2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Reducing Traveled Distance in the Vehicle Routing Problem with Time Windows using a Multi-Start Simulated Annealing

Humberto César Brandão de Oliveira, Germano Crispim Vasconcelos and Guilherme Bastos Alvarenga

Abstract—**Vehicle Routing Problems have been extensively analyzed to reduce transportation costs of people and goods. More particularly, the Vehicle Routing Problem with Time Windows (VRPTW) imposes the period of time of customer availability as a constraint, a very common characteristic in real world picking up and delivery problems. Using minimization of the total distance as the main objective to be fulfilled, this work implements an efficient hybrid system which associates a non-monotonic Simulated Annealing technique to a Hill Climbing Strategy with Random Restart. The algorithm performance is evaluated by comparing the results achieved with the proposed algorithm with the best published works found in the literature of the 56 Solomon instances.**

I. Introduction

COSTS with goods transportation are getting a special attention in the last decades, where logistic expenses minimization is a big concern and sometimes a survival issue for many companies in the market. Many studies found in the literature of Vehicle Routing Problem (VRP) have attempted to contribute to practical advances in this field [22][24]. Since in the real world this problem has to consider many constraints and particularities, some parameters have been considered in order to study problems closer to the real case situations. One of these constraints is the consideration of the capacity of the vehicle and the time window in which the customers must be reached. This class of problems is known as the Vehicle Routing Problem with Time Windows (VRPTW) and, at moment, it is the most popular class of the vehicle routing problem studied [19].

According to [1], the costs related to people and goods transportation are very significant and are growing rapidly, motivated by the continuing increase of business complexity experienced today. Studies suggest that from 10% to 15% of the final value of the traded goods correspond to its transportation cost [7]. A parcel of these costs can be reduced whit the treatment of diverse vehicle routing problems, where the VRPTW is a particular important case.

This work proposes an efficient hybrid system for the VRPTW and compares its performance with the best published results found so far in the literature. In Section II, the VRPTW is described. Section IV presents the new algorithm, describing the elements and principles considered for solving the problem. Section V comments the test bed used for performance measure and comparison; and Section VI describes the results achieved. Finally, Section VII analyses the results of this research.

H. C. B. Oliveira is with the Center for Informatics, Federal University of Pernambuco - Brazil (e-mail: hcbo @ cin.ufpe.br).

G. C. Vasconcelos is with the Center for Informatics, Federal University of Pernambuco - Brazil (e-mail: gcv @ cin.ufpe.br).

G. B. Alvarenga is with the Department of Computer Science, Federal University of Lavras - Brazil (e-mail: guilherme @ dcc.ufla.br).

II. Vehicle Routing Problem with Time Windows

The Vehicle Routing Problem with Time Windows (VRPTW) is a combinatorial optimization problem. This is a particular case of the well know Vehicle Routing Problem (VRP), introduced by [5]. In the VRP, the vehicle fleet must to visit and deliver a service to a set of customers. Every vehicle starts and finishes at a unique depot. For each pair of customers, or customer and the depot, there is an associated cost. This cost denotes how expensive it is for a vehicle to move from one costumer to another, with the constraint that each customer must be visited exactly once. Additionally, each customer demands a specific number of goods (denoted as the weight of the load). For each vehicle in a fleet, there is an upper limit of load capacity supported. In the basic case, considered here, all the vehicles are of the same type and have the same capacity. Then, basically, the objective of the VRP is to find a set of customers attended for each vehicle of the fleet in order to minimize the transportation costs [9].

With an additional complexity to the VRP, in the VRPTW each costumer has an associated time window that determines an interval within which a vehicle has to begin and finish the service to a specific customer. In the VRP as well as in the VRPTW several types of optimization objectives have been investigated in the literature. In particular, the total distance traveled is one of the most typical cost measure to be minimized by a given algorithm. Another parameter taken as target is to find the minimum set of possible routes to solve the problem, and to minimize the distance as a second objective. Yet some other authors also consider the time minimization for attending all the costumers as the objective to be fulfilled.

With the aim of minimizing the total traveled distance, such as in the work developed here, De Backer et al. investigated iterative improvement techniques within a Constraint Programming (CP) framework [3]. The improvement techniques are coupled to Tabu Search (TS) and Guided Local Search (GLS) to avoid the search of being trapped into local minima. The CP system is used only as a background operator to check the validity of the solutions found and to speed up legality checks of improvement

0-7803-9490-9/06/$20.00/©2006 IEEE

procedures. Riise and Stølevik [21] studied GLS and Fast Local Search (FLS) combined with simple move operators that relocate single tasks. Kilby et al. [20] introduced a deterministic GLS that use local search operators (2-opt, relocate, exchange and 2-opt*) with a so-called best-acceptance strategy (for details, see [23]). Alvarenga [2] studied the use of a Set Partitioning (SP) formulation after generating several solutions through a given genetic algorithm (GA). The routes of the generated solutions by the GA are combined by the exact method of Set Partitioning, finding the best combination of the routes without violating the constraints of the VRPTW.

