A particle swarm optimization method for periodic vehicle routing problem with pickup and delivery in transportation

Masoud Rabbani*[1], Neda Manavizadeh[2], Ali Shamekhi[3]

[1]Professor, Department of Industrial Engineering, College of Engineering, University of Tehran, mrabani@ut.ac.ir
[2]Assistant Professor, Faculty of Engineering & Technology, Alzahra University
[3]MSc Student, Department of Industrial Engineering, College of Engineering, University of Tehran

Abstract:
In this article, multiple-product PVRP with pickup and delivery that is used widely in goods distribution or other service companies, especially by railways, was introduced. A mathematical formulation was provided for this problem. Each product had a set of vehicles which could carry the product and pickup and delivery could simultaneously occur. To solve the problem, two meta-heuristic methods, both based on particle swarm optimization, were provided and ran for small and large class problems and their efficiency were demonstrated. Also, efficiency of binary PSO to general PSO was tested and BPSO was shown to outperform the general method. This approach can be used in railway transportation.

Keywords: *periodic vehicle routing, particle swarm optimization, binary particle swarm optimization, railway*

---

* Corresponding Author

1. Introduction

VRPs are one of the important optimization problems both in theory and practice. Many constraints have been added to general problem and extended it to new branches. In this paper, the focus was on periodic vehicle routing problem as a combination of two classic problems: vehicle routing and assignment problem. In PVRP, each customer should be visited with a predetermined service frequency on t-day period of planning horizon. These services occur on specific combinations. For example, if the service frequency is 2 and combinations are {1,3}, {2,5} and {4,6}, then each customer should be assigned to one of these combinations. The problem consists of simultaneously selecting a visit combination for each customer and establishing vehicle routes for each day of the planning horizon according to the VRP rules [8].

PVRP with pickup and delivery includes material pickup from suppliers to the main factory and delivery of products from central factory to customers. The highest application of this problem is in distributing products, e.g. in fuel distribution or logistic department of a manufacturing company. Consider a company that produces multiple kinds of products. Each product has known customers with known demand in each period. This company also has some needs for raw material. The fleet of company should both visit customers and providers such that the planned frequency is satisfied. For example, a company needs a batch of 100 units of raw material. Also, customer C demands 150 units of product A three times a week. Frequency, demand and supply of each product may be different. The company has multiple kinds of trains which are suitable for some products and cannot carry some others; for example, fluidic goods cannot be carried in one of the vehicles. It is supposed that inventory level is so as to satisfy demand of customers. There is no constraint in the case that pickup demands should be satisfied first or should be at the end of the routes. Even it can happen simultaneously. The objectives of the problem are to minimize total cost of tours over planning horizon. In this study, a general framework was provided that can be extended to be used in companies' logistic departments.

In Section 2, previous studies in the field of PVRP are described. In Section 3, a mathematical formulation of the problem is provided. Section 4 is about PSO in general and Section 5 describes the proposed algorithm. In Section 6, the result of numerical studies is provided and, finally, a summary of the article is reported in Section 7.

2. Literature review

Despite the application of PVRP, it has received scarce attention in the literature. Beltrami and Bodin [1] were one of the first authors who worked on PVRP. Russell and Igo [12] developed a heuristic in periodic assignment routing problem. Also, Chao et al. [14] developed a two-phase heuristic. Christofides and Beasley [2] proposed a mathematical model with allowable day combinations for the customers who needed multiple visits and solved the problem using a heuristic.

SDVRP (vehicle routing with site dependencies) introduced by Nag et al. [9]. Chao et al. [10] and Cordeau et al. [11] solves this problem as an special case of PVRP. Hadjiconstantinou and Baldacci [7] extended the problem to multiple-depot PVRP which was applicable to the utility sector.

Francis and Smilowitz [22] presented a continuous approximation model for the period vehicle routing problem with service choice (PVRP-SC). PVRP-SC is a variant of the period vehicle routing problem, in which visit frequency to nodes is a decision of the model.

