The Car Renter Salesman Problem: An Algorithmic Study

Paulo Henrique Asconavieta da Silva
Instituto Federal de Ciência e Tecnologia Sul-Riograndense
Praça 20 de Setembro, 455 – Pelotas – RS
Universidade Federal do Rio Grande do Norte
Campus da Lagoa Nova, Natal – RN
paulohas@pelotas.ifsul.edu.br

Marco César Goldbarg
Universidade Federal do Rio Grande do Norte
Campus da Lagoa Nova, Natal – RN
gold@dimap.ufrn.br

Elizabeth Ferreira Gouvêa Goldbarg
Universidade Federal do Rio Grande do Norte
Campus da Lagoa Nova, Natal – RN
beth@dimap.ufrn.br

Abstract

The Car Renter Salesman Problem (CaRS) is a new variant of the classic Traveling Salesman Problem where the salesman's tour can be decomposed into contiguous paths that are travelled by different rented cars. The goal is to determine the minimum cost Hamiltonian cycle, considering the cost of the route added to the cost of penalties paid for exchanging vehicles during the route. The penalty is due to fees paid to return the rented cars to their bases. This paper describes the general problem and some related variants. A GRASP hybridized with a Variable Neighborhood Descent procedure and a Memetic Algorithm are proposed for the new problem. The results of a computational experiment on a set with 40 Euclidean and non-Euclidean instances are reported.

Keywords: Car Renter Salesman Problem, Hamiltonian Cycle, Evolutionary Computation, Metaheuristic Algorithms.

1. Introduction

The Traveling Salesman Problem (TSP) is a classic Combinatorial Optimization problem. Given a graph $G=(N,M)$, where $N=\{1,...,n\}$ is the set of nodes and $M=\{1,...,m\}$ is the set of edges, and costs, c_{ij}, associated with each edge linking vertices i and j, the problem consists in finding the minimum length Hamiltonian cycle. The TSP is NP-hard (Garey&Johnson, 1979) and one of the combinatorial optimization problems more intensively investigated. The size of the larger non trivial TSP instance solved by an exact method evolved from 318 cities in the 80's (Crowder & Padberg, 1980), to 7397 cities in the 90's (Applegate *et al*., 1994) and 24978 cities in 2004. The best mark was reached in 2006 with the solution of an instance with 85900 cities (Applegate *et al*., 2006).

The TSP has several important practical applications and a number of variants (Gutin & Punnen, 2004). Some of these variants are classic such as the peripatetic salesman (Krarup, 1975) and the M-tour TSP (Russel, 1977) and others are more recent such as the Colorful TSP (Xiong *et al*., 2007) and the Robust TSP (Montemanni *et al*., 2007), among others.

The Car Renter Salesman Problem (CaRS) is a new TSP variant that both models important applications in the areas of tourism and transportation in manufacturing plants, as it represents a complex variant that challenges the state of the art. The CaRS Problem is introduced in Section 2, where several conditions under which this variant can be presented are examined. The Section 3 presents two metaheuristic methods for the investigated problem. These heuristics that establish the first limits for a set of instances named CaRSLib are compared and their results presented in Section 4. Final conclusions are presented in Section 5.

2. The Car Renter Salesman Problem

2.1 The rental car industry

Today over 90 significant economic size car rental companies exist in the world market (Car, 2008). The importance of the car rental business can be measured both by the enterprise turnover as by the size of the companies that provide the service. For example, Hertz is a company in the car rental segment with wide accessibility of providing the services at approximately 8,000 locations in approximately 145 countries (Hertz, 2009). The Enterprise has more than 878,000 vehicles in its rental and leasing fleet and operates across 6,900 local markets (Enterprise, 2009). Avis operates in more than 3,800 locations all over Europe, Africa, the Middle East and Asia. In December 2007, the company operated an average fleet of 118,000 vehicles (Avis, 2009). Avis Budget Group Inc. earned $ 5.1 billion dollars in 2009 (Avis, 2010). In 2009, the Enterprise Holdings Inc. which owns today the National Car Rental, Alamo Rent A Car and WeCar earned about 12.1 billion dollars (Conrad & Perlut (2006); Wikipedia, 2010). These numbers represent only part of the market that also has other major car rental networks such as Dollar and Hertz. The world market in 2012 is estimated at 52.6 billion dollars (Car Rental, 2008).

Besides being itself a major business, spending on car rentals may represent a significant portion of the activities involving tourism and business. Currently the rental options are becoming increasingly diversified with the expansion of the companies, justifying the search for rent schemes that minimize the total cost of this form of transport.

2.2 Models of Combinatorial Optimization in the rental car industry

Among the various logistical problems of this branch of activity, the literature describes specific studies of combinatorial optimization in the Fleet Assignment Problem (Lia &Taob, 2010), the Strategic Fleet Planning and Tactical Fleet Planning (Pachon *et al.*, 2003), the Demand Forecast (Edelstein & Melnyk, 1977) and the car fleet management problem with maintenance constraints (Hertz *et al*. 2009). Logistic problems that occur in the car rental industry are reviewed by Yang *et al*. (2008). These studies focus on the viewpoint of the car rental industry, however, the customer's point of view has not yet been the subject of published research.