In the present paper, a different approach based on the combination of simulated annealing and a hill climbing strategy with random restart (multi-start) is proposed.

III. Complexity of the VRPTW

As defined in [13], an instance of an optimization problem is a pair *(F,c)* where *F* is any set, the domain of feasible points; c is a cost function, i.e. a mapping

$$c : F \rightarrow R^1 \qquad (1)$$

The problem is to find an $f \in F$ for which

$$c(y) \leq c(f) \quad \forall \quad f \in F \qquad (2)$$

Such a point *y* is called a global optimal solution to a given instance or simply an optimal solution.

As the VRPTW is a problem known as NP-hard [10], only solutions of reduced order instances can be found using exact algorithms [4]. The impossibility of guaranteeing optimal solutions ($f \in F$) occurs because the set of functions *F* is huge, given the non-determinism in the search of all possible solutions to the problem, preventing the sweeping of every solution in a polynomial time. To overcome this problem, heuristics and meta-heuristics are frequently employed to find sub-optimal solutions to the VRPTW which are both performance effective and feasible to be determined in non-polynomial time [2].

IV. A Hybrid System (HS) to the VRPTW

This work aims at constructing a hybrid system that finds a solution $f \in F$ such that *y* is a good approximation to the optimal solution *y*, based on a combination of simulated annealing with a hill-climbing strategy.

A set of techniques were considered with the principle of generating a solution that could provide good results in the diverse set of possible situations of the VRPTW. A strategy based on simulated annealing and hill climbing took place inspired on the capability of simulated annealing to both evolve solutions to a given problem and escape from local minima and on the capacity of hill climbing to refine initially defined solutions. To complete the method, a technique called 'random restart' [14] of the system is applied in order to cope with the idea of producing solutions to varied configurations of the VRPTW, returning the best solution from the executed restarts. Such strategy performs multiple system restarts with the association of simulated annealing and hill climbing and finds better results by diminishing the variance between the different executions of the system, and consequently enhancing the robustness of the method. Such strategy performs multiple system restarts with the association of simulated annealing and hill climbing and finds better results by diminishing the variance between the different executions of the system, and consequently enhancing the robustness of the method. The idea behind the operation of the hybrid system can be viewed in the Fig. 1, and the characteristics of this combined approach are described in details in the next sections.

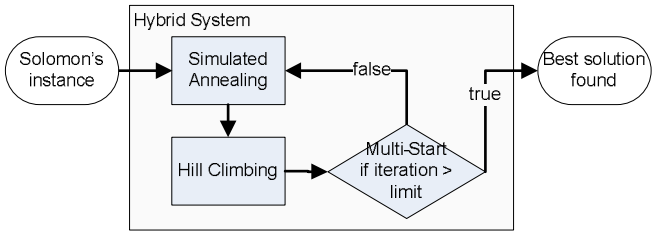

Fig. 1. Illustration of the Hybrid System

A. Simulated Annealing (SA) to Solve the VRPTW

Simulated Annealing is a probabilistic meta-heuristic algorithm, proposed originally in [8] (being a local search method), that accepts search movements that temporarily produces degradations in a current solution found to a problem as a way to escape from local minima.

This meta-heuristic is based on a natural method which uses an analogy to the thermodynamics simulating the cooling of a set of heated atoms, in a operation known as Annealing [8] (the term and operation of annealing are widely use in metallurgy).

In its formal description, the simulated annealing begins its search from a random initial solution. The iteration loops that characterize the main procedure randomly generate, in each iteration, only one neighbor *s´* of the current solution *s*. The variation Δ for the value of the objective function *f(x)* is tested for each neighbor generation. To test this variation, the following calculation is conducted:

$$\Delta = f(s') - f(s) \qquad (3)$$

If the value of Δ is less than *0* (zero), then the new solution *s´* is automatically accepted to replace *s*. Otherwise, accepting the new solution *s´* will depend on the probability established by the Metropolis Criteria, which is given by:

$$e^{-\Delta/T} \qquad (4)$$

where *T* is a temperature parameter, a key variable for operation of the method. The Metropolis Criteria accepts, with higher probability, solutions which have lower values

of Δ. Higher values of Δ will have lower chances if compared to lower values of Δ. The higher the temperature, the higher is the probability of accepting the solution s' as the new solution, explaining the algorithm analogy to the solid cooling.

