International Journal of Industrial Engineering & Production Research

June 2014, Volume 25, Number 2
pp. 139-149

http://IJIEPR.iust.ac.ir/

pISSN: 2008-4889

A Navel Heuristic Algorithm for the Periodic Vehicle Routing Problem

M. Alinaghian*

M.Alinaghian, *Department of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran*

KEYWORDS

Periodic vehicle routing problem;
Heuristic algorithms;
Data mining

ABSTRACT

Periodic vehicle routing problem focuses on establishing a plan of visits to clients over a given time horizon so as to satisfy some service level while optimizing the routes used in each time period. This paper presents a new effective heuristic algorithm based on data mining tools for periodic vehicle routing problem (PVRP). The related results of proposed algorithm are compared with the results obtained by best Heuristics and meta-heuristics algorithms in the literature. Computational results indicate that the algorithm performs well in terms of accuracy and solution time.

1. Introduction

The classical vehicle routing problem (*VRP*) is defined as follows: vehicles with a fixed capacity *C* must deliver order quantities d_i (*i=1,..N*) of goods to *N* customers from a single depot (*i=0*). Knowing the distance d_{ij} between customers *i* and *j* (*i,j=1,..,N*), the objective of the problem is to minimize the total distance traveled by the vehicles in such a way that only one vehicle handles the deliveries for a given customer and the total quantity of goods that a single vehicle delivers do not be larger than *C* (Drummond et al., 2001). The typical planning period in a classical VRP is a single day.

The period vehicle routing problem (*PVRP*) generalizes the classical *VRP* by extending the planning period to *D* days. The classical *PVRP* consists of a homogeneous vehicle fleet (vehicles with same capacities) which must visit a group of customers from a depot where the vehicles must start and return to at the end of their journeys. Each vehicle has a fixed capacity that cannot be exceeded and each customer has a known daily demand that must be completely satisfied in only one visit by exactly one vehicle. The planning period is *D* days. Each customer in *PVRP* must be visited S_i *(i=1,..N)* times, where $1 \le S_i \le D$. In the classical model of *PVRP*, the daily demand of a customer is always fixed. The *PVRP* can be seen as a problem of generating a group of routes for each day so that the constraints involved are satisfied and the global costs are minimized.

In PVRP problem, Each customer $i \in \{1,2,..,N\}$ specifies a set S_i of combinations, and the visit days are assigned to the customer by selecting one of these combinations. The vehicles must, thus, visit the customer *i* on the days belonging to the selected combination. For example, in a 6-day planning period, if the customer *i* specifies the two visit day combinations {1, 3, 5} and {2, 4, 6}, then the vehicles must visit the customer *i* on the days 1, 3 and 5 if the combination {1, 3, 5} is selected while selecting combination two means the vehicles must visit the customer *i* on the days 2, 4 and 6 (Alegre. et al. 2007).

In this paper, an effective heuristic algorithm has been developed based on data mining tools, for the PVRP. The algorithm has a significant ability in solving PVRPs in short amount of time compared with existing algorithms in the literature. This paper is organized as follows: second part is dedicated to a review of the literature in third section, the mathematical model of

* **Corresponding author: Mehdi Alinaghian**
Email: *Alinaghian@cc.iut.ac.ir*
Paper first received Nov. 25, 2012, and in accepted form May 29, 2013.

proposed problem is described. In section 4th and before explaining the heuristic algorithm, tools and algorithms used as important parts of the proposed algorithm are presented. Next, in the 5th section proposed heuristic algorithm (AD) is described. The results obtained from solving 13 test problems taken from literature and the result analysis are discussed in section 6 and section 7 discusses the conclusions.

2. Literature Review

Early formulations of the PVRP were developed by Beltrami and Bodin (1974) and by Russell and Igo (1979) who proposed heuristics applied to waste collection problems. Christofides and Beasley (1984) first formulated the problem mathematically and then presented a heuristic that assigns a priority level to each customer and, according to this priority, assigns a visiting schedule to each customer so that the increase of an estimation of the costs of the vehicle routing problems (VRPs) related to the schedules is minimized. Tan and Beasley (1984) use the idea of the generalized assignment method proposed by Fisher and Jaikumar (1981) and assign a visiting schedule to each vertex. Eventually a heuristic for the VRP is applied to each day.

Russell and Gribbin (1991) developed a heuristic organized in four phases. Solution methods in these papers have focused on two-stage (construction and improvement) heuristics. A heuristic for a variant of the PVRP, where the objective is to minimize the number of vehicles, is proposed by Gaudioso and Paletta (1992). Similarly, good results were obtained in the more recent work of Drummond et al. (2001), who implemented parallel genetic algorithms.

Alegre. et al. (2007) applied scatter search for solving the periodic vehicle routing problem. Authors used from scatter search only for optimizing the combination of customers. They applied two local search algorithms, OR interchange and CROSS, for dailies tours. Angelelli and Speranza (2002) suggest a tabu search algorithm for solving one special kind of problem. Their problem is near to multi depot periodic vehicle routing problem. Francis et. al. (2007) suggested two indexes, operational flexibility and complexity, for evaluating the solutions and the trade-off between them.

Hemmelmayr et al. (2009) present variable neighborhood search heuristic to solve problem. They used their algorithm for PTSP. The first contribution of their study was using VNS for PVRP and the second one was the best results they obtained in some test problems.

In this paper, a new heuristic algorithm has been developed based on clustering, applied to the PVRP. The algorithm has a significant ability in solving PVRPs and its noticeable point is that it can reach to the answer in quite small amount of time.

The strategy that used in this paper can be categorized as cluster-first route-second strategy. Some researchers applied this method (Fisher & Jaikumar, 1981; Gillett & Johnson, 1976.) In this strategy and in the first step, customers are clustered in to clusters and in the second step a constructing method is applied for determining the visit's order of tour's customers on each cluster considering the minimum cost for each rout. There are three types of cluster-first, route-second methods. The first category involve The simplest methods, like sweep algorithm (Gillett and Miller [20]) and Fisher and Jaikumar algorithm (17) . The algorithms in this category generally perform a simple clustering of the customers and use from the results to determine vehicle routes.

