Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 13 (16): 3239-3244, 2013
ISSN 1812-5654 / DOI: 10.3923/jas.2013.3239.3244

A Hybrid Genetic Algorithm with Perturbation for the Multi-depot Capacitated Arc Routing Problem

[1]Hongtao Hu, [2]Tiantang Liu, [1]Ning Zhao, [1]Yiting Zhou and [3]Dequan Min
[1]Logistics Engineering College, Shanghai Maritime University, Shanghai, China
[2]School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]Transportation Management College, Dalian Maritime University, Dalian, China

Abstract: The Capacitated Arc Routing Problem (CARP) is a difficult vehicle routing problem, where given an undirected graph, the objective is to minimize the total cost of all vehicle tours that serve all required edges under vehicle capacity constraints. In this study, a Hybrid Genetic Algorithm with Perturbation (HGAP) is proposed to solve the multi-depot CARP (MDCARP) which generalizes the CARP by extending the single-depot to the multi-depot. The proposed HGAP incorporates a Genetic Algorithm (GA), a local search, a new replacement method and a perturbation mechanism. The proposed HGAP is evaluated on the MDCARP benchmark instances and computational results show that the HGAP is very competitive.

Key words: Multi-depot capacitated arc routing problem, hybrid genetic algorithm, local search, perturbation

INTRODUCTION

Vehicle routing problems are widespread in logistics and distribution. To be specific, there are two classes of vehicle routing problems, one is node routing problems, such as Capacitated Vehicle Routing Problem (CVRP) and the other is arc routing problems, such as the Capacitated Arc Routing Problem (CARP).

The CARP introduced by Golden and Wong (1981) is an NP-hard problem; therefore, heuristics and meta-heuristics have been the mainstream approach to solve the CARP. For example, augment-merge (Golden and Wong, 1981), path-scanning (Golden *et al*., 1983), parallel-insert (Chapleau *et al*., 1984), construct-strike (Pearn, 1989), augment-insert are simple constructive heuristics; and route-first, cluster-second (Ulusoy, 1985) and cluster-first, route-second (Benavent *et al*., 1990) are two-phase constructive heuristics. In the recent decade, some meta-heuristics have been proposed such as simulated annealing (Eglese, 1994), tabu search (Hertz *et al*., 2000; Brandao and Eglese, 2008), variable neighborhood search (Polacek *et al*., 2008; Hertz and Mittaz, 2001), guided local search (Beullens *et al*., 2003), memetic algorithms (Lacomme *et al*., 2004) and ant colony optimization (Santos *et al*., 2010), etc. A recent review can be found in Liu *et al*. (2008) and Corberan and Prins (2010).

Applications of the CARP include winter gritting, street sweeping, garbage collection, mail delivery, meter reading, school bust routing and other pickup or delivery problems along the streets of a road network. However, the CARP has its limitation in modeling most real world applications. For instance, when more than one depot is used, the problem should be modeled as the multi-depot CARP (MDCARP) (Kansou and Yassine, 2010).

The MDCARP can be described as follows: given an undirected graph G = (V, E) with a set V of n nodes, an undirected edge set E, where $c_{ij} = c_{ji} (\geq 0)$ is the cost (length) of an edge $(I, j) \in E$ and $E_R \subseteq E$ is the set of t required edges (tasks). Let $D \subseteq V$ be the set of M depots. A fleet of homogeneous vehicles with capacity Q is stationed at multiple depots. The MDCARP is to determine a set of routes in such a way that: (1) Each vehicle route starts and ends at the same depot; (2) Each required edge is served exactly once; (3) The total demand of each route served by that vehicle does not exceed vehicle capacity Q; and (4) The total routing cost is minimized. The number of vehicles is a decision variable.

Our proposed approach in this Study is a hybrid Genetic Algorithm with Perturbation (HGAP) which integrates a Genetic Algorithm (GA), a Local Search (LS), a new replacement method, a perturbation mechanism. The structure of the remainder of this study is as follows: Section II gives the general framework and key components of the proposed HGAP. Computational experiments on benchmark data are presented in Section III and finally, conclusions are given in Section IV.