1) Starting Point for the Simulated Annealing

For constructing an initial solution for the SA algorithm, this work used the algorithm known as *Push-Forward Insertion Heuristic* (PFIH), introduced in [15]. As cited in [9], the PFIH has an efficient constructive strategy for calculating the cost of a new customer in a route. This cost is computed according to its geographic position, the end of its time window and the angle between it and the central depot. For a better understanding consider the Fig. 2 as a current solution before the insertion of the customer C5.

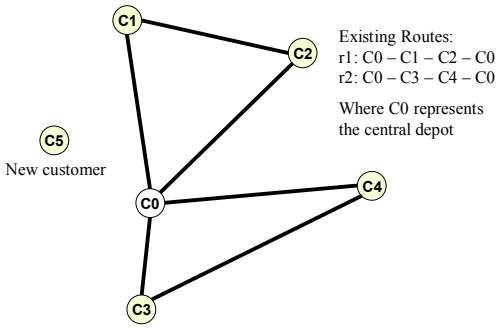

Fig. 2. Current incomplete solution before the insertion of a new customer in a route

The quality of the possible solutions is first checked with the insertion of the customer (C5) in each possible edge of the graph representing the current solution for the VRPTW. Then, the edge in which the insertion of the customer represents the lowest cost is chosen (with respect to the total distance) for the inclusion of the new customer in the route.

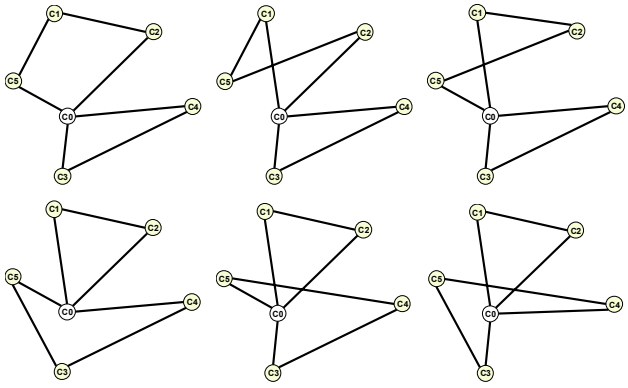

Fig. 3. Feasible solutions found by the PFIH algorithm for the introduction of a new customer

Besides the computation of the costs, the insertion of the customer is also examined to guarantee that it does not violate any of the restrictions involved in the VRPTW (see Section 2). If any of the existing (from the smallest to the biggest cost) route solutions (shown in Fig. 3) does not violate any constraint of capacity, load of the vehicle or attendance time to the customers, then this route becomes the definite solution for inclusion of the new costumer (Fig. 4). Otherwise, the current routes are discarded and a new route is created for representing the new costumer.

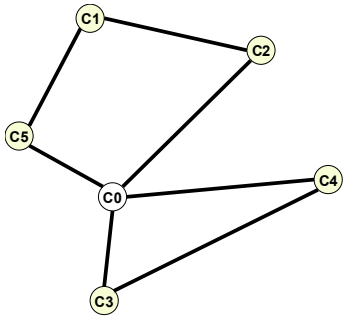

Fig. 4. Solution selected as the best option when inserting the customer C5

The order in which the customers are inserted into the VRPTW solution by the PFIH algorithm directly defines the quality of the final solution produced by the method. Keeping this in mind, Solomon developed in his work [15] a heuristic to determine the order in which the customers should be considered into the solution, according to the cost Equation:

$$C_i = -\alpha d_{oi} + \beta l_i + \gamma \left[\left(\frac{p_i}{360} \right) d_{0i} \right] \qquad (5)$$

Where:

$\alpha = 0.7$; $\beta = 0.1$; $\gamma = 0.2$;

d_{0i} = the distance between the central depot and customer i;

b_i = the upper limit of the time window for arrival of customer i;

p_i = polar coordinate angle of the customer i, with respect to the central depot;

The constant values for α, β e γ were defined empirically in [15].

From the first chosen customer, the remaining customers are tested one by one with respect to each possible route solution for construction, according to the cost Equation 6. The position and the customer that resulted in the lowest increase in the total traveled distance, without violating the time window capacity, are chosen. After there are no more customers to insert in the route under construction, this particular route is closed and the same process is restarted again with a new empty route, being the first customer the one with the lowest cost according to Equation 6, among those customers yet to be routed.

