End If
```

CASE STUDY

In this section, we verify our algorithms by a numerical example.

Example: Consider a carrier that has received M =20 goods from N =21 source nodes (not including the depot) to be delivered to their destination nodes. The carrier has a depot located at node 0. Each goods has a source node and destination node. The data of the goods and stores are listed in Table 1 and Table 2, respectively. The parameters of the DPSO algorithm are shown in Table 3. The maximal distance a vehicle can travel is limited to 102 km. The maximal weight for a vehicle is 22.

Table 1 Goods to be delivered

ID	v	w	cs (Node number)	cd (Node number)
0001	88.0	13169.0	957539	967352
0002	85.0	16254.0	967352	132758
0003	39.0	17302.0	132758	121231
0004	99.0	11332.0	121231	955681
0005	64.0	10789.0	955681	970864
0006	90.0	5606.0	970864	113425

0007	96.0	17750.0	113425	923307
0008	79.0	18700.0	923307	980207
0009	73.0	10393.0	980207	982568
0010	60.0	4208.0	982568	921459
0011	85.0	5905.0	921459	991159
0012	82.0	6007.0	991159	954758
0013	84.0	7291.0	954758	923558
0014	46.0	3673.0	923558	981657
0015	65.0	18328.0	981657	910530
0016	38.0	3414.0	910530	993018
0017	75.0	10498.0	993018	110884
0018	68.0	3044.0	110884	136510
0019	21.0	12188.0	136510	120847
0020	63.0	7355.0	120847	992358

Table 2 Location of nodes

ID	Node number	Latitude	Longitude
0	0000	24.0721568	120.71686629999999
1	970864	24.143739	120.67505600000004
2	113425	24.1454804	120.69221540000001
3	957539	24.085405	120.70822999999996
4	921459	24.1550609	120.73342060000004
5	121231	24.113147	120.66166999999996
6	110884	24.1627714	120.6533187
7	993018	24.1859803	120.66410250000001
8	120847	24.1041933	120.62461209999992
9	967352	24.0587358	120.69636390000005
10	981657	24.2687626	120.75018009999997
11	923307	24.1275322	120.69488580000007
12	980207	24.1048878	120.69592520000003
13	136510	24.1180451	120.627521
14	923558	24.2742643	120.78019760000006
15	954758	24.1941553	120.80658289999997
16	992358	24.2531006	120.52813229999992
17	132758	24.0628538	120.6628604
18	910530	24.2487509	120.69933400000002
19	982568	24.1445	120.72516700000006
20	955681	24.135268	120.657103
21	991159	24.1924759	120.746936

Table 3 Parameters

Parameters									
M	N	Q	L	I	w	c1	c2	PVND	T
20	22	22.0	102.0	10	0.999	0.999	0.999	0.999	100

The resulting routes found by our algorithm are as follows:

Routes obtained from Iteration 1									
0	12	3	9	8	10				
0	17								
0	21	16	5	20	2	1			
Routes obtained from Iteration 2									
0	12	11	2	19	21	15	14	10	18
0	20	13	8	16					

Total distance of the routes is 374.429.
The total distance obtained by our algorithm is less than that of VRPSPD.

CONCLUSION

Vehicle routing is a critical factor in reducing transportation costs. In the existing literature, a variety of vehicle routing problems (VRP) have been extensively studied. The VRP is NP-hard, therefore, heuristic algorithms are required for tackling real-life instances. Drawbacks of the existing methods include (1) the lack of integration with GIS and (2) the simplified models used. Most methods do not seamlessly integrate with GIS. Furthermore, a general assumption in the vehicle routing problem with simultaneous pickup and delivery (VRPSPD) is that all delivered goods must be originated from the depot and all pickup goods must be transported back to the depot. Although such assumption may lead to a simplified problem that can be solved more easily, the performance may be sacrificed due to the assumption. In this paper, we consider a variant of VRP called Vehicle Routing Problem with Arbitrary Pickup and Delivery points (VRPAPD). In this problem, we consider a set of goods to be picked up and delivered. Each goods has a source address and a destination address. The goods to be delivered may not be located in the depot originally. The destination address of goods may be a node other than the depot. Therefore, the above assumption of VRPAPD is different from VRPSPD. Each vehicle that transports the goods has finite capacity, including the maximal weight a vehicle can be carried and the maximal distance a vehicle can travel. Based on the above assumption, we propose an algorithm based on DPSO method. Our system generates routes and detailed individual vehicle routes based on Google Maps API and the inputs, which consist of capacities of car, the location of the depot, sources and destinations of goods to be delivered. We illustrate the effectiveness of our algorithm by an example.

