THE VEHICLE ROUTING PROBLEM WITH TIME WINDOWS AND MULTIPLE DELIVERYMEN VIA ANT COLONY OTIMIZATION

Vitória Pureza
Departamento de Engenharia de Produção, Universidade Federal de São Carlos
Via Washington Luiz, km 235 – São Carlos, SP – CEP: 13565-905
vpureza@dep.ufscar.br

Marc Reimann
Institute of Production and Operations Management, University of Graz, Austria
marc.reimann@uni-graz.at

Reinaldo Morabito
Departamento de Engenharia de Produção, Universidade Federal de São Carlos
Via Washington Luiz, km 235 – São Carlos, SP – CEP: 13565-905
morabito@ufscar.br

Resumo

Consideramos uma variante do problema de roteamento de veículos com janelas de tempo na qual é permitida a designação de entregadores extras. A variante foi previamente proposta para tratar ambientes de distribuição com tempos de serviço longos quando comparados aos tempos de viagem, o que dificulta o atendimento de todos os pedidos dentro da jornada de trabalho com um único entregador por veículo. Descrevemos um algoritmo de otimização de colônia de formigas (ACO) para obter rotas de mínimo custo. Seu desempenho é comparado com o de uma abordagem de busca tabu utilizando-se exemplos gerados a partir de instâncias clássicas da literatura.

Palavras-chave: Roteamento de veículos com janelas de tempo, Múltiplos entregadores, Otimização de colônia de formigas.

Abstract

This work addresses a variant of the vehicle routing problem with time windows for which, besides routing and scheduling decisions, a number of extra deliverymen can be assigned to each route in order to reduce service times. This variant was previously proposed to tackle distribution environments characterized by long service times when compared to travel times, which makes it difficult to serve all daily service requests within regular working day hours with a single deliveryman. We describe an ant colony optimization (ACO) algorithm for obtaining minimum cost routes. The performance of the ACO algorithm is compared to a tabu search approach using instances generated from a set of classic examples from the literature.

Keywords: Vehicle routing with time windows, Multiple deliverymen, Ant colony optimization.

1. INTRODUCTION

The class of vehicle routing problems addresses situations characterized by a variety of operational characteristics, such as the type of distribution, service nature, constrained customer visiting hours and routing time, vehicle capacity limitations, and restricted types of vehicles for serving specific customers. Such diversity of variants and their inherent difficulty justifies the intense research effort and numerous reports comprising surveys of problem classification, solution methods and applications of vehicle routing. Examples of such material include the works of Bodin *et al.* (1983), Golden and Assad (1988), Laporte (1992), Osman (1993), Fisher (1995), Desroisiers *et al.* (1995), Cordeau *et al.* (2002), Golden *et al.* (2002), Toth and Vigo (2002), Bräysy and Gendreau (2005a, 2005b), Cordeau *et al.* (2007), Parragh *et al.* (2008), Laporte (2009) and Baldacci *et al.* (2010), among others.

This work addresses a variant of the vehicle routing problem with time windows (VRPTW) which allows a number of deliverymen to be assigned to each route in order to reduce service times. This variant is derived from real life applications for which daily requests must be delivered on the same day, the total operation cannot be completed within the maximum routing time and violations to the latter are highly undesirable. Service times are thus also a function of the vehicle's crew size rather than fixed for a given request. This variant is called the vehicle routing problem with time windows and multiple deliverymen (VRPTWMD).

The VRPTWMD appears in situations faced by Brazilian companies that deliver goods in urban areas on a regular basis, such as soft drinks, beer and tobacco companies. Given the difficulty in parking the vehicles in those areas, customers close to each other are seen as a demand cluster, and a single parking site is elected for the vehicle serving the cluster. The goods are then delivered at each demand site of a given cluster by the vehicle's driver who visits the customers on foot. In these situations, typical service times are relatively large when compared to the vehicle traveling time, which very often implies that part of the total orders cannot be delivered during the driver's regular working hours.