Aiming at introducing flexibility for starting the SA algorithm from different positions in the search space, this work introduced a variation on the original cost formula, where, instead of been treated as constants in the original algorithm, the α, β and γ elements are turned into PFIH parameters. The new values change at each execution being captured by a normal distribution $N(\mu,\sigma)$, with an average in

the points ($\alpha\mu = 0.7$; $\beta\mu = 0.1$; $\gamma\mu = 0.2$), suggested as optimal by Solomon, and with a deviation by the unit ($\sigma = 1$). With this variation of the heuristic orders, the customers being inserted by the PFIH algorithm maintain a good arrangement because the averages are centered on the optimal values obtained empirically and the small variations cause perturbations that create different initial solutions for the SA method. This modification on the PFIH algorithm proposed here was inspired on the work of [11] that implements the change of the heuristic through the removal of the values of α, β and γ from a uniform distribution changing from 0 to 1.

2) Neighborhood Operators Applied to the Simulated Annealing

Given a feasible point $f \in F$ in a particular problem, it is useful in many situations to define a set $N(f)$ of points that are 'close' in the same sense to the point f [13], where F represents the set of any solution that satisfies the problem. For example, if $F = R^n$, then the set of points within a fixed Euclidean distance provides a natural neighborhood solution for F [13].

As this work focus its solutions on ordered lists without repetitions (routes with its respective customers), it is guaranteed, according to [6], that for this type of representation four basic permutation operators can describe a generic way to capture the neighborhood of a solution f. These operators perform permutations between the elements in order to capture the neighborhood of a solution f and were originally implemented as mutation operators in evolutionary algorithms [6]. These have been adapted in this research to be employed in the SA for the neighborhood definition.

The first neighborhood operator is the simplest and is called Swap Mutation (see the work of [6]). Its definition is described to find a neighborhood of a solution for the well known Traveling Salesman Problem (TSP) and is called "2-change" in [13]. The *swap* (*f*) operator can be described as:

swap(f) = {*g* : $g \in F$ e *g* can be obtained from *f* swapping 2 customers (*c1, c2*) of any routes (*r1, r2*)}.

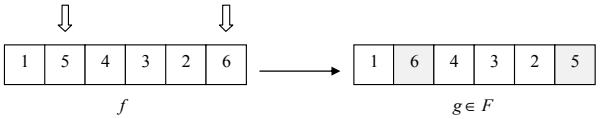

Fig. 5. Application of the swap neighborhood operator in the same route of solution *f*

The second neighborhood operator, also based on random changes, is called Insert Mutation and is formally defined as:

insert(f) = {*g* : $g \in F$ and *g* may be obtained by removing one customer from any route of *f* and inserting it again in any position of any route of *f*}.

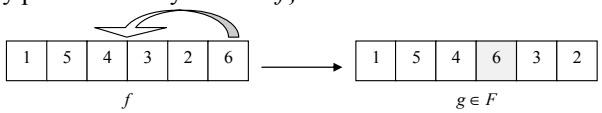

Fig. 6. Example showing the application of the Insert neighborhood operator in a route of solution *f*

The third operator is the Scramble Mutation, which is another random operator defined as:

scramble(f) = {*g* : $g \in F$ and *g* may be obtained choosing any continuous sequence *q* of customers in a route *r* chosen randomly from *f* and mixing the customers of *q* in order to create a sequence *q'* which will substitute *q* in *r* }

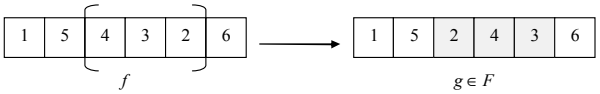

Fig. 7. Example of application of the neighborhood operator Scramble on a route *r* of the solution *f*

The fourth operator, totally random, is based on the customers' inversion and may be explained by the next definition:

inversion(f) = {*g* : $g \in F$ e *g* may be obtained by choosing sequence *s* of customers in a route *r,* randomly chosen from *f,* and after inverting them systematically for the generation of a new sequence *s'* that will replace *s* in *r*}.

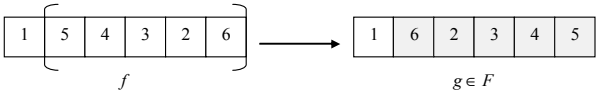

Fig. 8. Example of the application of the neighborhood operator inversion in the same route *r* of solution *f*

A fifth operator, based on the work of [12], was also employed to define a neighborhood solution. This is also a mutation operator for evolutionary algorithms that uses specific information on the problem (heuristics) to navigate in the solution space constructed by the instantiated VRPTW.

Firstly, *m* customers are withdrawn from each route of solution *f*. The number of withdrawn customers varies for each route *r* and is chosen by selecting a value from an uniform distribution that varies from 0 to the number of customers present in *r*.

After selecting the customers withdrawn from *f* (creating an incomplete solution *h*), all the selected customers are inserted back into *h* through the *PFIH* method, until a complete solution for the VRPTW is found. This neighborhood operator never generates a new solution *f'* that violates any constraints, because the *PFIH* algorithm does not allow such possibility.