Corresponding Author: Tiantang Liu, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China

HYBRID GENETIC ALGORITHM

The proposed hybrid genetic algorithm with perturbation (HGAP) is a combination of a population-based global search GA and an individual-based local search. Several characters of our HGAP differ from the memetic algorithm (called MDMA) of Kansou and Yassine (2010) are as follows:

- Our chromosome is a permutation of t required edges (tasks), without route delimiters, while in the MDMA, the chromosome is a sequence S of w sub-chromosomes S_d where w is the number of depots and each S_d is a sequence of a S_d tasks. That is to say, the chromosomes use depots as delimiters, only without route delimiters for routes from a given depot
- Due to the different chromosomes structure mentioned above, we propose a MD-Partition to convert each chromosome into a MDCARP solution, while the MDMA applies the Split procedure of Lacomme *et al.* (2004) to each S_d of S to get a MDCARP solution
- In our HGAP, one chromosome P_r is selected in the parent population using binary tournament replacement described in the Section II-E and it is replaced by one child C if the child C is not a clone of any other chromosomes in the parent population, else a double swap perturbation is implemented on the child C to diversify the population
- The proposed HGAP includes two phases, i.e., a main phase and a restart phase. In the restart, the first, the third … and the $(ps-1)^{th}$ chromosomes of population are kept and the rest are replaced by new randomly generated chromosomes, then, the main phase is repeated but with a higher local search probability p_r

The general framework of the HGAP can be summarized in Fig. 1. Several main components are described in detail in Section II-A~E.

Chromosomes structure and evaluation: To describe the tasks clearly, each required edge is identified by being marked a task number instead of pairs of nodes. Each edge $u \in E$ has a tail (start node) a(u), a head (end node) b(u) and a traversing (deadheading) cost tc(u). Each required edge (task) $u \in E_R$ has a demand d(u), a serving cost sc(u) and an inverse mark inv(u). Task inv(u) and u have the same traversing cost, demand and serving cost. Note that each edge task $u \in E_R$ can be served in either direction, i.e., only one of task u and inv(u) is served.

Step 1: The initial population is constructed using two heuristics and random generation
Step 2: Each iteration selects two parents P1 and P2 randomly and then two children C1 and C2 are obtained using OX and only one of them is randomly selected as child C
Step 3: The child C undergoes the local search (LS) in a given probability p_m
Step 4: Two chromosomes are selected from the parent population and the worse one P_r is replaced by the child C if the child C is not a clone of any other chromosome in the parent population, else an double swap perturbation is implemented on the child C to diversify the population
Step 5: The main phase stops after a maximum number of iterations (ni). After that, the restart procedure is implemented for nr times, where the first, the third, the fifth … and the $(ps-1)^{th}$ chromosomes are kept, and others are replaced by new randomly generated chromosomes, and then the main phase is repeated but with a higher local search probability p_r. That is to say, the total number of iteration is $(1+nr) \times ni$

Fig. 1: General framework of HGAP

Our chromosome T is a permutation of t required edges (tasks), without route delimiters. Implicit shortest paths are between consecutive tasks. It can be viewed as a RPP or a giant tour. Under this kind of chromosomes structure, the chromosome must be converted into a CARP solution by a multi-depot partition (MD-Partition) procedure which corresponds to chromosome decoding and can evaluate the performance of each chromosome. The fitness is the total cost of this solution.

Given a chromosome $T = (c_1, c_2, ..., c_t)$ where t corresponds to the number of tasks, the MD-Partition procedure works on an auxiliary directed acyclic graph $G_a = (X, Y, Z)$, where X is a set of t+1 vertex index from a dummy node 0 to t. Y is a set of arcs where one arc $(i, j) \in Y$ means that a trip serving tasks subsequence $(c_{i+1}, c_{i+2}, ..., c_j)$ is feasible in terms of capacity, i.e., $load(i+1, j) < Q$ where load(i+1, j) is the load of the trip, respectively. Z is the set of the weight of arcs where one weight z_{ijm} corresponds to the total cost of the vehicle from depot m to serve tasks subsequence $(c_{i+1}, c_{i+2}, ..., c_j)$. The optimal MD-Partitioning of chromosome T corresponds to a shortest path from node 0 to node t in G_a. Thus, this problem is a Shortest Path Problem (SPP) which can be solved in pseudo-polynomial time based on Bellman's algorithm.