The assignment of multiple deliverymen to each route is justified by the fact that it most likely affects individual service times at each demand site of the cluster, as well as the total service time associated to the cluster. The reduction of service times impacts the end of service, and consequently, the number of clusters that can be served during working hours on a given day. Note that in addition to the usual vehicle routing and scheduling decisions, a solution for the VRPTWMD also includes the number of deliverymen on each route.

Vehicle routing with multiple deliverymen is not a feature included in any current commercial routing systems despite its potential practical applications (Pureza and Morabito, 2008), and it has only been addressed in recent papers. In Ferreira and Pureza (2012), a modified savings algorithm and a tabu search algorithm are designed to tackle the vehicle routing problem (VRP). In Pureza *et al.* (2012), a mathematical model, a tabu search approach and an ant colony optimization (ACO) algorithm are proposed for the VRPTWMD. In this study we present the ACO algorithm and compare its computational performance to the tabu search approach (previously presented in Pureza and Morabito, 2010) by means of sets of examples with time windows based on the instances proposed in Solomon (1987).

The remainder of this paper is organized as follows. Section 2 briefly discusses the VRPTWMD tackled in this research. The ACO approach is presented in Section 3. In Section 4, we discuss the computational experiments and results obtained from the chosen sets of instances, followed by conclusions and perspectives of research in Section 5.

2. THE VEHICLE ROUTING PROBLEM WITH TIME WINDOWS AND MULTIPLE DELIVERYMEN

The VRPTWMD consists of the definition of minimum cost routes for a fleet of vehicles in order to satisfy a set of customer clusters with known demands for delivery of goods. In each route, service is performed by the vehicle's crew whose size must not exceed the cabin's capacity. All trips start from a single central depot, to which all vehicles should return after visiting the assigned clusters. All available vehicles may be used and each vehicle performs only one route.

We assume that each cluster (consisting of one or various customers) is defined by the user and its total demand must respect the vehicle's capacity to the route, as well as the total time of each route (comprising vehicle traveling times and service times in each cluster) should not violate a predefined limit associated to the end of the workday. Service times of each cluster are input parameters computed by the user, whose values depend on the crew's size, the deliverymen service strategy, and the cluster characteristics. We also assume that the total demand of each cluster must be met by a single vehicle within a given period of the day (time window) defined by the service deliverer and the cluster's customers.

The primary objective of the VRPTWMD is to minimize the fleet size (fixed cost). The second objective consists of the minimization of the total number of deliverymen, followed by the total distance traveled by the fleet (variable cost).

3. AN ANT COLONY OPTIMIZATION APPROACH

The heuristic approach for the VRPTWMD considered in this work consists of an implementation of Ant Colony Optimization (ACO). ACO was first proposed in Dorigo *et al.* (1996) as a population-based metaheuristic. Its motivation stems from the underlying metaphor concerning the collective behavior of real ant colonies leading to the exploitation of rich food sources. An overview of different variants of ACO can be found in Dorigo and Stützle (2004).

The ACO approach used in this research is a modification and adaptation of the algorithm originally proposed in Reimann *et al.* (2002). Specifically, the objective function and the local search were modified to allow the minimization of the number of deliverymen. A high-level description of the algorithm is shown in Figure 1 and discussed in the following sections.

1. Read the input data.
2. Initialize parameters and pheromone matrix (discussed in section 3.3).
3. Repeat until a pre-specified stopping criterion is met:
   3.1. For each ant:
      3.1.1. Construct a feasible solution (discussed in section 3.1)
      3.1.2. Apply Local Search (discussed in section 3.2.).
      3.1.3. Update the best found solution (if applicable).
   3.2. Update the pheromone matrix (discussed in section 3.3).
4. Return the best found solution.

Figure 1 – A high-level description of the ACO algorithm.