2.3 The Car Renter Salesman Problem

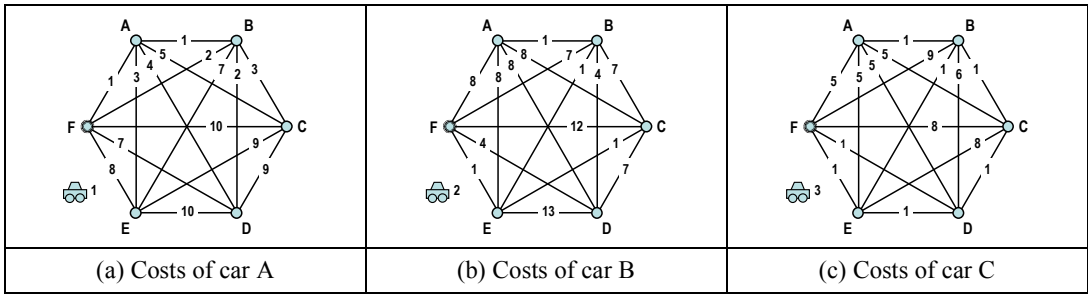
In general, under the viewpoint of a user of rented cars, the goal is to minimize the costs to move from a starting point to a destination. On the other hand, when someone rents a car, it is assumed that it meets the requirements of comfort and safety. During the travel, in addition to the costs of renting the car, at least the costs of fuel and the payment of fees to travel on the road should be considered. Let $G=(N, M, W)$ be a graph where $N=\{1,...,n\}$ represents the set of vertices, $M=\{1,...,m\}$ is the set of edges and $W=\{1,...,w\}$ is the set of distances between the vertices or the length of the edges of the set M. The problem described in this paper has the following features:

1. Several types of cars are available for rent, each of them has own characteristics, that is, specific operational costs. These costs include fuel consumption, fees that have to be paid to travel on roads and the value of the rent. The fees that have to be paid to travel on the roads may depend on the type of the car and on the specific roads chosen for the route. The value of the rent can also be associated with a cost per kilometer. Thus, without loss of generality, these costs can be considered as a function of each car on a value associated to the edges (i,j) of graph G. The operational cost of a given car k to traverse an edge (i,j) is denoted by c_{ij}^{k}.
2. A car rented in a given company can only be returned in a city where there is an agency of the same company. It is therefore not allowed to rent a car of a given company to travel on a certain segment of the route, if that car cannot be returned on the last city of the segment – there is not an agency of this company in the last city of the segment.
3. Whenever it is possible to rent a car in a city i and return it in city j, $i\neq j$, there is an extra for returning the car to its home city. The variable d_{ij}^{k} represents the expense to return car k to city j when it was rented in city i, $i\neq j$.
4. The tour begins and ends in the city where the first car is rented, the city that is the basis for the CaRS.
5. The return cost is null in case the tour is completed with a single car which is delivered in the same town it was rented. This case corresponds to the classic TSP considering the cost of other conditions associated only with the selected car.
6. Cars with the same characteristics rented in a single rental car company can be hired under different costs, depending on the city they where rented or on the contract negotiation. Therefore, without loss of generality, the designation of rent can be efficiently controlled by decisions related to cars, not considering the companies. The set $K=\{1,...,k\}$, $|K|=k$ is the set of different cars that can be in the solution.
7. The costs of returning the rented car may be strictly associated with the path between the city where the car is delivered and the city where the car was rented or these costs can be a result of independent calculation.

The objective of the proposed problem it to find the hamiltonian cycle that, starting on an initial vertex previously known, minimize the sum of total operating costs of cars in the tour. The total operating costs are composed of a parcel that unifies the rent and other expenses in a value associated to the edges, and a parcel associated to return the car to a city that is not its basis, calculated for each car and for each pair of cities origin/return in the cycle. The CaRS cycle may also be understood as obtained by the union of up to t Hamiltonian paths developed on up to t disjoint subsets of vertices of G. Each of the paths is accomplished with a different car or a car different from those used for the neighboring paths in the cycle. Therefore the cities that compose the cycle can be grouped into up to t different subsets of vertices of G that are covered by cars at least distinct from each other in the neighboring paths in the cycle.

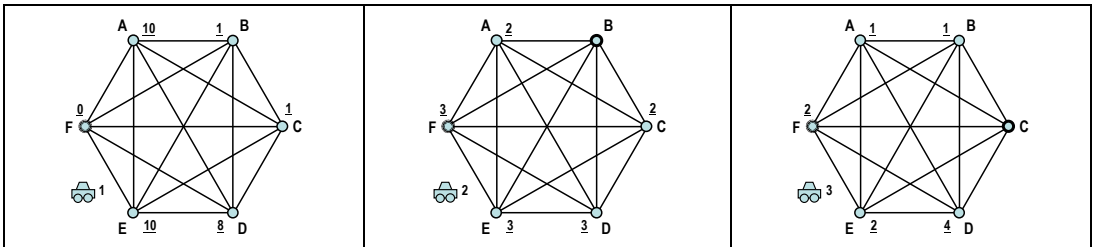
Figure 1 illustrates, in a complete graph with six vertices, a typical instance of CaRS. In the example there are three different rental cars. Figures 1(a), (b) and (c) show the accounting of the costs involved in the displacement of each type of car. Note that, unlike the classical traveling salesman cycle, the solution of CaRS depends on the city chosen to be the starting point of the

tour, the basis of the salesman. This fact is due to the rate of return is linked both to the starting city and the direction of devolution. In the example this city is represented by vertex F.