PVRP has a great application in real world. Baptista e al. [18] provided an extension of the heuristic algorithm proposed by Christofides and Beasley [2] to solve a real case: collection of recycling paper containers in the City Council of Almada. Shih et al. [19] also solved infectious waste collection problem as PVRP and extended the algorithm that was developed before [9]. Their algorithm worked with a computer program. Alegre et al. [17] solved periodic pickup of raw materials for a manufacturer of auto parts.

Paletta [16] provided a heuristic for PTSP and Doerner et al. [3] proposed a method based on variable neighborhood search (VNS), which could be also used in PTSP. Vanderbeck et al. [20] paid attention to tactical planning model which restricted its attention to scheduling and assigning visits to vehicles while leaving sequencing decisions for an underlying operational model. They proposed a model based on a truncated column generation to optimize regional compactness of the routes and balance the workload.

Some metaheuristics have been developed to solve the PVRP. Golden et al. [4] proposed a method using the concept of "record-to-record". Cordeau et al. [8] developed a general tabu search method that could be used in three types of routing problem: multi depot VRP, periodic travelers sales person and periodic vehicle routing problem. Later, they proposed its adapted version, which could handle several vehicle types and site dependency. Drummond et al. [15] proposed a parallel algorithm which was based on the concepts used in parallel genetic algorithms and local search heuristics.

Angelelli et al. [6] proposed a similar tabu search for a new extension of PVRP and considered that replenishment was allowed at intermediate facilities. It means that, if load of a capacity was loaded to maximum capacity, the vehicle could be unloaded at an intermediate facility, like a warehouse. They considered that each vehicle returned to depot only when its work shift was over.

Rocha [5] proposed an efficient hybrid genetic algorithm (HGA) to the problem that outperformed other heuristics in some cases. Gaudioso and Paletta [13] developed a heuristic to minimize the number of vehicles used to satisfy customers' need and route the vehicles over. Also, they paid attention to balancing the work assigned to vehicles on each day.

Prins et al. [21] provided a memetic algorithm for periodic capacitated arc routing problem (CARP). Their solution was able to simultaneously change tactical (planning) decisions, such as treatment days of each arc, and operational (scheduling) decisions, such as the trips performed for each day. However, it could be extended with prohibited turns, mixed graphs and possibility of tackling period or spacing-dependent demand and service cost. Considering railway in this article as a mode of transportation was a new contribution to VRP problems.

3. Formulation

In this section, the problem is formulated as a mathematical program to minimize total traveling cost. The problem is defined as a multi-graph G=(V,A) where $V=\{v_0,v_1,\ldots,v_n\}$ an v_i is set of vertex at time t and $A=\{(v_i,v_j)^{p,k,t} \mid k \in k(p)\}$ is set of arcs. Index k refers to vehicle and t refers to day. $k(p)$ is set of machines which can carry the p^{th} product.

In a T-days planning horizon, F_{ip} is equal to the number of services that each customer needs. This means that a combination should be assigned to a customer which at least covers F_{ip} days. F_{ip} varies between 1 and T which means a customer can be visited at most once a day. F_{ip} is determined by the need of companies or customers. For example, a company needs a batch of 100 units of product A twice in a planning horizon; so, F_{ip} is 2.

Visit combinations are defined by planners. Sometimes, they consist of every day service in the entire planning horizon or even once. List of indices, parameters and variables is given below:

The objective function is to minimize the fixed cost that

i,j: indices of customers and suppliers

k: index of transportation mode, like railway

p: index of product

t: index of time

r: index of visit combination

y_{ipr} : equals 1 if and only if the visit combination r is assigned to customer/supplier i in order to satisfy demand/supply of product p.

f_{ipkt} : equals 1 if and only if transportation mode k is assigned to customer/supplier i to satisfy the demand/supply of product p at time t.

x_{ijkpt} : equals 1 if transportation mode k travels path (i j).

u_{ipkt} : load of product p on transportation mode k when visiting customer i.

q_{ip} : demand of customer I for product p.

s_{ip} : amount of product p which should be satisfied by i^{th} should satisfy.

Q_k : capacity of train K

a_{rt} : equals 1 if day t belongs to visit combination r; otherwise, 0.