3.1. AN INSERTION BASED SOLUTION CONSTRUCTION

The solution construction is based on the I1-heuristic for the VRPTW from Solomon (1987). In this algorithm routes are constructed sequentially one by one. In the context of ACO this algorithm was modified in the following way (for further details, see also Reimann *et al.* (2002)). From the set of yet unserved nodes N_u, a node i is inserted probabilistically into the current route k according to the following random proportional rule:

$$P_{ik} = \frac{\kappa_{ik}}{\sum_{h \in N_u} \kappa_{hk}}$$

where κ_{ik} is a function of both the heuristic information and the pheromone level associated with inserting node i into route k. The computation of κ_{ik} differs depending on whether a new route is initialized or a current route is extended. In the former case it is given by:

$$\kappa_{ik} = \frac{1}{d_{1i}} \frac{(\tau_{1i} + \tau_{il})}{2\tau_{11}}, \quad \text{for } i \in N_u$$

while in case of a route extension it is computed as:

$$\kappa_{ik} = \max_{j \in R_{ik} \cup \{1\}} \left\{ \max\left[0, 2d_{1i} - \left(d_{ji} + d_{ij'} - d_{jj'}\right)\right] \frac{\left(\tau_{ji} + \tau_{ij'}\right)}{2\tau_{jj'}} \right\}, \quad \text{for } i \in N_u$$

where R_{ik} denotes the set of customers already assigned to route k after which node i could be feasibly inserted, and j' denotes the current successor of j in route k, while the pheromone information τ_{ji} represents the learned desirability of visiting a node i immediately after a node j on the same route. Thus, for each unserved node i, the best insertion position along route k is selected deterministically and the random proportional rule described above is only used to select which node to insert. Only if no more feasible insertions are possible in a route a new route is initialized. The algorithm stops once all nodes have been assigned to routes.

The solution construction assigns the maximum crew size (if available) to each route in order to keep the required fleet size as small as possible. Still, the solution returned by an ant may feature routes that actually can be feasibly performed with a smaller number of deliverymen. In such a case the number of deliverymen is iteratively decremented until a further reduction results in an infeasible route. The complete solution construction procedure is depicted in Figure 2.

1. Let N_u be the set of unrouted nodes, *tcrew* be the number of available deliverymen, $crew_k$ be the number of deliverymen in each route *k,* and *maxcrew* be the maximum crew size. Initialize solution $S = \emptyset$ and $N_u = \{2, \dots, n+1\}$.
2. Initialize the first route k = 1 with $crew_k = Min\,\{tcrew,\ maxcrew\}$. Set $= tcrew - crew_k$.
3. Repeat until $N_u = \emptyset$:
3.1. For each node i in N_u compute the best feasible insertion position in the current route k according to equations (11) or (12). If the node cannot be inserted feasibly set $\kappa_{ik} = 0$.
3.2. If $\kappa_{ik} = 0$ for all i in N_u, initialize a new route $k = k+1$, set $crew_k = Min\,\{tcrew,$ $maxcrew\}$ and go to step 3.1.
3.3. Otherwise, probabilistically select a node i' in N_u according to the random proportional rule (10).
3.4. Insert node i' into route k at its best possible insertion position.
3.5. Reschedule route k.
3.6. Update N_u and make $S = S + k$.
4. For each route in *S*, reduce the number of deliverymen and reschedule the route until any further reduction makes the route infeasible.
5. Return solution *S*.

Figure 2– Solution construction and reduction of deliverymen by each ant in the ACO algorithm.

3.2. LOCAL SEARCH

After an ant has constructed a feasible solution, a local search algorithm is applied to that solution in order to improve its quality. For this end, λ-interchange operators (Osman, 1993) with $\lambda = 1, 2$ are sequentially applied. Infeasible solutions are not permitted and a first improvement strategy is employed. The algorithm first tries to move $\lambda \leq 2$ nodes from the routes with the smallest number of nodes to the other routes with the goal to eliminate some of the routes, therefore reducing the number of vehicles. Then move and swap operators with $\lambda = 1$ are applied to improve the routing.