The second ones are based on Truncated Branch-and-Bound (Christofides, Mingozzi, and Toth [10]) and in third category , named petal algorithm (Balinski and Quandt [3], Foster and Ryan [18]) in the first step produces a number of overlapping clusters and selects from them a suitable set of routes.

A very recent paper in PVRP is paper of Pacheco et al. (2011) in which the authors study the problem of a bakery company in northern Spain. The objective of problem is minimizing total distance traveled for the daily routes over the week. The subject that distinguished the proposed problem with classical PVRP is existence of some flexibility in the dates of delivery. A new mixed-integer linear model are presented and solve the problem through a two phase algorithm. In the first phase of algorithm, a set of good and diverse solutions is generated, applying GRASP. In the second phase, path-relinking is applied for improving the solutions. Computational experiments are executed on real-data-based instances. Also the authors applied the necessary modifications to treat the problem as a PVRP, and then a complete compression is considered with algorithms in the literature. Recently, Vidal et al ,(2012) propose a new hybrid genetic algorithm in which individuals representing solutions and infeasible solutions are allowed in the first stages of algorithm. Different operators with the target of having good quality solutions are applied in each iteration and diversity of solutions in all iterations maintained respecting to specified value. They test the algorithm on benchmark instances for the PVRP and the MDVRP and set of instances for the multi-depot periodic vehicle routing problem (MDPVRP), the results shows good result. A complete and recent survey on the PVRP and its extensions was written by Francis et al(2008).

3. Model Formulation

Let G= (V,A) be a graph, where $V=\{v_1,v_1,...,v_N\}$ is the nodes set, and $A=\{(v_i,v_j): v_i,v_j \in V\}$ is the arcs set where each arc (v_i, v_j) is associated with a non-negative cost C_{ij}. Followings are a number of parameters used in the presented mathematical model.

3.1. Parameters

N Number of nodes (i.e., customers)

D The days set in the period

S_i the total number of combinations of node i

d_i Demands of node i in each day of combination (i=1,…,N)

nv Total number of vehicles

C Capacity of each vehicle.

t_i Required time for servicing the customer (node) i

t_{ij} Required time for traveling between customers i and j

B Arbitrary large number.

C_{ij} Travel cost (i.e., length of the arc) between customers i and j

L Maximum time that each vehicle can be used in each day.

$S = \{i \mid i = 1,\ldots,N\}$ Collection of nodes

3-2. Decision Variables

The decision variables are as follows:

$x_{ij\kappa}^{\delta} = \begin{cases} 1 & \text{If vehicle } k \text{ travels through route } [i\ ,\ j] \text{ in day of } \delta \\ 0 & \text{Otherwise.} \end{cases}$

$z_i^s = \begin{cases} 1 & \text{If s - th combination of node i is selected to serve} \\ 0 & \text{Otherwise.} \end{cases}$

$v_i^{\delta} = \begin{cases} 1 & \text{If the vehicle serves node } i \text{ in day } \delta \\ 0 & \text{Otherwise.} \end{cases}$

$a_{\delta si} = \begin{cases} 1 & \text{If day of } \delta \text{ is in s-th combination of node } i \\ 0 & \text{Otherwise.} \end{cases}$

3-3. Mathematical Model

Now, the problem can be mathematically formulated as follows.

$$Min\ Z_1 = \sum_{\delta=1}^{D}\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{\kappa=1}^{K} C_{ij} x_{ij\kappa}^{\delta} \quad (1)$$

$$\sum_{s=1}^{S_i} z_i^s = 1 \quad \forall\ i=1,2,..,N \quad (2)$$

$$v_i^{\delta} = \sum_{s=1}^{S_i} z_i^s . a_{\delta si} \quad \forall\ \delta = 1,2,..,D,\ i=1,2,..,N \quad (3)$$

$$\sum_{\kappa=1}^{K} x_{ij\kappa}^{\delta} \le \frac{v_i^{\delta} + v_j^{\delta}}{2} \quad \forall\ \delta = 1,2,..,D,\ i,j=0,1,..,N \quad (4)$$

$$\sum_{j=0}^{N} x_{ij\kappa}^{\delta} = \sum_{j=0}^{N} x_{ji\kappa}^{\delta} \quad \forall\ \kappa = 1,2,..,K,\ i=0,1,..,N,\ \delta = 1,2,..,D \quad (5)$$

$$\sum_{i=1}^{N} x_{i0\kappa}^{\delta} = 1 \quad \forall\ \delta = 1,2,..,D,\ \kappa = 1,2,..,K \quad (6)$$

$$\sum_{i=1}^{N} d_i \sum_{j=0}^{N} x_{ijk}^{\delta} \le C \qquad k = 1,2,...,K,\ \delta = 1,2,...,D \quad (7)$$

$$\sum_{i=0}^{N}\sum_{j=0}^{N} (t_i + t_{ij}).x_{ij\kappa}^{\delta} \le L \quad \forall\ \kappa = 1,2,..,K,\ \delta = 1,2,..,D \quad (8)$$

$$\sum_{i,j\in Q} x_{ij\kappa}^{\delta} \le |Q| - 1 \quad \forall\ Q \subseteq N,\ 2 \le Q,\ \delta = 1,2,...,D \quad (9)$$

$$z_i^s,\ x_{ij\kappa}^{\delta}, v_i^{\delta}, a_{\delta si} \in \{0,1\} \quad \forall\ i,j = 1,2,...,N,\ \delta = 1,2,...,D,\ \kappa = 1,2,..,K,\ s = 1,2,..,S$$
$$\delta \in D,\ i \in N$$

In this formulation, the objective function (Z1) minimizes the transportation cost. Constraint (2) ensures that among combinations for each customer only one combination is selected. Constraint (3) is related to select a day for serving to customer and constraint (4) states that the arc i-j in day of δ could be used if this day be in combination selected day of node i,j. Constraint (5) states if a vehicle arrives at a node (i.e., customer's location), it should leave it and constraint (6) ensures that depot is first location and final destination of each vehicle. Constraint (7) states maximum capacity that a vehicle can carry. Constraint (8) is related to maximum travel time and constraint (9) prevents from constructing a loop in each day.