D_k : maximum distance that *kth* train can travel

c_k : fixed cost of the traveling *kth* train per unit distance

c_k' : cost of the traveling *kth* train per unit distance per unit of weight

d_{ij} : distance between customers i and j

$$\text{Min } Z= \sum_i \sum_j \sum_k \sum_t c_k x_{ijkt} + \sum_i \sum_j \sum_k \sum_p \sum_t c_k' . u_{jpkt} . x_{ijkt} . d_{ij} \qquad (0)$$

$$\sum_r y_{ipr} =1 \qquad \forall i,p \qquad (1)$$

$$\sum_t \sum_r a_{rt} .y_{ipr} \geq F_{i,p} \qquad \forall i,p \qquad (2)$$

$$\sum_k \in_{k(p)} f_{ipkt} =\sum_r a_{rt} .y_{ipr} \qquad \forall i,p,t \qquad (3)$$

$$f_{ipkt} \leq \sum_j x_{ijkt} \qquad \forall i,p,k,t \qquad (4)$$

$$\sum i\ x_{ijkt} =\sum_i x_{jikt} \qquad \forall i,k,t \qquad (5)$$

$$\sum_i x_{ijkt} \leq 1 \qquad \forall j,k,t \qquad (6)$$

$$\sum_i \sum_j x_{ijkt} . d_{ij} \leq D_k \qquad \forall k,t \qquad (7)$$

$$u_{ipkt} - (q_{jp}-s_{jp}) .f_{jpkt} - u_{jpkt} \leq M (1-x_{ijkt}) \qquad \forall k,i,j,p,t \qquad (8)$$

$$u_{ipkt} - (q_{jp}-s_{jp}) .f_{jpkt} - u_{jpkt} \geq - M (1-x_{ijkt}) \qquad \forall k,j,p,t \qquad (9)$$

$$u_{1pkt} = \sum_i q_{ip} . f_{ipkt} \qquad \forall p,k,t \qquad (10)$$

$$\sum_p u_{ipkt} \leq Q_k \qquad \forall k,i,t \qquad (11)$$

$$ujpkt \leq M.\sum i\ xijkt \qquad \forall p,k,t,j \qquad (12)$$

$$x_{ijkt} , f_{ipkt} , y_{ipr} = \{0,1\} \quad u_{jpkt} \geq 0$$

each train consumes for traveling the distance and variable cost which depends on the load of train. Constraint (1) guarantees that demand/supply of each customer/supplier is assigned to a combination of days. Constraint (2) ensures that the tour which is assigned to the customer is at least equal to its planned frequency of visit. Constraints (3) and (4) stipulate that each node's demand/supply on the assigned combination should be satisfied by a train. Constraint (5) guarantees that each train entering a customer should exit from it. Constraint (6) confines the train to move on just one arc at a time. Constraint (7) is for maximum traveling of each train. Constraints (8) and (9) determine load of the trains when visiting customers and Constraint (10) determines load of each train when starting from depot. Constraint (11) is upper limit of the load of train. Constraint (12) means that load of each train can be greater than zero if and only if the train visits it.

4. Particle Swarm Optimization in General

Particle swarm optimization which was introduced by Kennedy and Eberhart [24] is inspired by motion of bird swarms. Individuals in a PSO have a position and velocity and are symbolized particles. The birds move to reach a place which has a greater amount of food. Each particle in this swarm has two kinds of intelligence: self intelligence and social intelligence, which is sharing information; so, individuals can use previous experience of all other particles. Social intelligence helps the birds; so, they move toward the optimal solution passed by the swarm; on the other hand, self intelligence helps them to search the neighborhood of the best place that have been seen. The position with the minimum fitness value is the entire swarm's global best (gbest) position, towards which other particles move. In addition, each particle's best position which has been visited is its personal best (pbest). Velocity of each bird corresponds to gbest and pbest and the algorithm is developed on the basis of these facts.