The *f'* solution generated through the *f* solution after the application of any neighborhood operator is accepted only if *f'* satisfies every constraint of the VRPTW. In any case of constraint violation, the solution *f* is kept into the SA for the next iteration of the system. This possibility may occur with the operators *swap, insert, scramble* and *inversion*.

For each iteration of the SA, an operator is chosen by withdrawing an positive number from a uniform distribution, which varies from 1 to *k*, being *k* the number of operators of the system (in this work k is equal to 5). The operators are stored in a vector of *k* positions and the one which index is raffled is applied.

3) Temperature Control

The temperature T used initially on the system was set to 100 and, on every 100 iterations, it was reduced to 95% of its current value:

$$T_{(t+1)} = 0.95 \times T_t \qquad (6)$$

In addition to the temperature decrease, the algorithm can also produce a temperature increase, characterizing the non-monotonic aspect of the T variable. This increase in temperature (by the unit) occurs when there is no improvement in the best solution in the last 1000 iterations of the system. In the total, 30000 iterations were used for each execution of the simulated annealing.

Another example of efficient use of non-monotonic control of the temperature is reported in [16].

B. Hill Climbing (HC) Strategy

After the conclusion of the SA process, the hill climbing strategy takes places to compose the hybrid solution (HS) proposed in this work.

The motivation to introduce the HC strategy comes from the observation that the solution of the SA may be found in the system when its temperature is considerably high, and, in this case, the neighborhood close (which may contain a better solution) to the best solution of the SA will probably never be explored. This happens because, when the temperature is high, the Metropolis Criteria will tend to easily carry out a drastic locomotion in the solution space to look for the solution and worse solutions may be accepted by the method with high probability. Taking that into account, it was considered the application of the HC strategy to find a local minimum equal or better than the returned solution of the SA method. When the best solution of the SA is found under low temperatures, however, this local search (HC) turns itself to be unnecessary since it is basically executed by the SA method itself, considering that the Metropolis Criteria will only accept worse solutions with very low probabilities.

The HC process is then executed three times at the end of the SA method, each execution corresponding to 1000 iterations, to consider that different executions of a hill climbing may drive to different regions of the search space (solutions). This is the case because the implemented hill climbing is non deterministic and the execution with the best result is returned by the method. The neighborhood operators are the same as described in Section A.2.

C. Random Restart

With respect to the issue of random initialization, different works have followed different paths. Some works choose the idea of performing a small number of short iterations, based on [14] which poses that, "if the problem is NP-complete, then in all likelihood we cannot do better than exponential time". Although theoretically, in such situations, there may be an exponential number of local maxima where the solution may get stuck on, fortunately, in practice, a reasonably good solution can be found after a small number of iterations. Other works, however, were based on the strategy of executing for each instance the evolutionary algorithm for a long period of time (40 minutes on average) [11].

In the work of [12] it was identified that short executions of the system (1 minute and 17 seconds on average), repeated many times, produced more robust results for the VRPTW. Since simulated annealing is also a stochastic algorithm, the quality of the final solutions over a number of runs shows a certain variance, the same strategy was adopted in implementing the proposed HS (SA with HC). 30 system restarts were applied and the solution that presented the shorter total distance of all the system restarts was considered as the HS solution to the VRPTW.

V. TEST BASE

There are many publications using heuristics and meta-heuristics in the resolution of the VRPTW, making easier comparisons and analysis on new proposed approaches. For discovering the quality and robustness of the different algorithms, these are frequently applied over the Solomon instances [15].

Likewise, the tests of this work were executed over the 56 Solomon instances with 100 customers each. The data descriptions and sources are public and may be found on the internet on the address: http://neo.lcc.uma.es/RADI-AEB/WebVRP/data/instances/solomon/solomon_100.zip.

The Solomon instances are divided into six classes: R1, R2, C1, C2, RC1 and RC2. The instances of type R1 and R2 present customers with random Euclidean coordinates. Instances of type C1 and C2, present customers grouped in clusters. Instances of type RC1 and RC2 present a mix of the two first characteristics (spare and clustered). One common characteristic of the types R1, C1 and RC1 is that their instances impose that few customers have to be attended by each vehicle, introducing the need for more vehicles to attend all the demand. The types R2, C2 e RC2 present few vehicles in the solution, to attend a great number of customers in each route.

VI. RESULTS

In subsection A, the best results obtained in the literature are compared with the best results of this work. Subsection B compares results of this work with other four related works ([20],[21],[3] and [2]). The average and the standard deviation for each Solomon's instance are described in subsection C and compared to this work [2].