This local search also comprises an additional move neighborhood, which ejects one customer from a route with 2 or 3 deliverymen whenever this reduces the deliverymen on this route. Then an attempt is made to insert this node into another route. If such an insertion is possible without (or with a smaller) increasing the number of deliverymen on the new route, the move is accepted.

3.3. PHEROMONE INITIALIZATION AND UPDATE

In the constructive phase of the ACO algorithm, decisions are based on both heuristic information and the pheromone values as described above. At the end of each iteration (all ants have gone through solution construction and local search), the pheromone update procedure is applied to these pheromone values. The pheromone management used in the algorithm is related to the Hypercube Framework presented in Blum and Dorigo (2004) and the Max–Min Ant System (see, e.g., Stützle and Hoos, 1999) and a variant of it was first presented in Reimann (2003). The pheromone update rule can be formally written as:

$$\tau_{ij} = \rho\tau_{ij} + (1-\rho)\Delta\tau_{ij}, \text{ for } (i,j) \in E$$

where E is the set of all edges, $0 \leq \rho \leq 1$ is called the trail persistence and $\Delta\tau_{ij}$ is the amount of reinforcement, which is defined as:

$$\Delta\tau_{ij} = \begin{cases} 1 & if\ (i,j) \in S^* \\ 0 & \text{otherwise} \end{cases}$$

where S^* is the best solution found up to the current iteration (regardless if it was found in the current iteration or earlier). At the beginning of the run, all pheromone values are initialized as:

$$\tau_{ij} = 1, \text{ for } (i,j) \in E$$

4. COMPUTATIONAL EXPERIMENTS

In this section, we compare the computational results obtained by ACO and the tabu search approach (TS), also proposed in Pureza *et al.* (2012). TS uses another modified version of the I1-heuristic for the VRPTW from Solomon (1987) to generate the initial solution, and its improvement phase employs an integrated intensification/diversification mechanism that modifies selected tabu search parameters (Glover & Laguna, 1997) based on the analysis of search trajectory patterns. The reduction of the number of vehicles, crew or distance are handled in different phases of the search. For further details on TS, the reader should refer to Pureza *et al*. (2012).

For these experiments, we used the following VRPTW data sets proposed in Solomon (1987): R1 and R2 (randomly generated geographical distribution of customer nodes), C1 and C2 (clustered problem sets), and RC1 and RC2 (mix of random and clustered structures), with $n = 100$ nodes and made up of 12, 11, 9, 8, 8 and 8 instances, respectively. The maximum crew size in each vehicle was set to $L = 3$. The original data was maintained except for the service times in each node *i*, which were replaced by:

$$ts_i = \min\{rs * q_i, T - \max\{a_i, tv_{1i}\} - tv_{i1}\} \quad (1)$$

where q_i is the product demand of node i and rs is the rate of service (i.e., number of products picked up per time unit in each node – in the experiments we used $rs = 2$). Note that ts_i in (1) is proportional to the demand q_i of node i when the first term on the right-hand-side of (1) is the minimum. The second term in (1) is to ensure that this service time does not turn the instance infeasible regarding the maximum return time T. In the experiments, the value of ts_i in (13) was considered the service time in node i and in mode 1, i.e., $ts_{i1} = ts_i$. For the sake of simplicity, for the cases with two and three deliverymen (modes 2 and 3), we simply divided ts_i by 2 and 3, so that $ts_{il} = ts_i/l$ for $l = 1,2,3$.

ACO was coded in C and run with a time limit of 600 seconds under the following (standard) parameter settings: 50 ants and a trail persistence of $\rho = 0.975$. For both algorithms, each instance was solved using up to five different initial solutions and the experiments were run on a microcomputer Intel Core2 2.4 GHz with 2 GB RAM. The performances of the approaches were then evaluated in terms of the lexicographic order of the objectives <fleet size, number of deliverymen, total distance> and the computer runtime requirements.