Consider a swarm with p particle, each of which is a feasible solution of the problem. For each particle of i, represents position of particle i, x^i Position of the particle would be updated at each iteration by the following formula:

$$x_k^{i+1} = x_k^i + v_{k+10}^i \qquad (1)$$

where k is index of iterations. Velocity is updated by the following formula:

$$v_{k+1}^i = \omega_k \cdot v_k^i + c_1 r_1 (p_k^i - x_k^i) + c_2 r_2 (p_k^g - x_k^i). \qquad (2)$$

K indicates the number of iteration, p_k^i is the best ever position of particle I at iteration k (cognitive contribution) and p_k^g is the global best position of swarm.r_1 and r_2 are random numbers which are uniformly distributed between 0 and 1.c_1 is selfishness coefficient and c_2 is sociality coefficient. ω_k is inertia of the particle which is impact of velocity of each particle at iteration k on velocity of iteration k+1. ω_k can be set constant or inferred from an equation; so, the inertia can be adopted at each iteration. A maximum and minimum level can be set for velocity, which helps the particle to move smoothly. It means that it helps the particles escape from local optimum and also does not allow them to move rapidly and jump over the local optimum. Also, similar parameters can be set fo inertia.

The inertia is usually set at the beginning maximum; so, searching larger space gradually decreases it to more searching at the neighborhood of particles.c_1 and c_2 are mostly set by fine tuning; but, it has been mentioned in the literature that it is better to set them $c_1 = c_2 = 2$ therefore, because r_1 and r_2 are uniformly distributed between 0 and 1, the particle would be equally affected by the global best and local best. The algorithm's pseudo-code is shown in Figure 1.

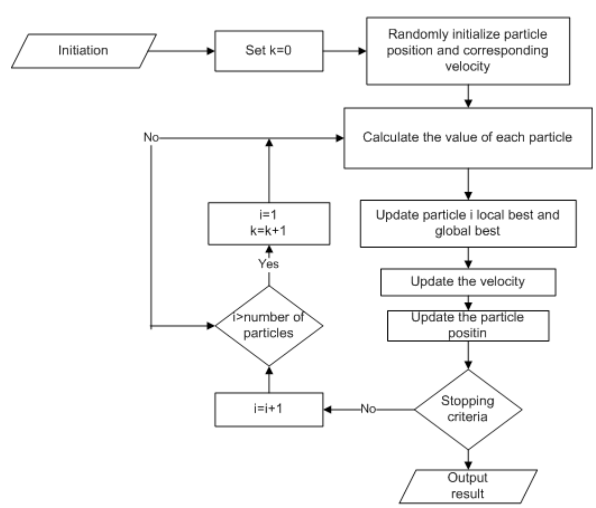


Figure 1. Particle Swarm Optimization Flow Chart

5. The proposed algorithm

Two algorithms were proposed based on particle swarm optimization. These two algorithms had two representations; so, their cost was different.

5.1. Binary PSO

General PSO is an efficient method for continuous variables. Although floor and similar operators can be used to develop the algorithm for discrete forms, when the variable is binary, moving space is so much narrow that may force the particle to be fixed on its position. Thus, a new improved algorithm based on PSO intelligence was proposed. In the new method, stage of updating velocity remained the same. After updating velocity, $s(v_i^{k+1})$ could be calculated such that:

$$s(v_i^{k+1})=1/(1+e^{-v_i^{k+1}}) \quad (3)$$

So, if ,$r < s(v_i^{k+1})$, then $x_k^{i+1} = 1$; else, $x_k^{i+1} = 0$; where r is a random number which is uniformly distributed between 0 and 1. $s(v_i^{k+1}) = 35\%$ means that there is 35% chance that particle I is assigned. This method was proposed by Kennedy and Eberhart [23] and used in some industrial problems with acceptable performance. They claimed that better values are likely choosing 1 with higher chance. The problem is the new particle position may not be feasible because the algorithm does not search just I feasible space. And, there was no guarantee that velocity was so that the constraints were satisfied.

5.2. The 1st proposed algorithm

In the proposed flowchart, only y_{ipr} and f_{ipkt} positions were moved using (EQ.3) to determine value of variables. Value of x_{ijkt} routes that should be traveled by train was constructed based on these two variables. The pseudo-code is here:

1. Set k=0.
1.1. Set NOP=number of particles.
1.2. Generate NOP feasible solution randomly.
1.3. Calculate fitness function for each particle.
1.4. Set values= pbest for each particle.
1.5. Set global best equal to minimum value (gbest).
2. Set k=k+1.
3. Calculate velocity by equation (EQ.2).
4. If r < $s(v_i^{k+1})$, set x_k^{i+1}=1; else, x_k^{i+1}= 0.
5. Check feasibility.
5.1. If particle is feasible, go to the next step.
5.2. If more than one train/combination is assigned to a customer's demand/supply, randomly omit the extra. Or, if it is not assigned to a train/combination, randomly assign it to one.
6. Construct all feasible modes of paths which the trains should travel. Select the route with minimum cost.
6.1. If load constraints are not satisfied, generate randomly; else, continue.
7. Calculate fitness function.
8. Calculate local best and global best.
9. Loop until stopping criteria.