4. The AD Heuristic Algorithm

In this section and before explaining the heuristic algorithm, K-means clustering and Clarke and Wright algorithm are explained and the VRPSOLVER software is analyzed. These methods are used in some

parts of the proposed heuristic algorithm, and then the heuristic algorithm is explained.

K-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem.(Han and & Kamber, 2001) The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define *k* cores, one for each cluster.

These cores should be placed in a cunning way since different location causes different result. So, the better choice is to place them as much far from each other as possible. The next step is to take each point belonging to a given data set and associate it to the nearest core. When no point is pending, the first step is completed and an early grouping is made. At this point we need to re-calculate k new cores as new centers of the clusters resulting from the previous step. After we have these k new cores, a new binding has to be done between the same data set points and the nearest new core. A loop has been generated.

As a result of this loop we may notice that the k cores change their location step by step until no more changes are done. In other words cores do not move any more. Figure 1 presents a simple view of how the algorithm makes clusters.

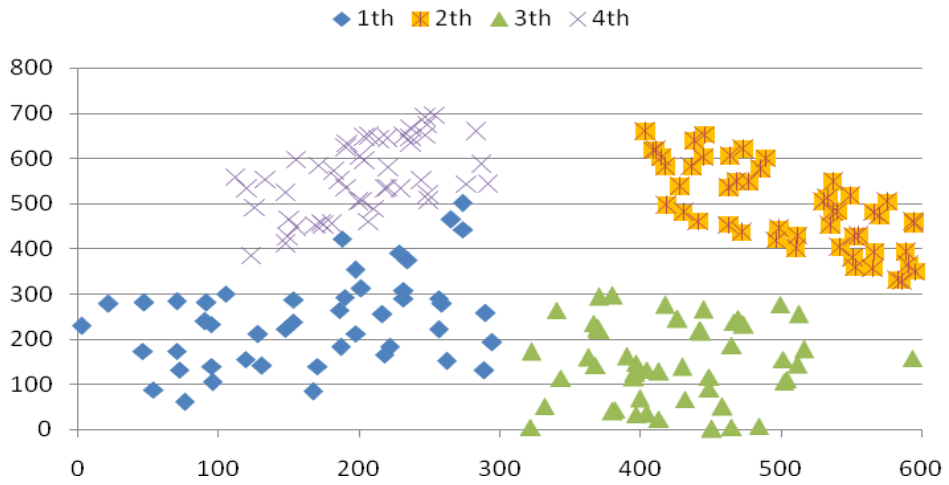


Fig. 1. an example of cluster customers to four parts.

The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intra cluster similarity is high but the inter cluster similarity is low.

The most famous heuristic to solving the *VRP* problem was proposed by Clarke and Wright (1964) and is based on the concept of saving, an estimate of the cost reduction obtained by serving two customers sequentially in the same route, rather than in two separate ones. If *i* is the last customer of a route and *j* is the first customer of another route, the associated saving is defined as $sij = ci0 + c0j - cij$. If *sij* is positive, then serving *i* and *j* consecutively in a route is profitable. The Clarke and Wright algorithm considers all customer pairs and sorts the savings in non-increasing order. Starting with a solution in which each customer appears separately in a route, the customer pair list is examined and two routes are merged whenever this is feasible. A lot of works have been arisen on literature to improve the classical version of the algorithm and a parallel and a sequential version are available. Gaskell (1967) and Yellow (1970) proposed modified version of algorithm and add extra parameter for calculating the saving formula of the algorithm. This parameter was called route shape parameter (λ).Golden, et al. (1977) report that the best setting of this parameter to get good results is $\lambda = 0.4$ or 1.0. Also Various effort have been done by a number of authors to speed up computation time (See Nelson et al. (1985) & Paessens(1988)). VRPSOLVER is developed by Snyder in Lehigh University. The software applies adaptation of the Clarke-Wright savings algorithm to solve VRP instances. Thorough information and the free version are available at http://www.lehigh.Edu/~lvs2/ download/ vrpsolver html. In the final phase of our algorithm, the software is employed to solve daily VRPs. Unfortunately there were no data on estimated error of the software; therefore, we solved the CVRP (Capacitated Vehicle Routing Problem) instances from Augerat, et al. available at http://www.branchandcut.org/VRP/data with the initial adjustments of the software. Table 1 shows the related solutions.

According to the result, the software has nearly %2 error. This error is calculated by $[(\text{ANSWER} - \text{BEST}) \times 100] / \text{BEST}$. In this formulation the ANSWER is the best answer obtained from the software, and the BEST is the best answer found in the literature.

Tab. 1. The results of vrpsolver, solving the Augerat, et al. test problems.

Instance	#Cu.	#Veh.	Tigh.	Opt.Sol	Vrp-Solver	
					BEST	ER
A-n32-k5.vrp	31	5	0.82	784.00	809.33	0.03
A-n33-k5.vrp	32	5	0.89	661.00	666.00	0.01
A-n33-k6.vrp	32	6	0.9	742.00	742.00	**0.00***
A-n34-k5.vrp	33	5	0.92	778.00	790.37	0.02
A-n36-k5.vrp	35	5	0.88	799.00	806.00	0.01
A-n37-k5.vrp	36	5	0.81	669.00	693.65	0.04
A-n37-k6.vrp	36	6	0.95	949.00	962.00	0.01
A-n38-k5.vrp	37	5	0.96	730.00	747.59	0.02
A-n39-k5.vrp	38	5	0.95	822.00	848.00	0.03
A-n39-k6.vrp	38	6	0.88	831.00	839.73	0.01
A-n44-k6.vrp	43	6	0.95	937.00	943.48	0.01
A-n45-k6.vrp	44	6	0.99	944.00	960.53	0.02
A-n45-k7.vrp	44	7	0.91	1146.00	1158.77	0.01
A-n46-k7.vrp	45	7	0.86	914.00	926.45	0.01
A-n48-k7.vrp	47	7	0.89	1073.00	1100.78	0.03
A-n53-k7.vrp	52	7	0.95	1010.00	1047.32	0.04
A-n54-k7.vrp	53	7	0.96	1167.00	1186.05	0.02
A-n55-k9.vrp	54	9	0.93	1073.00	1082.54	0.01
A-n60-k9.vrp	59	9	0.92	1354.00	1369.92	0.01
A-n61-k9.vrp	60	9	0.98	1034.00	1048.96	0.01
A-n62-k8.vrp	61	8	0.92	1288.00	1328.09	0.03
A-n63-k9.vrp	62	9	0.97	1616.00	1644.57	0.02
A-n63-k10.vrp	62	10	0.93	1314.00	1331.26	0.01
A-n64-k9.vrp	63	9	0.94	1401.00	1428.96	0.02
A-n65-k9.vrp	64	9	0.97	1174.00	1209.54	0.03
A-n69-k9.vrp	68	9	0.94	1159.00	1179.90	0.02
A-n80-k10.vrp	79	10	0.94	1763.00	1804.70	0.02
average						**0.02**