First we analyze the results obtained for the instances of set R1, which are characterized by tight time windows and randomly generated geographical distribution of the nodes. Note that short planning horizons (set type 1) better reflect the importance of the service times since the routes are shorter in general. Table 1 presents the results of ACO and TS for this set considering the number of deliverymen sufficiently large (i.e., $M = 50$). For each instance and method, columns Veh, Dmen, Dist and Cpu depict the best solution regarding the number of vehicles used, the number of deliverymen assigned, the total distance traveled and the runtime required (in seconds), respectively. The best solution for each instance is highlighted in bold.

The results show that ACO and TS have equivalent performances in terms of the number of vehicles (TS outperformed ACO in only 2 out of the 12 instances), number of deliverymen (ACO was slightly better than TS in 7 out of 12 instances) and distance (TS was slightly better than ACO in 7 out of 12 instances). Note also that the average results of these 12 instances obtained by TS and ACO and presented in the last row of Table 1 are close to each other.

Table 1 – Computational results of TS and ACO for data set R1.

Inst.	TS (Best run out of 5)				ACO (Best run out of 5)			
	Veh	Dmen	Dist	Cpu (s)	Veh	Dmen	Dist	Cpu (s)
R101	19	45	1740	645	**19**	**45**	**1721**	656
R102	**17**	**39**	**1520**	655	17	39	1527	655
R103	13	32	1285	959	**13**	**30**	**1341**	592
R104	**10**	**28**	**1057**	692	11	26	1090	627
R105	14	37	1446	463	**14**	**36**	**1413**	157
R106	12	31	1323	492	**12**	**31**	**1308**	758
R107	**10**	**29**	**1112**	473	10	30	1145	770
R108	10	27	967	953	**10**	**25**	**1029**	494
R109	12	33	1296	428	**12**	**31**	**1244**	431
R110	11	30	1217	620	**11**	**30**	**1174**	517
R111	**10**	**30**	**1137**	616	11	27	1142	536
R112	10	28	996	686	**10**	**27**	**1003**	716
Avg.	12.3	32.4	1258.1	640.1	12.5	31.4	1261.5	575.8

In Table 2, Best z correspond to the best solution values (z values) obtained by ACO and ACO. As both are randomized algorithms, in order to give an idea of their robustness, the remaining columns of the table present the average values and standard deviations (in parenthesis) of the numbers of vehicles and deliverymen, the distances traveled and the execution times, considering the five runs of the algorithms. The best average solution for each instance is highlighted in bold. Note that the standard deviations are relatively small if compared to the averages. Note also in the last row of Table 2 that ACO presents a slightly better average performance than TS.

Table 2 – Additional results of TS and ACO for data set R1.