5.4 The 2nd proposed algorithm

The second method is based on integer PSO. In this algorithm, the variables were represented by the indices. For example, if f_{2331}=1 (the demand/supply of third commodity of customer 2 at period 1 was assigned to train number 3, the particles would be i, p, k and t. So, if velocity of k and t for this demand/supply was -1.1 and -0.23, f_{2331} would become zero; instead, f_{2331} would be 1.

In this new method, first f_{ipkt} was generated in a manner that Constraint 2 was satisfied. So, routes were produced based on .f_{ipkt} Thus, first, the customers assigned to each train in each period were identified. Then, the tour was generated for it so that load constraints were satisfied. Then, f_{ipkt} indices were moved by velocity which was calculated by velocity equation. So, if r y_{ipr} and k f_{ipkt} were moved, the particles' position would be changed in a manner that most of the constraints' feasibility was conserved (note that r determines when the demand/supply of product p should be delivered or taken from the customer). The pseudo-code is below:

1. Set k=0.
1.1. Set NOP=number of particles.
1.2. Generate NOP feasible solution randomly.
1.3. Calculate fitness function for each particle.
1.4. Set values= pbest for each particle.
1.5. Set global best equal to minimum value (gbest).
2. Set k=k+1.
3. Calculate velocity of r and k by equation (EQ.2).
4. Calculate x,y,f and u.

If load constraints are not satisfied, examine another combination of customer visiting. If appro

1. priate combination of visiting could not be found, randomly generate f_{ipkt}.
2. Calculate fitness function for all feasible tours.
3. Calculate local best and global best.
4. Calculate velocity for r and k and find a new position.
5. Loop until stopping criteria

1. Also, the equations noted in general PSO were used; but, complementary rules were used to improve efficiency of this method. These side equations were for calculating velocity and inertia.
2. In both methods, initial feasible solutions were generated using the method shown above:

1. Randomly generate an integer in [1 r_{max}] $\forall i, \forall p$. (For example, for i =1,*and* p=1,3; so, demand/supply of the first customer for product #1 is satisfied on 3rd combination. Next, for i =1,*and* p=2, 2; so, the 1st customer's demand for product #2 is satisfied on 2nd combination.)
2. Determine t for each I and p. (a_{rt} can be illustrated as a matrix which determines planning of each day on each combinations. So, when # of the combination which is assigned to satisfy a demand/supply is determined, time would be easily calculated.)
3. Determine # of train for each *i,p,t.*
4. Determine the customers which are assigned to each train on each t.
5. Construct all feasible tours for each train and select the best of them.

6. Numerical results

As described above, two meta-heuristics were used to solve PVRP with pickup and delivery. Both methods were based on particle swarm optimization.

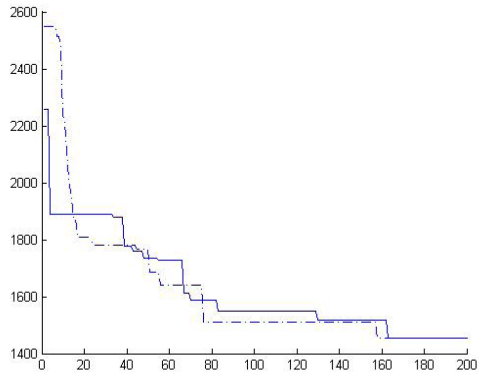
To show validation of the model and illustrate efficiency of the proposed meta-heuristics, they were tested for 5 small size problems. Exact solutions were solved by lingo 8.0 and metaheuristics #1 and #2 were coded by Matlab 7.7.0. Results are shown in Table 1. The number of variables and number of customers are demonstrated for each problem. Computational complexity was so high which could not be on a reasonable time with exact methods. Lingo runtime was also recorded to prove this; even the problems with greater dimension could not escape from feasibility to local optima. Note that the best known in the table was global optimum in lines 4 and 5. The software could not reach global optimum; so, the solver was interrupted on local optimum. The proposed methods could reach a better solution; i.e. they could solve the problem on a reasonable time, even compared to the exact solution.