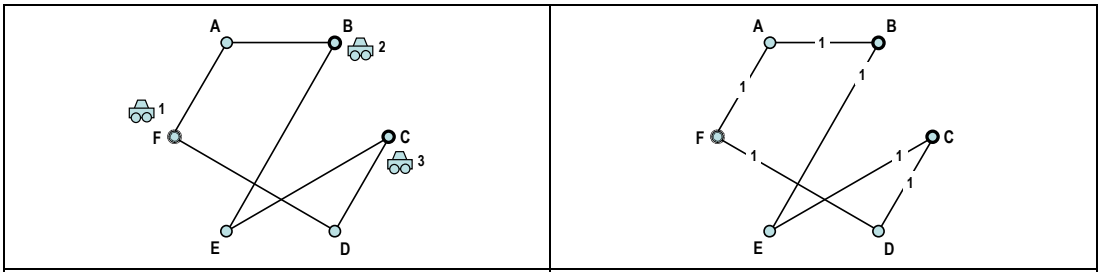
(a) Costs of car A | (b) Costs of car B | (c) Costs of car C

Figure 1: Costs associated to each rental car.

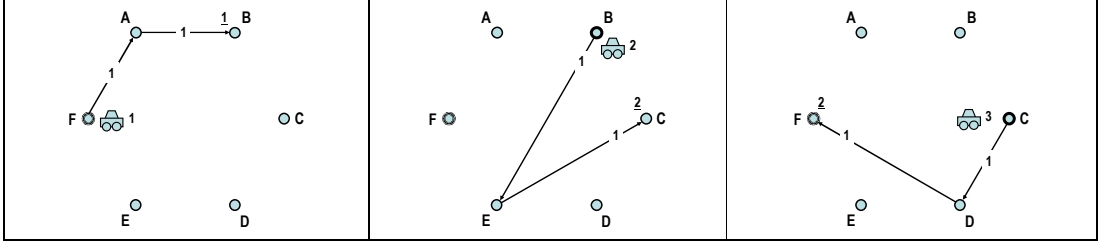


(a) Costs to return car 1 when rented on city F | (b) Costs to return car 2 when rented on city B | (b) Costs to return car 3 when rented on city C

Figure 2: Return costs



(a) Cars distributed in the route | (b) Route costs



(a) Costs of the route of car 1 | (b) Costs of the route of car 2 | (b) Costs of the route of car 3

Figure 3: Costs of the route of the exemplified problem.

Figure 2 shows, for the example in Figure 1, some of the costs of returning the cars to their bases. Delivery costs appear as underlined numbers next to vertices. Figure 2(a) shows the graph of return of car 1 when rented on vertex F. Figure 2(b) shows the graph of return of car 2 when rented on vertex B and Figure 2(c) the return of car 3 when rented on vertex C. In the general case return costs are known of all cars when rented in any of the cities.

A solution of the problem exemplified in Figures 1 and 2 is exhibited in Figure 3. This solution considers a case where all available cars are rented and no car is rented more than once. The cost of the cycle, according to the solution shown in Figure 3, corresponds to the cost of path F-A-B for car 1, added to the cost of path B-E-C for car 2, added to the cost of path C-D-F of car 3, in a total of 6 unities. To this value it is necessary to add the cost of returning the car to their bases. For car 1, the cost of the return from B to F is one unity. For car 2, the cost of the return from C to B is two unities and, for car 3, the cost to return to C when the car is delivered in F is two unities. Thus, the cost of the final solution is 11 unities.

The CaRS Problem has several variants in accordance with the real conditions of the problem. The problem can be classified according to the availability of cars, the alternatives of return, the existence of symmetry of the cost matrix and the existent links between the cities, etc.

2.4 The Difficulty of Solving CaRS

The problem basically consists in determining a Hamiltonian cycle in a graph G by composition of paths developed on the vertices of G. Let $T=\{1,...,t\}$ denote the set of indices of up to t subgraphs H_r of G, $r \in T$. Calling $V(H_r)$ the vertices of H_r, the subgraphs H_r of CaRS have the following properties.

$$\bigcup_{r=1}^{t} V(H_r) = N \tag{1}$$

$$\left| V(H_s) \bigcap V(H_r) \right| \leq 1 \qquad \forall r, s \tag{2}$$

Constraint (1) determines that the union of all paths visits all vertices of G. Constraints (2) implies that two different subgraphs never have more than one vertex in common, a condition to prevent formation of subcycles. Note that the constraints (1) and (2) are not sufficient to guarantee the cycle of the CaRS. It is also necessary that the t subgraphs considered three to three, four to four, and so on, until t-1 to t-1, do not have more than one vertex in common.