A. Best known results

For ease of visualization, Tables I, II, III, IV, V and VI point out if the hybrid system proposed in this work obtains equal or better results when compared to the best results found in the literature with respect to the total traveled

distance minimization for the VRPTW. The instances marked with ** denote the situations where the best previous individual result for that instance (picked up from many different authors) were overcame by the proposed method, whereas those marked with * represent the cases where the results of the method paired the previous results.

TABLE I
BEST RESULTS FOUND IN THE R1 PROBLEM CLASS

		Other works			This work	
	Instance	Vehicles	Distance	Work	Vehicles	Distance
*	**R101**	**20**	**1642.88**	**[1]**	**20**	**1642.88**
	R102	18	1472.62	[1]	18	1475.35
	R103	14	1213.62	[17]	15	1222.68
	R104	11	986.10	[2]	11	990.78
	R105	15	1360.78	[2]	15	1363.74
	R106	13	1241.52	[2]	13	1244.58
	R107	11	1076.13	[2]	11	1081.88
	R108	10	948.57	[2]	10	952.37
	R109	13	1151.84	[2]	12	1153.89
	R110	11	1080,36	[17]	12	1087.94
	R111	12	1053,50	[2]	12	1053.80
	R112	10	953,63	[17]	11	973.34

TABLE II
BEST RESULTS FOUND IN THE R2 PROBLEM CLASS

		Other works			This work	
	Instance	Vehicles	Distance	Work	Vehicles	Distance
**	**R201**	**5**	**1148.48**	**[2]**	**8**	**1147.80**
**	**R202**	**9**	**1042.35**	**[12]**	**8**	**1039.32**
**	**R203**	**5**	**876.94**	**[12]**	**6**	**874.87**
**	**R204**	**4**	**736.66**	**[12]**	**5**	**735.80**
**	**R205**	**5**	**960.07**	**[12]**	**5**	**954.16**
**	**R206**	**4**	**887.90**	**[12]**	**5**	**884.25**
**	**R207**	**4**	**811.93**	**[12]**	**4**	**797.99**
**	**R208**	**3**	**707.01**	**[12]**	**4**	**705.62**
**	**R209**	**5**	**860.11**	**[12]**	**5**	**860.11**
**	**R210**	**6**	**912.48**	**[12]**	**5**	**910.98**
**	**R211**	**4**	**761.75**	**[12]**	**4**	**755.82**

TABLE III
BEST RESULTS FOUND IN THE C1 PROBLEM CLASS

		Other works			This work	
	Instance	Vehicles	Distance	Work	Vehicles	Distance
*	**C101**	**10**	**828.94**	**[17]**	**10**	**828.94**
*	**C102**	**10**	**828.94**	**[17]**	**10**	**828.94**
*	**C103**	**10**	**828.06**	**[17]**	**10**	**828.06**
*	**C104**	**10**	**824.78**	**[17]**	**10**	**824.78**
*	**C105**	**10**	**828.94**	**[17]**	**10**	**828.94**
*	**C106**	**10**	**828.94**	**[17]**	**10**	**828.94**
*	**C107**	**10**	**828.94**	**[17]**	**10**	**828.94**
*	**C108**	**10**	**828.94**	**[17]**	**10**	**828.94**
*	**C109**	**10**	**828.94**	**[17]**	**10**	**828.94**

TABLE IV
BEST RESULTS FOUND IN THE C2 PROBLEM CLASS

		Other works			This work	
	Instance	Vehicles	Distance	Work	Vehicles	Distance
*	**C201**	**3**	**591.56**	**[17]**	**3**	**591.56**
*	**C202**	**3**	**591.56**	**[17]**	**3**	**591.56**
*	**C203**	**3**	**591.17**	**[17]**	**3**	**591.17**
*	**C204**	**3**	**590.60**	**[17]**	**3**	**590.60**
*	**C205**	**3**	**588.88**	**[17]**	**3**	**588.88**
*	**C206**	**3**	**588.49**	**[17]**	**3**	**588.49**
*	**C207**	**3**	**588.29**	**[17]**	**3**	**588.29**
*	**C208**	**3**	**588.32**	**[17]**	**3**	**588.32**

Each table in this section considers total traveled distance minimization as the main focus for the VRPTW.