Consider one example of vehicles capacity Q = 30, 2 depots and 4 edge tasks with their respective demands being 8, 14, 8 and 9. Figure 2a shows the chromosome tour $T = (c_1, c_2, c_3, c_4) = (1, 2, 3, 4)$ with demands in brackets. Thin dotted lines represent shortest paths between any two nodes, the numbers under t = 4 tasks are the serving costs and depot1 and depot2 represent the depots (Fig. 2a). The MD-Partition procedure build an

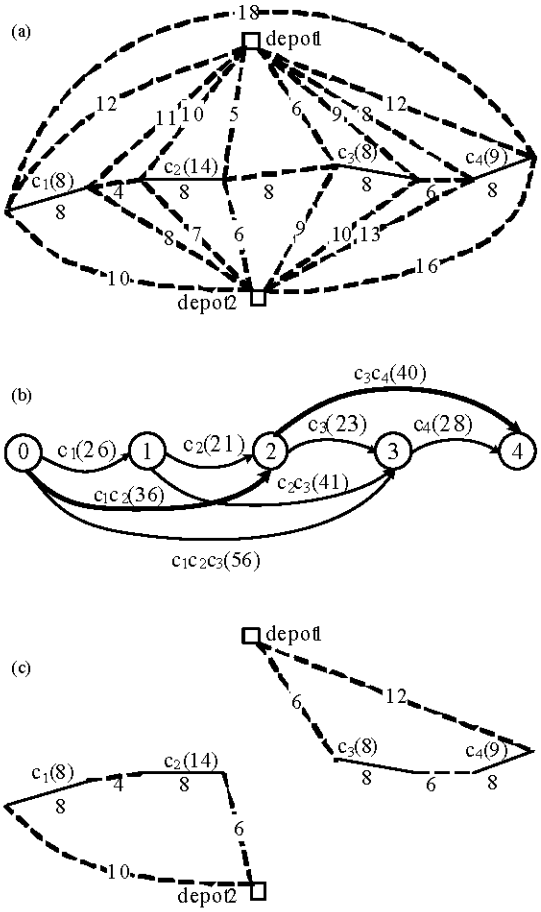

Fig. 2(a-c): Example of MD-Partition (a) Chromosome tour with 4 edge tasks (b) Auxiliary graph G_a, labels and shortest path and (c) Resulting trips

auxiliary graph G_a with t+1 nodes indexed from 0 to t, as shown in Fig. 2b where a convex downward (upward) arc represents that the vehicle from depot2 (depot1) serves the corresponding tasks. For example, Arc (0, 1) with cost = 26 represents the trip (0, c_1, 0), where the first 0 and the last 0 correspond to the depot2. A shortest path from node 0 to node t in G_a (bold) indicates the optimal MD-Partitioning of T: Two trips and total cost 76. The resulting MDCARP solution is trip (0, c_1, c_2, 0) with cost 36 served by the vehicle from depot2 and trip (0, c_3, c_4, 0) with cost 40 served by the vehicle from depot1, as shown in Fig. 2c.

Initial population: The population P is composed of ps chromosomes. The initial population consists of two good (low-cost) chromosomes and ps-2 random chromosomes. To be specific, two good chromosomes

i = 4 j = 7										
P1:	1	4	5	7	9	10	12	3	6	8
P2:	13	2	10	8	6	115	7	9	1	4
C1:	8	6	15	7	9	10	12	1	4	13
C2:	9	10	12	8	6	115	7	3	1	4

Fig. 3: Example of OX crossover

(RPP tours) P1 and P2 are constructed by using Frederickson heuristics (Frederickson, 1979).

All the above methods generate ps permutations of tasks (giant RPP tours) which are then converted into ps solutions by the MD-Partition procedure. Then each solution is concatenated into one chromosome where route delimiters (depots) are removed and all chromosomes are stored using an array in increasing cost order.

Selection and crossover: Two parents P1 and P2 are selected randomly and then a classic crossover Operatator, order crossover (OX) is used with slightly modification: first, the OX should take task u and inv(u) as the same edge when it finds them in two parents; second, the OX generates two children chromosomes which are both kept in the original OX, but only one child randomly chosen is preserved in our OX. Given two parents P1 and P2, the OX randomly selects two crossover points i and j ($1 \le i \le j \le t$) along the length of the chromosomes. The subsequence P1(i) to P1(j) is copied into C1(i) to C1(j) and P2 is scanned circularly from j+1 onwards to i-1, to fill C1 in parallel with the tasks missed in C1. The C2 can be obtained by exchanging the roles of P1 and P2. Finally, only one child C, randomly selected between C1 and C2, is kept. Figure 3 gives one sample of OX with i = 4, j = 7, where ten tasks are undirected and each edge task f is converted into two opposite arc tasks with inv(f) = 10+f, i.e., inv(1) = 11, inv(2) = 12 and so on. Therefore, when C1 is filled using P2(j+1) to P2(i-1), task 2 in P2 should be excluded because it's opposite arc 12 is the same task that has been included in C1.