*. The algorithm reached to the optimal solution

5. AD Heuristic Algorithm Structure

This algorithm is trying to simultaneously construct $m \times v$ tours in which m and v are respectively representing the number of days and the number of vehicles. To form the tours, the algorithm puts the adjacent customers in the same tour. In the first step the coordination of depot changed to the zero point of coordination system by subtracting the coordination of depot from nodes coordination. In the next step the coordination of the customers is changed from Cartesian to polar in which the value of radian is the base of distance. In the next step which let the procedure catch to final cores faster, an initial core is assigned to each tour, without considering the capacity limitation, using an assign procedure.

The algorithm then improves the initial cores by applying k-means clustering algorithm, again without considering capacity limitation. After attaining the improved cores, the capacity limitation is taken into consideration and the algorithm adjusts the cores by assigning the customer to the tours. The main procedure of the proposed algorithm is shown in Figure 2 as follow:

In practice, we face and solve two problems; first, there is the possibility of facing infeasible solutions which will be amended by applying feasibility procedure. Second, by adding the customers to the tours after capacity limitation the cores do not tend to a constant value rather they oscillate between some points. As an explanation, suppose that nodes A and B are assigned to a tour *i*, but after applying capacity limitation the node B cannot be in the tour anymore. At the end of assigning nodes, the core will be updated and because of the change in its coordination, in the next assigning procedure node A may no longer stay in the tour while node B may enter it.

The next update may change core's coordination again and this loop will continue. For illustrate Unstable cores in capacitated clustering figure 3 is shown in below:

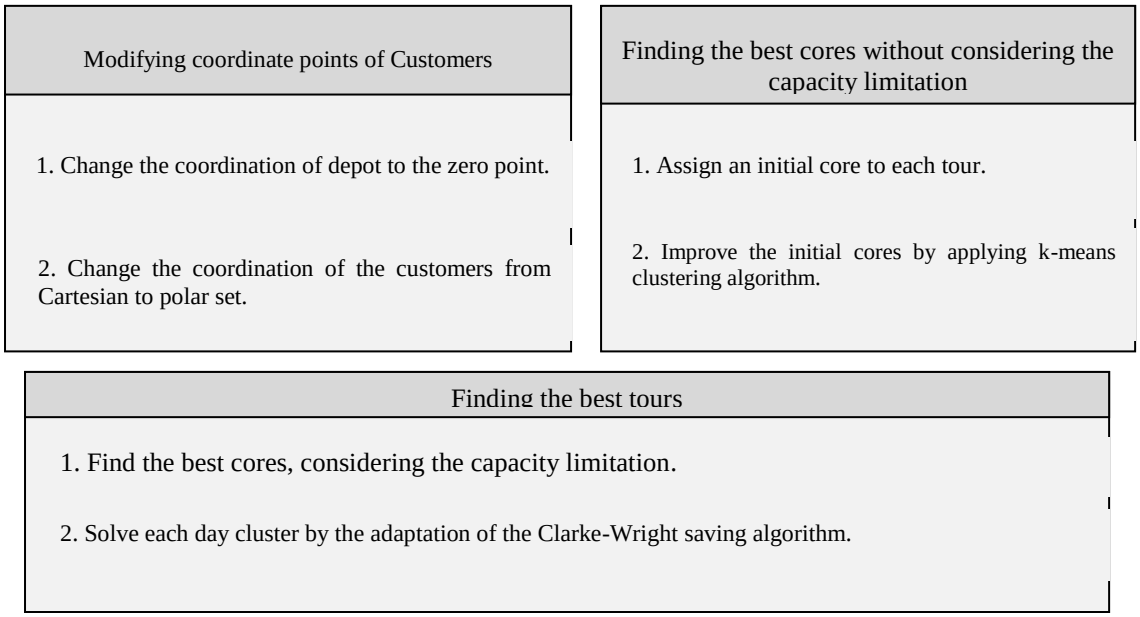


Fig. 2. The main procedure of the AD algorithm

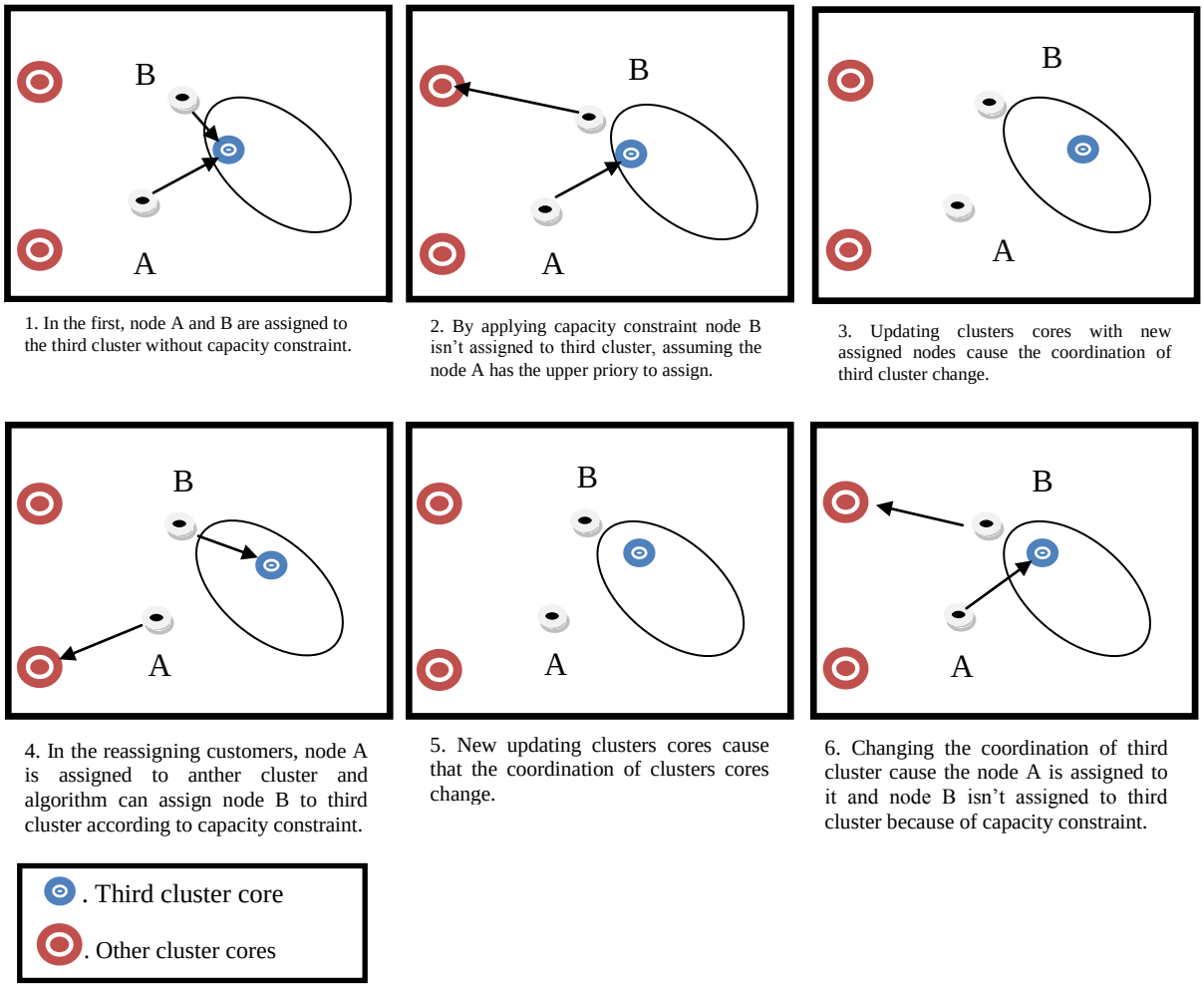


Fig. 3. Unstable cores in capacitated clustering.

In order to catch the best cores and tours the Davies-Bouldin criterion is being calculated and the cluster with the least Davies-Bouldin criterion is chosen. In the last step, the collection of each day's tours are considered as a VRP and solved by the adaptation of the Clarke-Wright savings algorithm.
Before exploring the algorithm, the two definitions and procedures used in the algorithm are explained.

5-1. Tour Core:
The tour core is determined by the average of radian value of customer's polar coordination. The initial value of the core is considered to be zero.

5-2. Davies-Bouldin Index:
The Davies-Bouldin criterion (Davies and Bouldin, 1979) is a measure for cluster desirability. This criterion is calculated by the equation 1.
In the equation 1, with defining the density of a cluster as the average of absolute node distances from the cluster's core, k, x_i and c_i are defined as follow:
k: the number of clusters.
x_i : the density of the cluster i.
c_i : the core of cluster i.
$D(c_i, c_j)$: distance between clusters cores.

$$DB = \frac{1}{k}\sum_{i=1}^{k} \max\left(\sum_{j=1\&j\neq i}^{k} \left(\frac{X_i + X_j}{D(c_i, c_j)}\right) \right) \quad (11)$$

Davies-Bouldin criterion is initiated on this concept that in clustering whatever the density of each cluster is smaller and distance of cluster's cores is greater the cluster is better.

5-3. Assign Procedure:
This procedure assigns the customers to a tour without considering capacity limitation in following steps:
1. Calculate the cost of all combinations of a customer. This cost is the sum of combination's daily costs which is the minimum distance (according to their radian value) between tour cores of the day and the customer. Figure 3 illustrates the procedure of finding the cost of allocating customer to its combinations.
2. For each customer, Choose the combination with the least cost and assign it to the tour with the least distance in related day.
3. Update the tour cores which new customer has been assigned to. Use the average value of assigned customer's radian value for each tour.

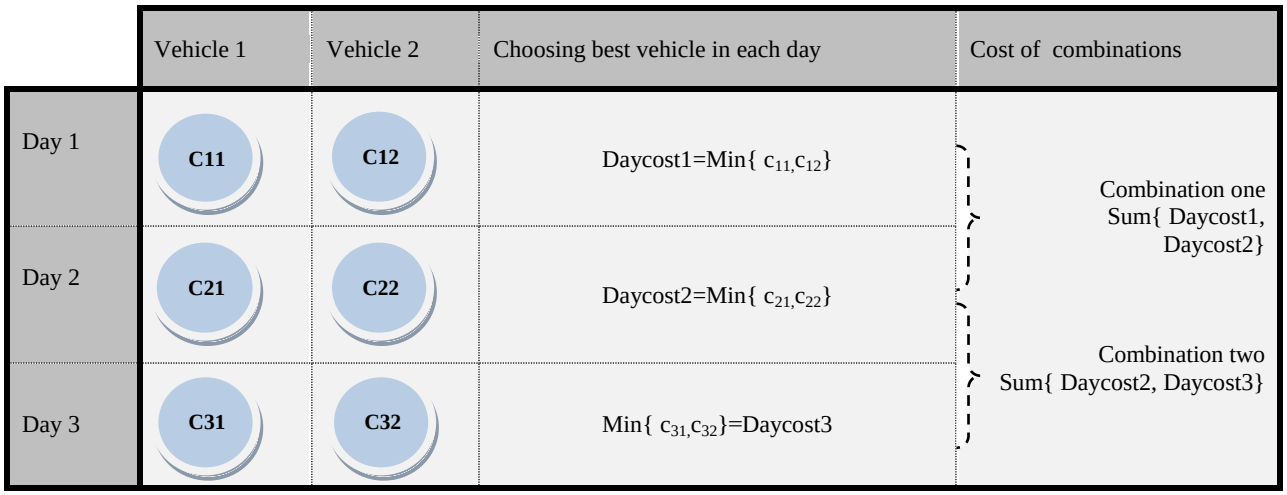


Fig. 4. The procedure of finding the cost of allocating a customer for example with two vehicles and three days and two combinations ([1, 2] and [2, 3]).