Once this problem deals with Hamiltonian paths, each path done by a car in one subgraph H_r visits all vertices of H_r. The path of subgraph H_r has to be assigned to a car different from the cars assigned to neighbor paths during the construction of an Hamiltonian cycle in G. The costs of the edges of each subgraph correspond to the operation costs of the car traversing H_r. Furthermore, when $t \geq 2$ the total cost considers the return cost of each car rented in city i and returned in city j, $i \neq j$. Hamiltonian cycle and Hamiltonian path problems are well known NP-complete problems (Garey & Johnson, 2003). Due to what was previously exposed, the difficulty of solving CaRS is at least the same as the TSP. Nevertheless, although some solutions of the TSP are also solutions of CaRS, the latter has a number of feasible solutions greater than the former and incorporates all the requirements of the TSP, like other several classes of vehicle routing problems which are known to be more difficult than the TSP (Ralphs *et al.*, 2003).

3. Metaheuristic Algorithms

This section presents two heuristics for the investigated problem. The first one is a Greedy Randomized Adaptive Search Procedure (GRASP) (Feo & Resende, 1995) combined with Variable Neighborhood Descent (VND) (Mladenović & Hansen, 1997) in the local search phase. The second heuristics is a Memetic Algorithm (Moscato, 1989).

3. 1. GRASP/VND

This algorithm has a pre-processing phase where *nCar* optimal TSP solutions are obtained with the Concorde TSP Solver (Applegate *et al.*, 2001), one for each available car, where *nCar* is

the number of cars available for the instance being considered. The constructive phase of GRASP/VND starts with a random selected car at the home city. Each iteration a path is built between two cities: a known origin and a destination city randomly chosen among the cities yet not considered by the algorithm. A Restricted Candidate List (RCL), with size α, is built with the cities that have the cheapest return rates for the car being considered in the current iteration. The destination city is selected at random with uniform probability from the RCL and a path between the origin and the destination city is built. Except for the last iteration, the path is built based on the tours built in the pre-processing phase. In the first iteration the path between the origin and destination cities is obtained in the optimal solution correspondent to the first selected car. The path of the *i-th* car, $1 < i < nCar$, is also obtained from the optimal solution that corresponds to the *i-th* car, but in this case a procedure to remove cities already considered in paths constructed in previous iterations may be necessary. Suppose that city *b*, between cities *a* and *c* in the path built in the *i-th* iteration, is already in a path built in iteration j, $j < i$. Then the procedure removes city *b* from the *i-th* path and includes a link between cities *a* and *c*. The initial starting city is the origin of the first iteration. The origin city of iteration i, $i > 1$, is the destination city of iteration i-1. The destination city of the last iteration is the initial starting city. In the last iteration, the nearest neighbor heuristic is used to build the path of the last considered car.

In the local search phase a VND metaheuristic was used to explore the search space of three neighborhood structures named *InvertSol*, *Insert&Saving* and 2-*Swap*. *InvertSol* is a simple low time consuming heuristics that reverses the sequences of cities of an input solution. With the reversal, though the same cars traverse the same sets of cities, the cost of rent and fees for returning the car to where they were rented change. The *Insert&Saving* procedure searches for a car insertion in a given solution that yields a decreasing of its cost. Let *s* be a solution in which there is at least one car that is not assigned to a path in *s*. Insert&Saving method randomly chooses a not assigned car and searches the best position to insert it in *s*. The procedure verifies the cost of the insertion of the new car in every point of *s*. If any of these insertions produces a solution with a cost lower than the cost of *s*, the new solution is set as the current solution in the local search. The procedure continues until all non-assigned cars have been considered for insertion. The third procedure, *2-Swap*, a neighboring solution *s*' of a solution *s* is generated by exchanging two cities within the path of one car of *s*. Procedures *Insert&Saving* and *2-Swap* use the first pivoting rule.

3. 2. Memetic Algorithm

Algorithm 1 shows the pseudo-code of the Memetic Algorithm (MA) developed for CaRS. The input parameters are: number of generations, population size, recombination rate (the number of individuals that reproduce in each generation), mutation rate and renewal rate of the population each generation.

Bi-dimensional arrays with *n* elements are used to represent the chromosomes. Figure 4 illustrates a chromosome for a CaRS instance with $n = 11$ and 5 available cars. The starting city (city 0) is not represented in the chromosome. The first row contains the cars assigned to each path. The last city a car visits is not assigned to the car, since it is returned on that city. Four of the five available cars are used in the solution illustrated in Figure 4. The tour begins at city 0 with car 2 which passes through cities 6, 4, 3, 10 and is delivered in city 7 where car 1 is rented. Car 1 proceeds to city 9 passing through city 1. Car 5 is rented in city 9, passes through cities 2 and 5 and is delivered in city 8 where the last car, 4, is rented. Car 4 is delivered in the starting city. The fitness of each chromosome is given by the inverse of the objective function, which means that the lower the value of the objective function the fittest chromosome is.

2	2	2	2	1	1	5	5	5	4
6	4	3	10	7	1	9	2	5	8

Figure 4: Chromosome

The initial population is generated with a version of the nearest neighbor heuristics adapted for CaRS in procedure *generateInitPop*( ) that receives the size of the population as input parameter. Let *nCar* be the number of available cars of a given instance. The algorithm randomly selects a car c and a destination city j for c, $j \neq 0$. Then a path between cities 0 and j is built with the nearest neighbor heuristic. City j is set as the new origin and a new car and a new destination city are randomly selected. The procedure continues until all cities are added to the tour or until there is only one car available. In the latter case, the last available car is assigned to a path built with the same heuristics between the previous destination city and the starting city, closing the tour. In step 4, each individual of the initial population is subjected to the same VND procedure used on GRASP/VND.