TABLE V
BEST RESULTS FOUND IN THE RC1 PROBLEM CLASS

	Other works			This work	
Instance	Vehicles	Distance	Work	Vehicles	Distance
RC101	15	1623.58	[17]	16	1642.83
RC102	14	1466.84	[2]	15	1480.46
RC103	11	1261.67	[18]	13	1308.64
RC104	10	1135.48	[19]	11	1162.75
RC105	16	1518.60	[1]	15	1534.60
RC106	13	1377.35	[1]	13	1386.82
RC107	12	1212.83	[1]	12	1247.53
RC108	11	1117.53	[1]	11	1135.87

TABLE VI
BEST RESULTS FOUND IN THE RC2 PROBLEM CLASS

		Other works			This work	
	Instance	Vehicles	Distance	Work	Vehicles	Distance
**	**RC201**	**8**	**1267.27**	**[12]**	**9**	**1266.11**
*	**RC202**	**8**	**1096.75**	**[12]**	**8**	**1096.75**
**	**RC203**	**5**	**941.31**	**[12]**	**5**	**926.89**
**	**RC204**	**4**	**788.66**	**[12]**	**4**	**786.38**
**	**RC205**	**7**	**1161.32**	**[12]**	**7**	**1157.55**
**	**RC206**	**7**	**1059.88**	**[2]**	**6**	**1056.21**
**	**RC207**	**5**	**970.78**	**[12]**	**6**	**966.08**
	RC208	4	779.84	[12]	4	780.72

B. Comparisons between different works

TABLE VII
COMPARISON BETWEEN DIFERENT WORKS THAT OPTIMIZE THE TOTAL TRAVELED DISTANCE OF THE VRPTW

Class \ Work		[20]	[21]	[3]	[2]	This work
R1	NV	12.67	13.92	14.17	13.25	13.33
	TD	1200.33	1211.22	1214.86	1183.38	1186.94
R2	NV	3.00	4.91	5.27	5.55	5.36
	TD	966.56	917.54	930.18	899.90	878.79
C1	NV	10.00	10.56	10.00	10.00	10.00
	TD	830.75	846.88	829.77	828.38	828.38
C2	NV	3.00	3.88	3.25	3.00	3.00
	TD	592.24	598.70	604.84	589.86	589.86
RC1	NV	12.13	13.75	14.25	12.88	13.25
	TD	1388.15	1399.76	1385.12	1341.67	1362.44
RC2	NV	3.38	5.63	6.25	6.50	6.13
	TD	1133.42	1055.61	1099.96	1015.90	1004.59
All classes	CNV	423	502	508	489	488
	CTD	57423	56682	56998	55134	55020

Table VII summarizes the comparisons of some best works ([20],[21],[3] and [2]) to the algorithm described here through the Solomon's benchmarks. Those works consider the total traveled distance as the main objective function, such as in the case of the HS proposed here.

The columns in Table VII represent the algorithm, the lines show the average number of vehicles and the total traveled distance for each class and the last two lines represent the accumulated number of vehicles and total distance for each class of the Solomon's benchmark, respectively. For each algorithm, the average results with respect to Solomon's benchmarks are reported through NV

(number of vehicles) and TD (total distance). CNV and CTD indicate the cumulative number of vehicles and cumulative total distance over all 56 test problems.

Technical Information about work, hardware, runs, and average time to instance, are respectively as follows: [20] DEC Alpha, 3 runs, 48.3 minutes. [21] Pentium 200 MHz, 1 run, 29 minutes. [3] Hardware, number of runs and time not reported. [2] Pentium 4 2.4GHz, 3 runs, 60 minutes. [This work] Centrino 1.7 GHz, 15 runs, 11 minutes.

Fig. 9 illustrates graphically the cumulative distance for all the works considered for performance comparison.

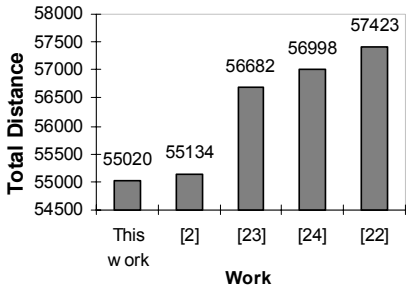

Fig. 9. Cumulative total distance for all the Solomon's instances

C. *Average result of the Hybrid System*

TABLE VIII
AVERAGE AND DEVIATION OF THE SYSTEMS IN THE R1 CLASS

Instance	Total Distance: Work [2]		Total Distance: This work	
	Average ($\bar{x}$)	Standard Deviation (s)	Average ($\bar{x}$)	Standard Deviation (s)
r101	1643.2	0.5	1644.4	1.3
r102	1472.8	0.2	1478.7	2.8
r103	1216.2	3.5	1227.2	2.8
r104	989.3	2.8	1000.7	5.0
r105	1364.6	6.0	1369.6	3.7
r106	1244.1	3.0	1251.0	4.3
r107	1078.6	2.5	1088.1	4.2
r108	951.1	2.6	958.1	3.9
r109	1153.0	1.2	1161.8	5.4
r110	1093.4	1.1	1103.0	8.1
r111	1057.5	6.1	1061.8	5.4
r112	965.8	6.5	980.6	5.4