5-4. Feasibility Procedure
The Feasibility procedure is applied when there is no feasible combination for a customer. In order to solve this problem these steps are taken.
1.Consider all combinations of a given customer and define the infeasible days of them.
2. Sort combinations of a given customer non_ decreasingly according to number of infeasible days.
3. Select sorted combinations from the top, in each infeasible day of selected combination, sort the tours non-increasingly according to their remained capacity.
4. Start from the first tour. Now, check whether it is possible to remove one (some) customer(s) from the tour and assign it(them) to another tour to use the released capacity for accommodating the infeasible customer? In selecting the other tour, it is preferred that the removed customer(s) be serviced in the same combination. If this change is possible for at least one tour of each infeasible day of selected combination, do the movements and exit the procedure otherwise check the next combination.
5. Announce the problem infeasible if there is no left combination.

5-5. Algorithm Steps

In this subsection the algorithm is described in details.

1. First Step:

- Change the coordination of depot to the zero point of coordination system by subtracting its coordination from all the nodes.
- Change the coordination of the nodes from Cartesian to polar using the two equations 2 and 3. in these equations x and y are the coordinates of the node:

$$R = (x^2 + y^2) \quad (12)$$

$$\theta = \arctan(\frac{y}{x}) \quad (13)$$

It is obvious that Radian coordinate of the nodes can vary from 0 to π and from - π to 0.

2. Second Step:

In this phase the initial cores are considered for the tours and then are improved by the k-means clustering algorithm. The generated cores in this phase are used to build feasible tours regarding to capacity limitation.

- Core formation

1. Sort the nodes in a non-decreasing order according to their radian value.
2. Let the initial value of the cores be equal to zero.
3. Pick one of the possible combinations of the first customer with minimum number of days and assign it to a random tour in a random day combination.
4. Update the core of the tours which nodes have been assigned to by using average value of customer's radian value with the zero angle of central depot.
5. Apply the assigning procedure to the all other customers.

- Core improvement using k-means clustering algorithm

1. Remove all the customers from tours.
2. Appoint the created core as the initial core.
3. Sort the customers non-decreasingly according to their radian part.
4. Assign the customers to the tours using assignment algorithm.
5. Compare the new core with the initial one. If the difference between sum of the new cores and the initial ones is less than ε=0.01, go to next step, otherwise make new cores the initials and repeat the subroutines of core improvement.

3.Third Step

Up to now, the best tours have been created but the capacity limitation has not been satisfied. In this phase the customers are clustered according to the improved cores gained from the later phase. The problem in this part, as mentioned before, is that the cores do not tend to a constant value which is solved by applying Davies-Bouldin criterion.

Tour creation considering capacity limitation

1. Sort the customers non-increasingly according to their demand.
2. Pick the customers from the top and assign them using assign procedure and considering feasible combinations. A feasible combination is a combination in which the customer can be serviced in all days of combination. In other words, the capacities in all days of the combination allow the customer to be serviced.
3. Apply feasibility procedure if no feasible combination is found for a customer.
4. Save the created core and calculate the Davies-Bouldin criterion of created tours.

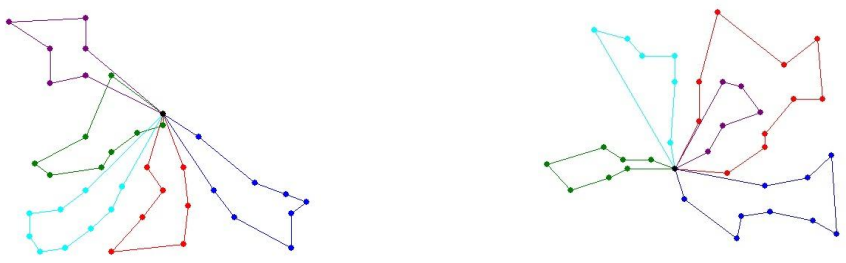
Appoint the core from former step to the initial core and repeat the sub routines 10 times. Let's mention ,testing on three test problems, results shown that increasing the number of repetition from 10 times does not affect on finding better cores, so this parameter is set to 10.

5. Select the tours with the least value of Davies-Bouldin criterion from all created tours.

4. Fourth Step

The aim of this phase is to finalize the tours of the last step. Adaptation of the Clarke-Wright savings algorithm is applied to assign the customer of each day to its tours.

The Clarke and Wright savings algorithm terminates when no two routes can feasibly be merged, i.e., no two routes can be merged without violating the route capacity constraints. Consequently, the number of routes may exceed the number of available vehicles. In that case, a route with the fewest customers is identified and the customers in this route are moved to other routes of day (minimizing the increase in costs).Figure 5 shows one day tours obtaining from solving a problem with the algorithm.



a.First day tours **b. Second day tours**

Fig. 5. Tours obtained from solving a 4th sample test with the algorithm.

6. Computational Results

The aim of this section is to compare the proposed algorithm result with the ones from literature. The computer code was written using Matlab 7 on Intel dual core, 2.33 GHz processor and 2 GB of RAM-memory. For comparing the algorithm with other algorithms in literature the 13 instances are chosen. The first 10 instances were introduced by Eilon et al. (1971) and adapted to the PVRP by Christofides and Beasley (1984), the 11th instance was proposed by Russell and Igo (1979) and finaly, the 12th and 13th by Russell and Gribbin (1991).

6-1. Compression with Heuristic Algorithms

Since the proposed algorithm is categorized as heuristic algorithms catagory, In the first, the proposed algorithm's results are compared with those obtained by the heuristic algorithms in the literature. These heuristic algorithms are Tan and Beasley (1984) (TB), Christofides and Beasley (1984) (CB), Russell and Gribbin (1991) (RG). The results shown in the table 2.