Inst.	TS Best and average values (standard deviation)					ACO Best and average values (standard deviation)				
	Best z	Avg. Veh	Avg. Dmen	Avg. Dist	Cpu (s)	Best z	Avg. Veh	Avg. Dmen	Avg. Dist	Cpu (s)
R101	23.67	20.2 (1.2)	44.8 (2.4)	1750.7 (31.3)	739.1 (121.5)	23.67	**19.2 (0.5)**	**45.0 (0.7)**	**1727.9 (8.0)**	371.4 (182.0)
R102	21.05	18.0 (0.6)	38.0 (1.3)	1581.8 (40.5)	958.0 (169.8)	21.05	**17.0 (0.0)**	**39.0 (0.0)**	**1541.3 (14.1)**	598.2 (203.3)
R103	16.33	13.6 (0.5)	31.8 (1.0)	1319.8 (23.1)	865.7 (82.4)	16.13	**13.0 (0.0)**	**30.6 (0.6)**	**1353.5 (10.7)**	562.6 (184.9)
R104	12.91	**10.6 (0.5)**	**29.0 (0.9)**	**1066.8 (10.9)**	802.0 (85.6)	13.71	11.0 (0.0)	26.4 (0.6)	1082.7 (40.1)	498.2 (167.0)
R105	17.84	14.6 (0.5)	37.2 (0.7)	1458.9 (30.1)	543.0 (102.4)	17.74	**14.0 (0.0)**	**36.4 (0.6)**	**1418.0 (8.3)**	387.4 (215.3)
R106	15.23	12.0 (0.0)	32.6 (1.0)	1320.9 (8.9)	513.7 (19.5)	15.23	**12.0 (0.0)**	**31.4 (0.6)**	**1304.6 (18.2)**	459.2 (272.0)
R107	13.01	**10.7 (0.5)**	**29.3 (0.5)**	**1125.5 (13.3)**	618.4 (98.3)	13.11	10.8 (0.5)	28.0 (1.2)	1141.3 (8.9)	588.4 (212.7)
R108	12.80	10.0 (0.0)	27.4 (0.5)	994.8 (36.5)	711.7 (158.9)	12.60	**10.0 (0.0)**	**25.8 (0.5)**	**1013.8 (11.8)**	699.6 (145.3)
R109	15.43	**12.0 (0.0)**	**33.4 (0.5)**	**1298.3 (36.5)**	556.6 (98.2)	15.22	12.2 (0.5)	31.4 (0.9)	1245.4 (24.3)	367.6 (112.0)
R110	14.12	**11.0 (0.0)**	**30.8 (0.7)**	**1196.6 (23.4)**	643.8 (31.3)	14.12	11.0 (0.0)	31.0 (0.7)	1163.7 (21.0)	539.8 (165.9)
R111	13.11	**10.8 (0.4)**	**29.8 (0.4)**	**1140.4 (5.8)**	632.3 (70.8)	13.81	11.0 (0.0)	27.8 (0.5)	1132.9 (23.9)	503.6 (156.8)
R112	12.90	10.0 (0.0)	28.6 (0.8)	1014.8 (17.5)	638.0 (89.9)	12.80	**10.0 (0.0)**	**27.2 (0.5)**	**1032.7 (21.2)**	530.0 (239.8)
Avg.	15.70	12.8	32.7	1272.4	685.2	15.77	**12.6**	**31.7**	**1263.2**	508.8

Table 3 presents the average of the best results of ACO and TS obtained for the instances of the remaining sets R2, C1, C2, RC1 and RC2, together with the average of the best results for set R1. Note that the average values of ACO are slightly worse than the ones of TS, although one algorithm does not dominate the other. The solutions of ACO outperformed the solutions of TS in 8 out of the 12 instances of set R1 and in 8 out of the 11 instances of set R2. Regarding sets C1 and C2, the TS solutions outperformed the ACO solutions in 5 out of the 9 instances of C1 and in 6 out of the 8 instances of C2. For sets RC1 and RC2, ACO was better than TS in 5 out of the 8 instances of RC1, but worse than TS in 5 out of the 8 instances of RC2.

Table 3 – Average results of TS and ACO for data sets R1, R2, C1, C2, RC1 and RC2.

Set	TS (Best run out of 5)				ACO (Best run out of 5)			
	Veh	Dmen	Dist	Cpu (s)	Veh	Dmen	Dist	Cpu (s)
R1	**12.3**	**32.4**	**1258.1**	640.1	12.5	31.4	1261.5	575.8
R2	**2.9**	**7.5**	**1034.0**	425.4	3.1	6.5	1064.2	600.6
C1	**10.0**	**10.0**	**830.7**	265.1	10.0	10.0	833.6	375.2
C2	**3.0**	**3.0**	**597.2**	236.8	3.0	3.0	609.3	243.3
RC1	**13.0**	**34.9**	**1527.9**	677.1	13.0	35.5	1480.1	508.6
RC2	**3.4**	**9.3**	**1230.4**	419.1	3.6	8.5	1296.0	462.0

In Table 4, Best z corresponds to the average of the best solution values (z values) obtained by TS and ACO, while the remaining columns depict the average values of the numbers of vehicles and deliverymen, the distances traveled and the execution times, considering the five runs of the algorithms. The best average values (of the five solutions) are highlighted in bold.