Algorithm 1 – Main Procedure of Memetic for CaRS

```
1   main(nameInstance,sizePop,nOffspring,txCros,txMuta,txRenw)
2     instanceRead(nameInstance)
3     Pop[] ← generateInitPop(sizePop)
4     VNDlocalSearchPhase(Pop)
5     for i of 1 to nOffspring do
6       for j of 1 to sizePop*txCros do
7         dad,mom ← parentsSelection()
8         citiesCrossover(dad,mom)
9         son1, son2 ← carsMutations(son1,son2,txMuta)
10        VNDlocalSearchPhase(son1,son2)
11        if son1,son2 < Pop[dad],Pop[Mom]
12           Pop[dad] ← son1, Pop[mom] ← son2
13      generateNewIndividuals(sizePop*txRenw)
14  return(Pop[0])
```

Parents for recombination are selected with the roulette wheel method. A multi-point recombination operation adapted for CaRS is used to generate two children. The recombination operator is illustrated in Figure 5, considering an instance with n = 11 and 3 cars. Two parent chromosomes, A and B, generate offspring C and D. In Figure 5 a 2-point operator is used. The first and third parts of chromosomes A and B are inherited by chromosomes C and D, respectively. A restoration procedure may be necessary to restore feasibility regarding the routes and car assignments. For example, after recombination the route of chromosome C is [3 1 8 10 1 9 4 5 10 6] which is not feasible since cities 1 and 10 appear twice each and cities 2 and 7 are missing. Thus the route of chromosome C is replaced by [3 1 8 10 * 9 4 5 * 6] with asterisks replacing the second time cities 1 and 10 appears. Each asterisk is then replaced at random by cities 2 and 7. The row corresponding to the car assignment for chromosome C after recombination is [1 1 1 2 3 3 2 2 2 2 3] what for the problem considered in this paper is not feasible, since each car can be assigned only to one path. Thus, the car assignment of chromosome C is replaced by [1 1 1 2 3 3 * * * * 3] and each asterisk is replaced by car 3. A similar analysis is done for chromosome D.

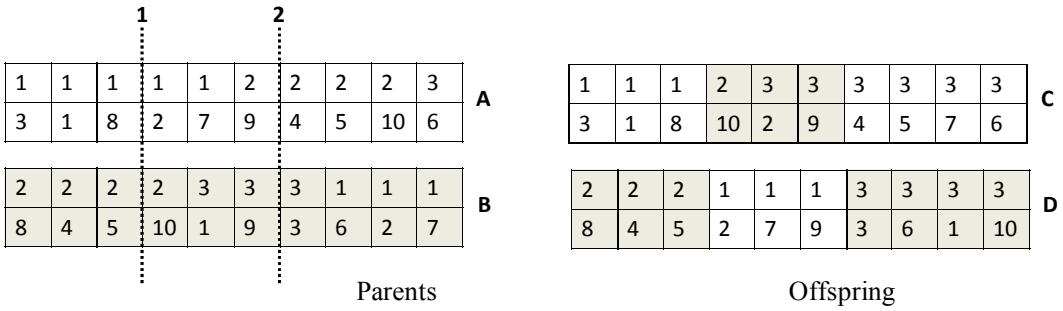


Figure 5: Recombination operator

The solutions resulting from the recombination are subjected to mutation. The mutation operator changes the vehicles used in pre-established parts of the route of each solution. The size of the segments is determined as the mutation rate parameter.

After crossover and mutation, the offspring is subjected to the VND method. The resulting solutions are compared with their parents and the best two individuals survive. Finally, part of the current population is replaced by new solutions generated with the constructive method used to create the initial population. The number of new individuals created with that procedure is given by the renewal rate and the individuals chosen to be replaced are those with the worst values of fitness. This renewal process promotes diversification and prevents premature convergence.

4. Computational Experiments

This section presents the instances used in the experiments, the parameter settings and the computational results.

Since this is a new problem, a library of instances, named CaRSLib, was created to test the proposed algorithms. The instances of the computational experiment reported in this paper refer to the specific CaRS problem that has the following properties: *total* (all cars can be rented in all cities), *unrestricted* (all cars can be delivered in any city), *without repetition* (same type of car can only be rented once on the tour), *free* (the costs of returning the cars do not have correlation with the topology of the problem), *symmetric* (symmetric distances between the cities) and *complete* (the graph of the problem is complete). The set consists of Euclidean and non-Euclidean instances. For each set, three groups of instances were created, the first based on real maps, the second was formed with randomly generated data and the third one based on the TSPLIB instances. The dataset, the description of each group of instances and file formats are available at http://www.dimap.ufrn.br/lae/en/projects/CaRS.php.

A preliminary computational experiment was done to tune two parameters of GRASP/VND: α and maximum number of GRASP/VND iterations. The experiments were done on an Intel Core Duo 1.67GHz, 2GB of RAM running Linux with C++. The experiments were performed on a subset of CaRSLib with 20 instances, with number of cities ranging from 14 to 300 and 2 to 5 vehicles. The parameter α was set to 0.25. The maximum number of re-starts was set to 300. An additional stopping criterion fixed 90 re-starts without improvement of the best current solution.