Tab. 2. Results of proposed algorithm compared to Heuristic algorithms

instances	TB	CB	RG	AGBD	ER
1	–**	547	537.3	540.98	0.007
2	1481	1443	1355.4	1368.5	0.010
3	–	547	-	569.94	0.042
4	–	844	867.8	864.02	0.024
5	2193	2187	2141.3	2207.84	0.031
6	–	938	-	**903.5**	**0.00***
7	–	839	833.6	863.52	0.036
8	2282	2151	2108.3	2241.35	0.063
9	–	875	-	**873.29**	**0.00***
10	1834	1674	1636.5	1705.04	0.042
11	878.5	847	820.3	**820.06**	**0.00**
12	–	–	1312	1345.04	0.025
13	–	–	3638.1	3684.08	0.013
Average gape					**0.019**

*. The algorithm reached to the optimal solution.

**. The algorithm could not find a feasible solution for the instance.

In this table, the first column is related to the number of instances and columns two to four are related to the answers obtained from heuristics algorithms in the literature.

Furthermore 5th column is related to the answers of the proposed algorithm. The 6th column shows the percentage error of the proposed algorithm. Unfortunately, we couldn't find any information about CPU time of heuristic algorithms in the literature, so only the answer's quality is considered for comparison.

Percentage error of the algorithm is calculated with $(BEST_{GABD} - BEST_{HEU}) / BEST_{HEU}$. In this formulation, the $BEST_{GABD}$ is the best answer obtained from proposed algorithm and $BEST_{HEU}$ is the best answer obtained from heuristic algorithms in the literature.

As it has shown in the table 2, the proposed algorithm reached to the best solution for 6th and 9th instances; also the average gap is about 1.9% that is shows, the very good quality of answers of the proposed algorithm.

6-2. Compression with Meta Heuristic Algorithms

For more detailed analysis of performance of the proposed algorithm, two most recently developed meta heuristics algorithms- Alegre et al. (2007)(ALP) and Hemmelmayr et al.(2009) - are considered. Since the desired algorithms have been implemented on different computers, direct compare of solving times is difficult. To prevent from this problem the information gathered by the benchmark of Dongarra (2009) is used.

The related information of this benchmark is available at http://www.netlib.org/benchmark/performance.ps. This paper deals with comparing computer speeds and applies flop/s (floating point instructions per second) criterion for this purpose.

Since there were no computers with characteristics of the ones in ALP, VNS and AD, in the first step the information of the most similar computers was used for ALP and AD. The computer characteristics are presented in table 4.

Tab. 4. Specification of computers performance that used for comparing the algorithms abilities

Algorithms	Real specification	Nearest specification	Mflop/s
ALP algorithm	Pentium III, 600 MHz	Pentium III 800 MHZ	2216
AD algorithm	Intel dual core , 2.33 GHz processor	Intel Core 2 Q6600 Kensfield (2 core, 2.4 GHz)	9669

Considering flop/s criterion, our algorithm times are multiplied by 4.36 to be comparable with time of ALP algorithm. For comparing the computational time of VNS the reported computational times in Hemmelmayr et al.(2009) are used. These computational times are comparable with ALP computational times and thus are comparable with modified Computational times of AD algorithm. The result is shown in table 5. In this table, the columns second to 5th are related to the ALP algorithm, and respectively are the best answers,

computational time, percentage of quality error and percentage of computational error, also other columns are related to the VNS algorithm and AD algorithm, in addition AD algorithm has two columns for computational time-columns 11th and 12th- first one is the detected computational time of the algorithm and second is corrected run time of the algorithm according to the flop/s criterion.

Tab. 5. Results of AD algorithm compared to meta-heuristic algorithms considered from literature

	ALP				VNS-10[7]				AGBD				
#	**Best.Obj**	CPU.T	ER.O	ER.T	**Best.Obj**	CPU.T	ER.O	ER.T	**Best.Obj**	CPU.T	R.CPU.T	ER.O	ER.T
1	531.02	268	0.01	20.65	524.61	98.3	0.00	33.61	540.98	2.84	12.38	0.03	0.00
2	1324.74	494	0.00	20.02	1332.01	81.6	0.01	14.14	1368.5	5.39	23.5	0.03	0.00
3	537.37	45	0.02	40.28	528.97	100.5	0.00	401.00	569.94	0.25	1.09	0.08	0.00
4	845.97	1426	0.00	90.88	847.48	67.2	0.00	17.88	864.02	3.56	15.52	0.02	0.00
5	2043.75	1280	0.00	62.81	2059.74	68	0.01	13.78	2167.8	4.6	20.06	0.06	0.00
6	840.1	1797	0.00	64.11	884.69	76	0.05	11.01	903.5	6.33	27.6	0.08	0.00
7	829.65	199	0.00	6.56	829.92	183.2	0.00	29.33	863.52	6.04	26.33	0.04	0.00
8	2052.21	3584	0.00	130.52	2058.36	142.9	0.00	21.86	2141.35	6.25	27.25	0.04	0.00
9	829.65	970	0.00	27.74	834.92	193.1	0.01	23.95	873.29	7.74	33.75	0.05	0.00
10	1621.21	9467	0.00	296.89	1629.76	170	0.01	22.32	1705	7.29	31.78	0.05	0.00
11	782.17	6492	0.00	124.45	791.18	253.7	0.01	20.37	825.06	11.87	51.75	0.05	0.00
12	1230.95	515	0.00	11.91	1258.46	354.7	0.02	37.78	1312	9.147	39.88	0.07	0.00
13	_	1491.6	_	20.65	3835.9	97.2	0.04	5.15	**3684.08**	15.8	68.89	**0.00***	0.00
	Average		**0.002**	**70.575**			**0.012**	**50.168**				**0.047**	**0.000**

In this table, the percentage of quality error calculated using (h-z*)/z* , that h and z* are the answer of the selected algorithm and best answer from other algorithms respectively, Also the percentage of computational error calculated using ((CPU(t)-BestCPU(t))/ BestCPU(t), In this equation the CPU(t) and BESTCPU(t) respectively are solving time of the selected algorithm and the best solving time from other algorithms.