Table 4 – Additional results of TS and ACO for data sets R1, R2, C1, C2, RC1 and RC2.

Set	TS Best and average values					ACO Best and average values				
	Best *z*	Avg. Veh	Avg. Dmen	Avg. Dist	Cpu (s)	Best *z*	Avg. Veh	Avg. Dmen	Avg. Dist	Cpu (s)
R1	15.70	12.8	32.7	1272.4	685.2	15.77	**12.6**	**31.7**	**1263.2**	508.8
R2	3.76	**3.0**	**7.9**	**1046.8**	393.0	3.84	3.1	6.8	1070.3	527.5
C1	11.1	10.0	10.0	847.7	245.7	11.1	**10.0**	**10.0**	**838.8**	389.1
C2	3.4	3.0	3.0	653.5	261.9	3.4	**3.0**	**3.0**	**623.7**	263.8
RC1	16.6	13.4	35.7	1511.4	686.5	16.7	**13.4**	**35.3**	**1496.3**	472.8
RC2	4.4	**3.4**	**9.7**	**1251.3**	400.5	4.6	3.7	8.6	1307.9	455.6

ACO was also applied to sets R1, R2, C1, C2, RC1 and RC2 with their original service times and with a single deliveryman in order to evaluate its performance of the algorithms when tackling the classical VRPTW. The average number of vehicles and the average distance traveled obtained by ACO for these sets were 7.59 and 1016.8, respectively, while the best known results are 7.23 and 1021.1, respectively (http://www.sintef.no/Projectweb/TOP/Problems/VRPTW/Solomon-benchmark/). Since the best-known solutions are taken from several papers featuring highly specialized algorithms for the VRPTW and ACO was designed for tackling the additional decision of assigning deliverymen, the results obtained can be considered reasonable.

5. CONCLUSIONS AND PERSPECTIVES

In this study we discussed a solution approach tailored for the VRPTWMD based on ant colony optimization. The performance of the algorithm was compared to a tabu search approach and discussed by means of computational experiments comprising 100 demand cluster instances derived from classic VRPTW. The performances of the heuristics were comparable when considering the lexicographic order of objectives <fleet size, number of deliverymen, total distance>, and we conclude that one algorithm does not dominate the other.

An interesting perspective for future research is to extend ACO to deal with more general VRPTWMD cases of heterogeneous fleet and multiple depots. Other interesting lines of research are to adapt the approach to deal with the problem in situations with simultaneous (or mixed) pickup and delivery routes and when the objective is to maximize the total number of serviced customers within regular working hours. In addition, a study of how service times vary with the cluster demand and configuration, the crew size, and the delivery strategy is mandatory for supporting vehicle routing with deliverymen decisions in practice.

6. REFERENCES

[1] Baldacci, R., Toth, P., Vigo, D. (2010). Exact algorithms for routing problems under vehicle capacity constraints. *Annals of Operations Research*, 175, 213-245.

[2] Blum, C., Dorigo, M. (2004). The hyper-cube framework for ant colony optimization, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 34, 1161-1172.

[3] Bräysy, O., Gendreau, M. (2005a), Vehicle routing problem with time windows, part I: Route construction and local search algorithms, *Transportation Science*, 39:1, 104-118.

[4] Bräysy, O., Gendreau, M. (2005b), Vehicle routing problem with time windows, part II: Metaheuristics, *Transportation Science*, 39:1, 119-139.

[5] Bodin, L., Golden, B., Assad, A., Ball, M. (1983), Special issue – Routing and scheduling of vehicles and crews – the state of the art, *Computers & Operations Research*, 10.