ACKNOWLEDGEMENT

This paper is supported in part by Ministry of Science and Technology, Taiwan, under Grant NSC102-2410-H-324-014-MY3.

REFERENCES

[1] Dantzig, G. B., & Ramser, J. H. (1959) ‘The truck dispatching problem’, *Management Science*, Vol. 6, No. 1, pp.80–91.

[2] Toth P, Vigo D. (1998) ‘Exact solution of the vehicle routing problem In: Crainic TG, Laporte G’, *Fleet management and logistics*, Dordrecht: Kluwer, pp. 1–31.

[3] Laporte G, Gendreau M, Potvin JY, Semet F. (2000) ‘Classical and modern heuristics for the vehicle routing problem’, *International Transactions in Operational Research*, Vol. 7, pp.285–300.

[4] Christo5des N, Mingozzi A, Toth P. (1979) ‘The vehicle routing problem. In: Christofides N, Mingozzi A, Toth P, Sandi C’, *Combinatorial optimization*. New York: Wiley, pp. 315–38.

[5] Gendreau M, Laporte G, Potvin J-Y. (1998) ‘Metaheuristics for the vehicle routing problem’, *GERAD research report G-98-52*, Montreal, Canada.

[6] Golden BL, Wasil EA, Kelly JP, Chao IM. (1998) ‘The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: Crainic TG, Laporte G’, *Fleet management and logistics*, Dordrecht: Kluwer, pp. 33–56.

[7] D.O. Casco, B. L. Golden and E. A. Wasil (1988) ‘Vehicle routing with backhauls: models, algorithms and case studies’, *In Vehicle Routing: Methods and Studies* (Edited by B. Golden and A. Assad, pp.127-147. North-Holland, Amsterdam.

[8] Michel Gendreau, Alain Hertz, Gilbert Laporte (1996) ‘The Traveling Salesman Problem with Backhauls’, *Computers & Operations Research*, Vol.23, No.5, pp. 501-508.

[9] Marc Goetschalckx, Charlotte Jacobs-Blecha (1989) ‘The vehicle routing problem with backhauls’, *European Journal of Operational Research*, Vol.42, No.1, pp.39–51.

[10] B. Kalantari, A.V. Hill and S.R. Arora (1985) ‘An algorithm for the traveling salesman problem with pickup and delivery customers’, *European Journal of Operational Research*, Vol. 22, No.3, pp.377-386.

[11] Daniele Vigo, Paolo Toth (1997) ‘An Exact Algorithm for the Vehicle Routing Problem with Backhauls’, *Transportation Science*, Vol. 31, No.4, pp. 372 - 385.

[12] Sam R. Thangiah, Jean-Yves Potvin, Tong Sun (1996) ‘Heuristic approaches to vehicle routing with backhauls and time windows’, *Computers & Operations Research*, Vol. 23, No.11, pp.1043-1057.

[13] M. W. P. Savelsbergh (1985) ‘Local search in routing problems with time windows’, *Annals of Operations Research*, Vol. 4, No.1, pp. 285-305.

[14] Fatma Pinar Goksal, Ismail Karaoglan, Fulya Altiparmak (2013) ‘A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery’, *Computers & Industrial Engineering*, Vol. 65, No.1, pp.39–53.

[15] Chen, J. F., & Wu, T. H. (2006) ‘Vehicle routing problem with simultaneous deliveries and pickups’, *Journal of the Operational Research Society*, Vol. 57, pp.579–587.

[16] Prins, C. (2004) ‘A simple and effective evolutionary algorithm for the vehicle routing problem’, *Computers & Operations Research*, Vol. 31, pp. 1985–2002.

[17] Pan, Q. K., Tasgetiren, M. F., & Liang, Y. C. (2008) ‘A discrete particle swarm optimization algorithm for the no-wait flow shop scheduling problem’, *Computers & Operation Research*, Vol.35, No.9, pp.2807–2839.

[18] Liao, C. J., Tseng, C. T., & Luarn, P. (2007) ‘A discrete version of particle swarm optimization for flow shop scheduling problems’, *Computers & Operations Research*, Vol.34, No.10, pp.3099–3111.

[19] Salhi, S., & Nagy (1999) ‘A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling’, *Journal of the Operational Research Society*, Vol.50, No.10, pp.1034–1042.