As it is shown in table 5, the proposed algorithm reached to the best solution for 13th instance in addition the average percentage of quality error is about 4.7% ,that is the very good result for a heuristic algorithm compared with meta heuristic algorithms. The average computational run time of algorithms 7057.5%, 5016.8% and 0% for ALP, VNS and proposed algorithm respectively. As it is clear computational times of the proposed algorithm are the best for all instances, and our algorithm is supreme in computational time. The average CPU time for ALP VNS and AD algorithm are 2156.46,145.108, and 29.214 respectively. evidences show that proposed algorithm obtained to the answers with good quality in short period of time.

7. Conclusions

The aim with this paper was to develop a swift heuristic algorithm for the periodic vehicle routing problem. As it is apparent from the results, our heuristic algorithm reached to the better solution for two instances compared with answers of heuristic algorithms from the literature, Also this compression shown that the algorithm has the ability to find very good answers for all instances. In compression with Meta heuristic algorithms the results showed that the proposed algorithm has good quality, with reaching to the better solution in one instance.

In addition the proposed algorithm was supreme in computational time. Especial attributes make it appropriate to be used for large scale problems as an upper bound for PVRP. Applications of this algorithm is in situations in which finding an acceptable answer for periodic vehicle routing problem in the very limited time is required, the location periodic vehicle routing problem and robust periodic vehicle routing problem are some applications area. Furthermore, the answers can be used as initial solutions for improvement algorithms like metaheuristics which can be considered in next studies.

References

[1] Alegre. J., Laguna. M., Pacheco, J., *Optimizing the Periodic Pick-up of Raw Materials for a Manufacturer of Auto Parts*. European Journal of Operational Research. 137, 2, 2007, 233–247.

[2] Angelelli. E., Speranza. M.G., *The Periodic Vehicle Routing Problem with Intermediate Facilities.* European Journal of Operational Research 137, 2, 2002, 233–247.

[3] Augerat. P., Belenguer. J.M., Benavent. E., Corberán. A., Naddef. D., Rinaldi. G. *Computational Results with a Branch and Cut Code for the Capacitated Vehicle Routing Problem*. Research Report, 949-M, Universite Joseph Fourier, Grenoble, France.

[4] Beltrami, E.J., & Bodin, L.D., *Networks and Vehicle Routing for Municipal Waste Collection*. Networks 4, 1974, 932.

[5] Christofides. N., & Beasley, J.E., *The Period Routing Problem.* Networks, 14, 1984, 237±256.

[6] Clarke. G., Wright. J.W., *Scheduling of Vehicles From a Central Depot to a Number of Delivery Points*. Operations Research, 12, 1964, 568-581.

[7] Davies .D.L., Bouldin. D.W., *A Cluster Separation Measure. IEEE Trans*. Pattern Anal. Machine Intell. 1 (4). 1979, 224-227.

[8] Djiconstantinuo. Ha, Baldacci. E., *A Multi-Depot Period Vehicle Routing Problem Arising in the Utilities Sector*. Journal of Operations Research Society, 49 ,12, 1998, 1239–1248.

[9] Dongarra J., *Performance of Various Computers Using Standard Linear Equations Software,* Computer Science and Mathematics Division Oak Ridge National Laboratory Oak Ridge, TN 37831, 2009.

[10] Drummond. Lúcia, M.A., Ochi. Luiz, S., Vianna. Dalessandro, S., *An Asynchronous Parallel Metaheuristic for the Period Vehicle Routing Problem*, Future Generation Computer Systems, 17, 2001, 379–386.

[11] Eilon. S., Gandy. W., Christofides. N., *Mathematical Modeling and Practical Analysis*. Distribution Management Griffin, London, 1971.

[12] Francis, P.M., Smilowitz, K.R., Tzur, M., *Flexibility and Complexity in Periodic Distribution Problems*. Naval Research Logistics, 2007.

[13] Francis, P., Smilowitz, K., Tzur, M., *The Period Vehicle Routing Problem and its Extensions*, in: The Vehicle Routing Problem: Latest Advances and New Challenges, Vol. 43, Springer, 2008 (Chapter).

[14] Fisher. M.L., Jaikumar. R., *A Generalized Assignment Heuristic for Vehicle Routing.* Networks,11, 1981, 109±124.

[15] Gaudioso. M., Paletta. G.A., *Heuristic for the Periodic Vehicle Routing Problem*. Transportation Science, 26, 2, 1992, 86±92.

[16] Gillett, B., Johnson, J., *Multi-Terminal Vehicle-Dispatch Algorithm*. Omega, 4, 1976, 711–718.

[17] Hadjiconstantinou, E., Baldacci, R., *A Multi-Depot Period Vehicle Routing Problem Arising in the Utilities Sector.* The Journal of the Operational Research Society 49 (12), 1998, 1239–1248.

[18] Han. J., Kamber. M., *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers, 2001.

[19] Hays. W.L., *Statistics. New York*: Holt, Rinehart, and Winston, 1981.

[20] Myers, Jerome L., Arnold, D., Well., *Research Design and Statistical Analysis* (second edition ed.). Lawrence Erlbaum. 2003, pp. 508

[21] Osman. H., Ibrahim., *.Met Heuristics: Models, Design and Analysis*, Proceedings of the Fifth Asia Pacific Industrial Engineering and Management Systems Conference, 2004.

[22] Pacheco J., Alvarez, A., García, I., Angel-Bello F., *Optimizing Vehicle Routes in a Bakery Company Allowing Flexibility in Delivery Dates*, Journal of the Operational Research Society, 2011.

[23] Russell. R.A., Gribbin. D., *A Multiphase Approach to the Period Routing Problem*. Networks, 21, 1991, 747±765.

[24] Russell. R.A., Igo, W., *An Assignment Routing Problem*. Networks, 9, 1979, 1±17.

[25] Tan. C.C.R., Beasley. J.E.,. *A Heuristic Algorithm for the Period Vehicle Routing Problem*. OMEGA, International Journal of Management Science, 12, 5, 1984, 497±504.

[26] Vidal, T., Crainic, T., Gendreau, M., Lahrichi, N., Rei, W., *A Hybrid Genetic Algorithm for Multi-Depot and Periodic Vehicle Routing Problems*, Operations Research 60 (3), 2012, 611–624.