NOP (number of particles) for all of them was set at 50 and the loop was done 200 times. PSO parameters were set by fine tuning. As can be observed, all of them reached a reasonable solution and it was not so strange. 200 times run with 50 particles were likely to search a large space in the feasible solutions because of small feasible solution corresponding to number of variables and constraints. To compare metaheuristics 1 and 2, their path to global optima for the 5th problem is shown in Figure 2. Initial solutions were different because the codes of producing initial solutions were separately run. It showed that, unlike greater value of initial solution, the second method could find a better solution at the same iteration. Because the second method was outlined in a manner that could only search the feasible space, it had enough time to decide on the direction to move on; so, it reached a better solution at the same time.

Table 1. Numerical Results for small size problems

#	Number of customers	Number of trains	Number of products	Number of combinations	Number of constraints	Number of variables	Run time(h)	Best known	M1	M2
1	3	4	4	2	800	1933	0.03	104	104	104
2	7	4	4	2	7838	2112	0.36	298	298	298
3	12	4	4	2	20977	4472	1.83	462	462	462
4	15	5	5	2	49556	8480	4.98	915	906	906
5	20	5	5	2	85981	13230	10.4	1516	1456	1456

*. M: Metaheuristic



***Metaheuristic 1:__ *Metaheuristic 2: _._**

Figure 2. results 5[t] small scale problem

Large size problems were tested and the results are given in Table 2. Because of large feasible space and number of constraints, global optima or even a feasible solution could not be found via software. But to compare the solutions, the problems were run for some large scale problems and the programs were run with NOP=100 and 200 iterations. Because the results were close to each other, it was probably that they reached close to global optimum. The results are shown in Table 2.

Although runtime for both methods depended to dimension of the problem, run time relatively increased as the dimension increased; but, it was less than 2 min in all the cases.

As said above, the second metaheuristic was based on the method proposed by Kennedy and Eberhart [23]. This method was proved in their study and other similar works, in which it had even better efficiency. Also, the hypothesis was tested and the results were similar to those obtained by others.

The general PSO that was used had a simple structure and its steps were all like general PSO. To map velocity (continuous variable) to binary space, a simple rule was set: if new positions were bigger than 0.5, then, assign 1 to them; else, 0.

They were also tested and compared for some small and large classes of problems, the results of which re recorded in Table 3.

All the parameters were set equal and initial solution was the same for both methods. Table 3 shows behavior of two methods. As is shown, the proposed metaheuristic had greater velocity in reaching global optimum. Because general PSO provided a narrow space for binary variables, it could not escape from local optimum and was fixed in its place.

Table 2. large scale problems

#	Number of customers	Number of trains	Number of combinations	Number of products	M1	M2
1	50	8	2	5	5756	5802
2	50	8	2	8	6134	6134
3	80	10	2	5	8360	8360
4	80	10	2	8	9634	9687
5	90	10	2	5	9838	9857
6	90	10	2	8	10284	10322
7	100	10	2	5	10360	10402

Table 3. compare general PSO and BPSO

#	Number of customers	Number of trains	Number of combinations	Number of products	General PSC	BPSO
1	7	8	2	5	5756	5802
2	12	8	2	8	6134	6134
3	20	10	2	5	9237	8360
4	80	10	2	8	9734	9687
5	90	10	2	5	10428	9857
6	100	10	2	8	11352	10322

7. Conclusions

This paper presented a new problem in PVRP considering railway transportation. The new problem consisted of multi-product version of general PVRP with pickup and delivery. At first, a mathematical representation of method was provided and the problem was solved in small scale by exact method. Because of complexity of the problem, optimum could not be obtained in large scale problems.