The same set of instances and computational platform was used to tune the parameters of the Memetic Algorithm. Twenty independent executions were performed for each instance. Two groups of tests were done: first the algorithm was executed without the VND method, and then the VND was included. The parameter settings are: maximum of 20 generations, population with 30 chromosomes, recombination rate of 60%, mutation rate of 40% and renewal rate of 15%. The stopping criterion was 30% of the maximum number of generations without improvement of the best current solution. A number between 1 and 4 was randomly selected to be the number of points each time the multi-point recombination operator was applied.

GRASP/VND and MA were executed on a PC Intel Core Duo 1.67GHz,2GB of RAM running Linux. The experiments reported in this paper were performed with 40 CaRS instances, 20 of them from the Euclidean group and divided into 10 instances from group 1 (manual), 5 from group 2 (random) and 5 from group 3 (TSPLIB). The remaining instances are non-Euclidean and are divided in 3 groups as well. During the tests, 30 independent executions of each algorithm were performed for each instance. Two groups of experiments were performed with fixed processing times. In the first group the average processing time spent by GRASP/VND to find its best solution for each instance was given to both algorithms. In the second group the average processing time of the MA for each instance was fixed for both algorithms. The results are, respectively, reported in Tables 1 and 2. These tables show the name of the instance (*Name*), the number of cities (*City*), the number of available cars (*Car*), the best solution found (*#Best*), the processing time in seconds (*T*), the worse (*Worse*), the average (*Avg*) and the best solution (*Best*), de standard deviation (*Dev*) and the number of times (*Freq*) the best reported solution was found by each algorithm.

Table 1 - Comparison of results with time determined by GRASP

INSTANCES				GRASP/VND						MEMETIC				
Name	City	Car	#Best	T(s)	Worse	Avg	Dev	Best	Freq	Worse	Avg	Dev	Best	Freq
BrasilRJ14e	14	2	294	1	297	297	0	297	30	297	294	0	294	29
BrasilRN16e	16	2	375	1	375	375	0	375	30	387	375	2	375	29
BrasilPR25e	25	3	510	2	512	510	0	510	29	522	515	5	510	16
BrasilAM26e	26	3	467	3	498	495	1	495	27	495	485	9	469	1
BrasilMG30e	30	4	563	5	604	603	2	595	1	604	599	7	575	1
BrasilSP32e	32	4	611	8	637	633	5	626	9	630	621	4	611	2
BrasilRS32e	32	4	510	8	560	537	9	529	11	539	522	6	510	1
BrasilCO40e	40	5	779	18	812	807	2	805	1	829	822	10	779	1
BrasilNO45e	45	5	886	23	1008	1008	0	1008	30	1008	978	34	886	1
BrasilNE50e	50	5	822	43	973	963	5	940	1	964	954	27	822	1
Betim100e	100	3	1401	78	1741	1723	8	1708	4	1739	1692	89	1410	1
Vitoria100e	100	5	1598	155	2006	1802	75	1642	1	1935	1891	87	1598	1
PortoVelho200e	200	3	2827	466	3251	3142	29	3041	1	3254	3149	129	2827	1
Cuiaba200e	200	3	3052	686	3670	3379	88	3212	1	3461	3414	80	3217	1
Belem300e	300	4	4031	1804	5091	4635	121	4563	6	4443	4425	76	4031	1
berlin52eA	52	3	8948	20	9108	9020	35	8991	7	9132	9081	72	8948	4
eil76eB	76	4	1940	87	2311	2228	42	2158	1	2157	2077	43	1940	1
rat99eB	99	5	3339	194	3511	3439	42	3351	1	3696	3513	75	3365	1
rd100eB	100	4	9951	103	10278	10107	81	9951	1	10555	10364	172	10054	1
st70eB	70	4	2037	77	2308	2201	44	2085	1	2236	2151	46	2042	1
BrasilRJ14n	14	2	167	1	171	171	0	171	30	168	167	0	167	4
BrasilRN16n	16	2	190	1	203	203	0	203	30	200	194	2	192	3
BrasilPR25n	25	3	235	5	334	311	9	305	20	270	255	7	239	1
BrasilAM26n	26	3	204	5	266	242	6	239	21	221	213	4	206	2
BrasilMG30n	30	4	279	11	403	375	11	352	1	357	330	14	298	1
BrasilSP32n	32	4	285	12	366	336	16	298	1	306	295	5	285	1
BrasilRS32n	32	4	297	15	406	372	15	344	1	360	337	14	297	1
BrasilCO40n	40	5	655	39	944	826	42	755	1	780	718	37	655	1
BrasilNO45n	45	5	664	55	977	889	42	770	1	818	753	39	664	1
BrasilNE50n	50	5	707	81	1149	1044	60	874	1	906	844	43	761	1
Londrina100n	100	3	1450	192	1928	1783	80	1629	3	1672	1564	51	1450	1
Osasco100n	100	4	1150	191	2118	2000	60	1910	1	1681	1443	109	1265	1
Aracaju200n	200	3	2467	903	4027	3686	212	3223	1	3096	2802	136	2588	1
Teresina200n	200	5	2192	1407	4026	3793	144	3261	1	2788	2480	143	2192	1
Curitiba300n	300	5	3676	3388	6545	6125	202	5680	1	4681	4081	202	3749	1
berlin52nA	52	3	1480	41	1926	1777	82	1661	6	1732	1640	51	1543	1
ch130n	130	5	2487	478	5139	4706	307	3855	1	3672	2940	245	2487	1
d198n	198	4	4807	1330	7874	7138	333	6529	1	5886	5332	269	4807	1
kroB150n	150	3	3824	464	6525	5368	434	4414	3	4643	4312	194	3824	1
rd100nB	100	4	1890	205	3243	2953	169	2623	1	2479	2274	118	2083	1