[6] Cordeau, J-F., Desaulniers, G., Desrosiers, J., Solomon, M. M., F. Soumis (2002), The VRP with time windows. In Toth, P., Vigo, D. (Eds.), *The vehicle routing problem. SIAM, Monographs on Discrete Mathematics and applications*, Philadelphia, PA, SIAM, 157-193.

[7] Cordeau, J-F., Laporte, G., Savelsbergh, M., Vigo, D. (2007). Vehicle routing. In *Transportation, Handbooks in Operations Research and Management Sciences*, vol. 14, Elsevier, Amsterdam, 367-428.

[8] Desroisiers, J., Dumas, Y., Solomon, M., Soumis, F. (1995). Time constrained routing and scheduling. In Ball, M. O., Magnanti, T. L., Monma, C. L., Nemhauser, G. L. (Eds.), *Network routing. Handbooks in operations research and management science*, Amsterdam, North-Holland, 35-139.

[9] Dorigo, M., Maniezzo, V., Colorni, A. (1996). The ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26, 29-41.

[10] Dorigo, M., Stützle, T. (2004). *Ant Colony Optimization*. MIT Press/Bradford Books, Cambridge, MA.

[11] Ferreira, V. O., Pureza, V. (2012). Some experiments with a savings heuristic and a tabu search approach for the vehicle routing problem with multiple deliverymen. *Pesquisa Operacional*, http://dx.doi.org/10.1590/S0101-74382012005000016.

[12] Glover, F., Laguna, M. (1997), *Tabu Search.* Kluwer Academic Publishers, Massachusetts.

[13] Golden, B., Assad, A. (1988). *Vehicle routing: Methods and studies.* North-Holland, Amsterdam.

[14] Golden, B., Assad, A., Wasil, E. (2002), The vehicle routing problem. *SIAM, Monographs on Discrete Mathematics and applications*, Philadelphia, PA, 245-286.

[15] Laporte, G. (1992), The vehicle routing problem: an overview of exact and approximate algorithms, *European Journal of Operational Research*, 59, 345-358.

[16] Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43, 4, 408-416.

[17] Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41, 421-452.

[18] Parragh, S., Doerner, K., Hartl, R. (2008). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58, 1, 21-51, doi 10.1007/s11301-008-0033-7.

[19] Pureza, V., Morabito, R. (2008). Desenvolvimento de um modelo matemático e um protótipo computacional para planejamento de rotas de atendimento. *Technical Report of Project Transmute* 863*04, FAI-UFSCar, São Carlos, Brazil.

[20] Pureza, V., Morabito, R. (2010). Designando entregadores extras no roteamento de veículos com janelas de tempo. *Anais do XLII Simpósio Brasileiro de Pesquisa Operacional*, Bento Gonçalves, Brazil.

[21] Pureza, V.; Morabito, R., Reimann, M. (2012). Vehicle routing with multiple deliverymen: modeling and heuristic approaches for the VRPTW. *European Journal of Operational Research* 218:3, 636–647.

[22] Reimann, M. (2003). Combining an exact algorithm with an ant system for travelling salesman problems. *Working paper*, University of Vienna.

[23] Reimann, M. Dörner, K.F., Hartl, R.F. (2002). Insertion based ants for vehicle routing problems with backhauls and time windows, in Dorigo, M., Di Caro, G., Samples, M. (Eds.): *Ant Algorithms*, Springer LNCS 2463, Berlin/Heidelberg, 135-147.

[24] Solomon, M. (1987), Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research*, 35:2, 254-265.

[25] Stützle, T., Hoos, H. (1999). The max-min ant system and local search for combinatorial optimization problems, in Voß, S., Martello, S., Osman, I.H., Roucairol, C. (Eds.): *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Boston, 313-329.

[26] Toth, P., Vigo, D. (Eds.) (2002), The vehicle routing problem. *SIAM, Monographs on Discrete Mathematics and Applications*, Philadelphia, PA, 157-193.