Local search: A Local Search (LS) is adopted with a fixed probability p_m in our HGAP, to produce a better offspring after each crossover. The LS works on a CARP solution obtained by implementing the MD-Partition procedure on the child C, because if it Operatates directly on the chromosome C without route delimiters, a large amount of time will be spent to evaluate each move of it.

Let tasks i and j are served after tasks f and g in their respective routes and all move types are described below where M4 and M5 are not used in the MDMA of Kansou and Yassine (2010):

M1: move task f after task g
M2: move two consecutive tasks (f, i) after task g
M3: swap task f and g
M4: swap task f and (g, j)
M5: swap task (f, i) and (g, j)
M6: 2-opt moves

The LS scans each pair of tasks (f, g) in $o(t^2)$ and each iteration of the LS implements M1~M6 and stops when it finds the first improving move and then the solution is updated and the next iteration is continued until all pairs of tasks are scanned; however, the whole M1~M6 process for each pair of tasks are repeated as long as the solutions can be further improved.

There are several points to be noted in the LS. First, each type of move is implemented in the same route or in two different routes. Second, in M1-M5, if a task f is moved to another position, it can appear in either as f or inv (f). Third, in M1 and M2, g can be the start depot of its route. At last, some routes are removed if they become empty.

The final best feasible solution of LS is converted into a chromosome by concatenating these routes and excluding route delimiters (depots). Then, the chromosome is converted into a solution by the optimal MD-Partition which sometimes can bring a better solution for the same chromosome.

Replacement and restart: Inspired by binary tournament selection, we propose a new replacement method which can be called binary tournament replacement, i.e., two chromosomes are selected from the parent population and the worse one P_r is replaced by the child C if the child C is not identical as any other chromosomes of the parent population. After replacement, the ps chromosomes are stored in increasing cost order again.

In the replacement process, if the child C is a clone of a chromosome other than P_r of the parent population, one random double swap perturbation described is implemented on child C to avoid clone and diversify the population. The double swap perturbation randomly swaps two consecutive tasks with another two and the resulting chromosome is converted into a MDCARP solution by the MD-Partition and improved by the local search described in Section II-D. Then, the improved solution is concatenated into a new chromosome where route delimiters (depots) are removed. If the new chromosome is still a clone of a chromosome other than P_r of the parent population, then it replaces the chromosome that it ‘‘clones”.

The proposed HGAP includes two phases, i.e., a main phase and a restart phase. The main phase stops after a maximum number of iterations (ni). After that, the restart procedure is implemented for nr times, where the first, the third, the fifth … and the $(ps-1)^{th}$ chromosomes of population are kept and the rest are replaced by new randomly generated chromosomes. Then, the main phase is repeated but with a higher local search probability p_r. That is to say, the total number of iteration is $(1+nr) \times ni$. It is worth noting that the proposed partial replacement procedure in the restart phase can reserve good chromosomes and increase the diversity of population because any two adjacent chromosomes in the sorted population are often close to clones after many iterations of each phase.

COMPUTATIONAL EXPERIMENTS

Instances and parameters settings: Our HGAP are implemented in C and executed on an Intel (R) Pentium (R) Dual 1.4 GHz PC under Windows XP.

The benchmark instances set (mdgdb) for the MDCARP is available from Kansou and Yassine, 2010. The mdgdb set is 23 instances with 7~27 nodes, 11~55 edges which are all required edges (tasks).

By preliminary experiments, we determine the parameters of the HGAP as follows: the population size ps is 30, the maximum times try_max to generate each random non-clone chromosome is 10, the local search probability p_m and p_r in the main phase and restart phase are 0.2 and 0.4, respectively; the stopping criteria of the main phase is the maximum iteration number ni=2000, the maximum number of restarts nr is 2.

Computational results: In this section, we compare our HGAP with the MDMA and HACA on the mdgdb instances set of Kansou and Yassine, 2010 for the MDCARP and the gdb instances set for the CARP.

The results of the benchmark mdgdb set for the MDCARP and the gdb set for the CARP are presented in Tables 1 and 2, respectively. For each instance, the columns headed $|V|$ and $|E_R|$ contain the number of vertices and required edges, respectively. The columns headed LB and BKS in Table 2 list the lower bound and the best known solutions for the CARP reported in recent publications, respectively. The columns headed HACA, SEC_H, MDMA, SEC_M, HGAP and Sec. in Table 1 and 2 indicate the solutions obtained and computing times (in seconds) of each metaheuristic. In Table 1 and 2, if the solution of our HGAP is superior to that of the HACA and MDMA of Kansou and Yassine, 2010 then it is shown in bold.