Table 2 - Comparison of results with time determined by Memetic

INSTANCES				MEMETIC						GRASP/VND				
Name	City	Car	#Best	T(s)	Worse	Avg	Dev	Best	Freq	Worse	Avg	Dev	Best	Freq
BrasilRJ14e	14	2	294	1	297	294	1	294	25	297	297	0	297	30
BrasilRN16e	16	2	375	1	394	376	4	375	27	375	375	0	375	30
BrasilPR25e	25	3	510	2	522	515	5	510	17	510	510	0	510	30
BrasilAM26e	26	3	467	4	495	481	10	467	3	495	495	0	495	30
BrasilMG30e	30	4	563	6	604	596	10	563	1	604	602	2	595	1
BrasilSP32e	32	4	611	8	637	624	5	615	1	637	632	4	626	6
BrasilRS32e	32	4	510	8	541	523	7	512	2	557	536	9	529	14
BrasilCO40e	40	5	779	17	832	824	7	801	1	814	806	2	806	22
BrasilNO45e	45	5	886	25	1008	993	27	897	1	1008	1008	0	1008	30
BrasilNE50e	50	5	822	31	964	963	2	953	1	981	962	11	908	1
Betim100e	100	3	1401	128	1739	1642	110	1401	1	1741	1720	7	1708	6
Vitoria100e	100	5	1598	98	1935	1922	29	1814	1	2033	1859	77	1676	1
PortoVelho200e	200	3	2827	766	3254	3134	117	2871	1	3138	3128	22	3041	1
Cuiaba200e	200	4	3052	701	3461	3415	96	3052	1	3594	3365	56	3334	2
Belem300e	300	4	4031	2016	4443	4434	30	4282	1	5099	4621	125	4563	11
berlin52eA	52	3	8948	27	9132	9094	65	8948	4	9108	9013	24	8991	9
eil76eB	76	4	1940	61	2158	2069	43	1986	1	2329	2226	56	2129	1
rat99eB	99	5	3339	128	3715	3525	71	3339	1	3574	3468	54	3348	1
rd100eB	100	4	9951	161	10555	10385	209	9994	1	10168	10055	54	9951	1
st70eB	70	4	2037	54	2299	2158	67	2037	1	2260	2212	31	2137	1
BrasilRJ14n	14	2	167	1	168	167	0	167	2	171	171	0	171	30
BrasilRN16n	16	2	190	1	202	195	3	190	1	203	203	0	203	30
BrasilPR25n	25	3	235	4	282	256	10	235	1	339	316	12	305	15
BrasilAM26n	26	3	204	5	228	212	4	204	1	255	242	5	239	17
BrasilMG30n	30	4	279	8	354	328	15	279	1	403	378	14	352	3
BrasilSP32n	32	4	285	13	313	296	7	287	2	362	331	14	300	3
BrasilRS32n	32	4	297	9	379	340	16	304	1	411	378	18	344	1
BrasilCO40n	40	5	655	20	809	743	33	668	1	918	839	41	710	1
BrasilNO45n	45	5	664	32	834	764	39	667	1	1001	919	43	814	1
BrasilNE50n	50	5	707	46	1012	861	61	707	1	1143	1068	57	924	1
Londrina100n	100	3	1450	146	1666	1592	50	1471	1	1964	1767	85	1629	1
Osasco100n	100	4	1150	125	1755	1442	139	1150	1	2151	2046	80	1817	1
Aracaju200n	200	3	2467	922	2975	2744	135	2467	1	3974	3594	236	3106	1
Teresina200n	200	5	2192	836	2954	2551	182	2233	1	4106	3866	126	3611	1
Curitiba300n	300	5	3676	2384	4507	4076	216	3676	1	6291	6050	160	5647	1
berlin52nA	52	3	1480	38	1866	1642	78	1480	1	1889	1748	63	1661	4
ch130n	130	5	2487	237	3575	3020	228	2493	1	5408	4863	345	3813	1
d198n	198	4	4807	823	6193	5449	318	4887	1	7904	7407	378	6250	1
kroB150n	150	3	3824	418	4618	4259	208	3845	1	5859	5313	272	4871	2
rd100nB	100	4	1890	140	2569	2271	143	1890	1	3360	2962	180	2685	2

The results shown in both tables are very similar regardless the processing time fixed in each group of experiments. The MA presents an overall better performance than the GRASP/VND. The former presents 32 and 31 best average results against 9 best results of the latter in the two groups of experiments. A similar behavior can be noticed for the best solutions found by each algorithm.