To solve the problem, two methods were used, both of which were based on particle swarm optimization; one of them was inspired by BPSO and the second one had a new representation of variables. They were run for small and large classes of problems. Both methods in small– medium sizes were very efficient and reached global optimum. As an exact solution could not be obtained for problems, both methods were run and it was demonstrated that they converged to a point which was probably near global optima. At the end, general PSO was used and compared with BPSO and it was shown that BPSO outperformed general method. Further research should improve efficiency of the proposed PSO in PVRP family and compared it with other methods like ant colony optimization.

References

- Beltrami, E., Bodin, L. Networks and vehicle routing for municipal waste collection, Networks 1974; 4:65–94.
-
- Christofides, N., Beasley, J.E. The period routing problem, Networks 1984; 14: 237–256.
-
- Vera C. Hemmelmayr, Karl F. Doerner, Richard F. Hartl. A variable neighborhood search heuristic for periodic routing problems, European Journal of Operational Research 2007; 791–802
-
- B.L. Golden, I.M. Chao, E. Wasil. An improved heuristic for the period vehicle routing problem, Networks 1995; 25–44.
-
- M.L. Rocha, A hybrid genetic algorithm for the periodic vehicle routing problem, M.Sc. Thesis, Department of Computer Science, Federal Fluminense University. In Portuguese.
-
- Angelleli, E., Speranza, M.G. The periodic vehicle routing problem with intermediate facilities, European Journal of Operational Research 2002; 137: 233–247
-
- Hadjiconstantinou, E., Baldacci, R. A multi-depot period vehicle routing problem arising in the utilities sector, The Journal of the Operations Research Society 1998; 49(12):1239–1248.
-
- Cordeau J-F, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems, Networks 1997;30: 105–19.
- Nag, B., Golden, B., Assad, A. Vehicle routing with site dependencies. Golden, B., Assad A, editors. Vehicle routing: methods and studies. Amsterdam: Elsevier 1988; 149–59.
-
- Chao I-M, Golden B, Wasil E. A computational study of a new heuristic for the site-dependent vehicle routing problem, INFOR 1999;37(3): 319–36.
-
- Cordeau J-F, Laporte G. A tabu search algorithm for the site dependent vehicle routing problem with time windows, INFOR 2001; 39:292–8.
-
- Russell, R.A., Igo, W. An assignment routing problem, Networks 1979; 9:1–17.
-
- Gaudioso, M., Paletta, G. A heuristic for the periodic vehicle routing problem, Transportation Science 1992; 26 (2): 86– 92.
-
- Chao, I.M., Golden, B.L., Wasil, E.A. An improved heuristic for the period vehicle routing problem, Networks 1995; 26: 25–44.
-
- Drummond, L.M.A., Ochi, L.S., Vianna, D.S. An asynchronous parallel metaheuristic for the period vehicle prouting problem, Future Generation Computer Systems 2001; 17: 379–386.
-
- Paletta, G. The period traveling salesman prob-

lem: a new heuristic algorithms, Computers & Operations Research 2002; 29:1343–1352.

- 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 2007; 179, 736–746.

- Susana Baptista, Rui Carvalho Oliveira, Eduardo Zuquete, A period vehicle routing case study, European Journal of Operational Research 2002; 139:220–229.

- Shih, L., Chang, H. A routing and scheduling system for infectious waste collection, Environmental Modeling and Assessment 2001; 6:261–269.

- Vanderbeck, M. Column generation based heuristic for tactical planning in multi-period vehicle routing, European Journal of Operational Research 2007; 183:1028–1041.

- Lacomme, P., Prins, C., Ramdane-Cherif, W. Evolutionary algorithms for periodic arc routing problems, European Journal of Operational Research 2005; 165:535–553.

- Francis, P., Smilowitz, K. Modeling techniques for periodic vehicle routing problems, Transportation Research 2006; 40:872–884

- Kennedy, J., Eberhart, R. A discrete binary version of the particle swarm algorithm, Proceedings of the world multi-conference on systemics, cybernetics and informatics 1997; 4104–9.

- Kennedy, J., Eberhart, R. Particle swarm optimization. Proceedings of the 1995 IEEE International Conference on Neural Networks 1995;4:1942-1948