Table 1: Results for the gdb set

Name	|V|	$\|E_R\|$	HACA	SEC_H	MDMA	SEC_M	HGAP	Sec
mdgdb1	12	22	300	<1	300	<1	300	<1
mdgdb2	12	26	331	<1	321	<1	321	<1
mdgdb3	12	22	267	<1	263	<1	**259**	<1
mdgdb4	11	19	266	<1	266	<1	266	<1
mdgdb5	13	26	364	<1	361	<1	361	<1
mdgdb6	12	22	285	<1	291	<1	**282**	<1
mdgdb7	12	22	325	<1	325	<1	325	<1
mdgdb8	27	46	350	1.5	350	1.7	**328**	4.3
mdgdb9	27	51	314	1.7	309	2.1	**279**	4.7
mdgdb10	12	25	275	<1	275	<1	275	<1
mdgdb11	22	45	407	<1	403	<1	**387**	1.9
mdgdb12	13	23	450	<1	440	<1	**420**	<1
mdgdb13	10	28	540	<1	540	<1	**528**	<1
mdgdb14	7	21	96	<1	96	<1	96	<1
mdgdb15	7	21	56	<1	56	<1	56	<1
mdgdb16	8	28	127	<1	127	<1	**125**	<1
mdgdb17	8	28	91	<1	91	<1	91	<1
mdgdb18	9	36	160	<1	158	<1	158	<1
mdgdb19	8	11	55	<1	55	1.1	55	<1
mdgdb20	11	22	121	<1	121	1.5	121	<1
mdgdb21	11	33	158	<1	158	1.8	**154**	2.8
mdgdb22	11	44	202	1.3	201	2.6	**196**	4.7
mdgdb23	11	55	235	1.7	235	3.2	**229**	6.4

Table 2: Results for the *gdb* set

Name	|V|	$\|E_R\|$	LB	BKS	HACA	SEC_H	MDMA	SEC_M	HGAP	Sec
gdb1	12	22	316	316	316	1	316	<1	316	<1
gdb2	12	26	339	339	339	<1	339	1.5	339	<1
gdb3	12	22	275	275	275	<1	275	1.3	275	<1
gdb4	11	19	287	287	287	1	287	1.3	287	<1
gdb5	13	26	377	377	377	<1	377	1.4	377	<1
gdb6	12	22	298	298	298	<1	298	1.2	298	<1
gdb7	12	22	325	325	325	<1	325	<1	325	1.0
gdb8	27	46	348	348	360	2	353	2.9	**348**	30.0
gdb9	27	51	303	303	333	2	319	3.1	**303**	26.3
gdb10	12	25	275	275	275	<1	275	1.2	275	1.1
gdb11	22	45	395	395	405	1	397	1.7	**395**	1.3
gdb12	13	23	458	458	468	<1	460	2.7	**458**	4.95
gdb13	10	28	536	536	544	1	544	1.6	**536**	5.88
gdb14	7	21	100	100	100	<1	100	<1	100	<1
gdb15	7	21	58	58	58	<1	58	<1	58	<1
gdb16	8	28	127	127	127	<1	127	1.8	127	<1
gdb17	8	28	91	91	91	<1	91	1.3	91	<1
gdb18	9	36	164	164	164	<1	164	1.5	164	<1
gdb19	8	11	55	55	55	<1	55	<1	55	<1
gdb20	11	22	121	121	121	<1	121	<1	121	1.1
gdb21	11	33	156	156	156	<1	156	1	156	1.4
gdb22	11	44	200	200	202	1	200	2.2	200	2.5
gdb23	11	55	233	233	237	3	237	3.2	**233**	16.2

The results show that the HGAP is superior to the HACA and the MDMA. For the MDCARP, the HGAP finds 11 new better solutions out of 23 instances in the mdgdb set; and for the CARP, the HGAP finds all the optimal solutions on 23 instances in the gdb set. The comparison between our HGAP, the HACA and MDAM shows that our computing times are a little more than the HACA and the MDMA but still very small.

The good results can be explained as follows: (i) Our chromosome is a permutation of t required edges (tasks) without route delimiters and a chromosome can be converted into an optimal MDCARP solution subject to chromosome sequence by the MD-Partition procedure; (ii) The perturbation mechanism to ensure a broad exploration of the search space; (iii) The binary tournament replacement and the second, the fourth, the sixth ... and the ps^{th} chromosomes replacement in the beginning of each restart phase increase the population diversity.