5. Conclusions

This paper introduced the Car Renter Salesman Problem (CaRS), a new variant of the classic Traveling Salesman Problem. Two metaheuristic methods were compared to establish the initial limits for a set of 40 CaRS instances: a GRASP hybridized with VND and a Memetic Algorithm. The computational experiments show that the Memetic Algorithm outperforms the GRASP/VND algorithm both concerning quality of solution and processing time.

The introduced problem has a number of variants opening a wide area of investigation on exact and heuristic methods to solve it.

References

Applegate, D.L.; Bixby, R.; Chvátal, V.; Cook, W. (1994) Finding cuts in the TSP: a preliminary report distributed at The Mathematical Programming Symposium, Ann Arbor, Michigan.

Applegate, D.L.; Bixby, R.; Chvatal, V.; Cook, W. (2001) TSP cuts which do not conform to the template paradigm, *Computational Combinatorial Optimization*, M. Junger and D. Naddef (editors), 261-303.

Applegate, D.L. Bixby, R.; Chvátal, V.; Cook, W. (2006) *The Traveling Salesman Problem: A Computational Study*, Princeton University Press.

Avis (2009) Avis Europe plc - Financial and Strategic Analysis Review, Global Markets Direct, 17, available at http://www.researchandmarkets.com/reports/690999.

Avis (2010) Avis Budget Rental Car Funding. Available at http://www.dbrs.com/research/232065/avis-budget-rental-car-funding-aesop-llc-series-2010-2-3/rating-report-series-2010-2-3.pdf

Car Rental (2008) Web Page, available at http://thrifty4.com/article.cfm/id/284920

Car (2008) Car Rental Business - Global Strategic Business Report Global Industry Analysts, Inc., 278. Available at http://www.researchandmarkets.com/reports/338373/.

Conrad, C.; Perlut, A. (2006) Enterprise Rent-A-Car Hits New Billion-Dollar Revenue Mark for 3rd Consecutive Year, Enterprise rent-a-car, available at: http//www.enterpriseholdings.com/NewsReleases/Enterprise_FYO6_Sept06.pdf

Crowder, H.; Padberg, M. W. (1980) Solving large scale symmetric traveling salesman problems to optimality, *Management Science* 26, 495-509.

Edelstein, M.; Melnyk, M. (1977) The pool control system, *Interfaces* 8(1), 21-36.

Enterprise (2009) Enterprise Rent-A-Car Company - Strategic Analysis Review, Global Markets Direct, 9. Available at http://www.researchandmarkets.com/reports/690685/.

Feo, T.A.; Resende, M.G.C. (1995) Greedy randomized adaptive search procedures, *Journal of Global Optimization* 6, 109-133.

Garey, M. R.; Johnson, D. S. (2003) *Computers and Intractability. A Guide to the Theory of NP-completeness*. W.H. Freeman and Company, New York.

Gutin, G.; Punnen, A. P. (2004). The Traveling Salesman Problem and Its Variations, Series: *Combinatorial Optimization* 12, Springer.

Hertz (2009) The Hertz Corporation - Strategic Analysis Review, Global Markets Direct, 12. Available at http://www.researchandmarkets.com/reports/690555/.

Hertz, A.; Schindl, D.; Zufferey, N. (2009) A solution method for a car fleet management problem with maintenance constraints, *Journal of Heuristics* 5, 425–450.

Krarup, J. (1975) The peripatetic salesman and some related unsolved problems, in *Combinatorial Programming Methods and Applications*, Reidel, Dordrecht, 173–178.

Lia, Z.; Taob, F. (2010) On determining optimal fleet size and vehicle transfer policy for a car rental company, *Computers & Operations Research* 37, 341-350.

Mladenovic, N.; Hansen, P. (1997) Variable neighborhood search. *Computers & Operations Research* 24, 1097-1100.

Montemanni, R.; Barta, J.; Mastrolilli, M.; Gambardella, L. M. (2007) The robust traveling salesman problem with interval data, *Transportation Science* 41(3), 366–381.

Moscato, P. (1998) *On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithm*. Caltech Concurrent Computation Program. California Institute of Technology, USA.

Pachon. J. E.; Iakovou, B.; Ip, C.; Aboudi, R. (2003) A synthesis of tactical fleet planning models for the car rental industry, *IIE Transactions* 35(9), 907-916.

Ralphs, T. K.; Kopman, L.; Pulleyblank, W. R.; Trotter, L. E. (2003) On the capacitated vehicle routing problem, *Mathematical Programming*, Ser. B 94, 343–359.

Reinelt, G. (1995) TSPLIB95. Available at http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95.

Russel, R. (1977) An effective heuristic for the m-tour traveling salesman problem with some side conditions, *Operations Research* 25, 517-524.

Wikipedia (2010) Enterprise Rent-a-Car page. Available at http://en.wikipedia.org/wiki/Enterprise_Rent-a-Car.

Xiong, Y.; Golden, B.; Wasil, E. (2007) The Colorful Traveling Salesman Problem, Extending the Horizons: Advances in Computing, Optimization, and Decision Technologies, Operations Research/Computer Science Interfaces Series 37, Springer US, 115-123.

Yang, Y.; Jin, W.; Hao, X. (2008) Car Rental Logistics Problem: A Review of Literature, *IEEE International Conference on Service Operations and Logistics, and Informatics. IEEE/SOLI 2008.* 2, 2815 – 2819.