CONCLUSION

This study presents a Hybrid Genetic Algorithm with Perturbation (HGAP) for the Multi-depot Capacitated Arc Routing Problem (MDCARP). In the HGAP, we propose the perturbation mechanism, the binary tournament replacement and the second, the fourth, the sixth ... and

the ps^{th} chromosomes replacement in the beginning of each restart phase which all increase the population diversity and thus prevent the HGAP from trapping in a local optimum.

Benchmark instances for the MDCARP and the CARP are tested and the results show that the HGAP is competitive with existing metaheuristics and the computing times are reasonable. Our research can be further extended in several ways. First, an efficient lower bound can be provided to evaluate our HGAP. Second, more complex and practical constraints can be involved, such as time windows and periodic demands for tasks.

ACKNOWLEDGMENTS

This research is supported by the National Natural Science Foundation of China (No. 71201099, 71101090), Shanghai Pujiang Program (No. 13PJC066), the Ministry of Transport Research Projects (No. 2012-329-810-180), Doctoral Fund of the Ministry of Education Jointly Funded Project (20123121120002, 20123121120004), Shanghai Municipal Education Commission Project (No. 12ZZ148, 12ZZ149, 13YZ080), Shanghai Top Academic Discipline Project-Management Science and Engineering.

REFERENCES

Benavent, E., V. Campos, A. Corberan and E. Mota, 1990. The capacitated arc routing problem. A heuristic algorithm. Questiio, 14: 107-122.

Beullens, P., L. Muyldermans, D. Cattrysse and D. van Oudheusden, 2003. A guided local search heuristic for the capacitated arc routing problem. Eur. J. Operat. Res., 147: 629-643.

Brandao, J. and R. Eglese, 2008. A deterministic tabu search algorithm for the capacitated arc routing problem. Comput. Operat. Res., 35: 1112-1126.

Chapleau, L., J.A. Ferland, G. Lapalme and J.M. Rousseau, 1984. A parallel insert method for the capacitated arc routing problem. Operat. Res. Lett., 3: 95-99.

Corberan, A. and C. Prins, 2010. Recent results on arc routing problems: An annotated bibliography. Networks, 56: 50-69.

Eglese, R.W., 1994. Routeing winter gritting vehicles. Discrete Applied Math., 48: 231-244.

Frederickson, G.N., 1979. Approximation algorithms for some postman problems. J. ACM, 26: 538-554.

Golden, B.L. and R.T. Wong, 1981. Capacitated arc routing problems. Networks, 11: 305-315.

Golden, B.L., J.S. DeArmon and E.K. Baker, 1983. Computational experiments with algorithms for a class of routing problems. Comput. Operat. Res., 10: 47-59.

Hertz, A. and M. Mittaz, 2001. A variable neighborhood descent algorithm for the undirected capacitated arc routing problem. Transport. Sci., 35: 425-434.

Hertz, A., G. Laporte and M. Mittaz, 2000. A tabu search heuristic for the capacitated arc routing problem. Operat. Res., 48: 129-135.

Kansou, A. and A. Yassine, 2010. New upper bounds for the multi-depot capacitated arc routing problem. Int. J. Metaheuristics, 1: 81-95.

Lacomme, P., C. Prins and W. Ramdane-Cherif, 2004. Competitive memetic algorithms for arc routing problems. Ann. Operat. Res., 131: 159-185.

Liu, T., Z. Jiang, F. Chen, R. Liu and S. Liu, 2008. Combined Location-arc routing problems: A survey and suggestions for future research. Proceedings of the International Conference Service Operations and Logistics and Informatics, October 12-15, 2008, Beijing, pp: 2336-2341.

Pearn, W.L., 1989. Approximate solutions for the capacitated arc routing problem. Comput. Operat. Res., 16: 589-600.

Polacek, M., K.F. Doerner, R.F. Hartl and V. Maniezzo, 2008. A variable neighborhood search for the capacitated arc routing problem with intermediate facilities. J. Heuristics, 14: 405-423.

Santos, L., J. Coutinho-Rodrigues and J.R. Current, 2010. An improved ant colony optimization based algorithm for the capacitated arc routing problem. Transport. Res. B: Meth., 44: 246-266.

Ulusoy, G., 1985. The fleet size and mix problem for capacitated arc routing. Eur. J. Operat. Res., 22: 329-337.