TABLE IX
AVERAGE AND DEVIATION OF THE SYSTEMS IN THE R2 CLASS

Instance	Total Distance: Work [2]		Total Distance: This work	
	Average ($\bar{x}$)	Standard Deviation (s)	Average ($\bar{x}$)	Standard Deviation (s)
r201	1149.3	1.2	1152.3	4.7
r202	1055.1	5.0	1042.1	2.0
r203	915.2	20.9	876.4	2.3
r204	781.4	8.4	737.3	1.0
r205	977.7	10.5	956.5	2.4
r206	904.7	7.0	885.3	1.2
r207	846.7	10.2	802.4	3.9
r208	733.5	8.6	706.9	0.9
r209	885.4	9.1	862.6	3.6
r210	934.4	1.7	913.0	1.1
r211	802.7	13.2	759.9	2.6

Tables VIII, IX, X, XI, XII and XIII describes the averages ($\bar{x}$) and the standard deviations (s) of the 15 executions of the hybrid system described in this work. This information is tabulated with the results obtained from the work of [2] with comparative means through 3 executions of the system. This work was particularly chosen for further method comparisons since it represented the best results previously published in the problem and was also the only one to provide information that could be used for measuring stability of the method (standard deviation over different runs).

TABLE X
AVERAGE AND DEVIATION OF THE SYSTEMS IN THE C1 CLASS

Instance	Total Distance: Work [2]		Total Distance: This work	
	Average ($\bar{x}$)	Standard Deviation (s)	Average ($\bar{x}$)	Standard Deviation (s)
C101	828.9	0.0	828.9	0.0
C102	828.9	0.0	828.9	0.0
C103	828.1	0.0	828.1	0.0
C104	824.8	0.0	827.5	7.3
C105	828.9	0.0	828.9	0.0
C106	828.9	0.0	828.9	0.0
C107	828.9	0.0	828.9	0.0
C108	828.9	0.0	828.9	0.0
C109	828.9	0.0	828.9	0.0

TABLE XI
AVERAGE AND DEVIATION OF THE SYSTEMS IN THE C2 CLASS

Instance	Total Distance: Work [2]		Total Distance: This work	
	Average ($\bar{x}$)	Standard Deviation (s)	Average ($\bar{x}$)	Standard Deviation (s)
c201	591.6	0.0	591.6	0.0
c202	591.6	0.0	591.6	0.0
c203	591.2	0.0	591.2	0.0
c204	595.9	4.8	590.6	0.0
c205	588.9	0.0	588.9	0.0
c206	588.5	0.0	588.5	0.0
c207	588.3	0.0	588.3	0.0
c208	588.3	0.0	588.3	0.0

TABLE XII
AVERAGE AND DEVIATION OF THE SYSTEMS IN THE RC1 CLASS

Instance	Total Distance: Work [2]		Total Distance: This work	
	Average ($\bar{x}$)	Standard Deviation (s)	Average ($\bar{x}$)	Standard Deviation (s)
rc101	1646.4	6.4	1654.6	5.9
rc102	1475.8	7.8	1483.6	2.8
rc103	1273.2	7.3	1325.8	9.9
rc104	1147.0	10.0	1169.7	6.4
rc105	1534.9	20.5	1550.4	8.1
Rc106	1383.8	10.3	1403.0	10.3
Rc107	1219.0	9.5	1259.7	7.5
Rc108	1125.1	8.3	1158.2	8.9

TABLE XIII
AVERAGE AND DEVIATION OF THE SYSTEMS IN THE RC2 CLASS

Instance	Total Distance: Work [2]		Total Distance: This work	
	Average ($\bar{x}$)	Standard Deviation (s)	Average ($\bar{x}$)	Standard Deviation (s)
Rc201	1277.9	3.2	1273.6	5.8
Rc202	1119.5	6.1	1099.3	1.6
Rc203	962.8	14.6	933.9	4.9
Rc204	808.9	9.1	788.4	1.8
Rc205	1163.4	2.4	1159.2	2.0
Rc206	1076.5	14.9	1062.0	5.6
Rc207	987.3	9.6	971.8	4.1
Rc208	814.9	28.4	785.5	3.0

VII. CONCLUSIONS

This work presented a hybrid system (HS) that combines simulated annealing with non-monotonic temperature control, random start and a hill climbing strategy for the optimization of the total traveled distance of the Vehicle Routing Problem with Time Windows (VRPTW).

The hybrid system offers a relevant contribution in the research of the best techniques to solve the VRPTW. Its main gain, in relation to the previous works conducted so far, is a substantial improvement in solving the R2 problem classes (R2, C2 and RC2) of the Solomon's benchmark. The solutions found by the HS for each instance contains few routes, and these, by themselves, contain many customers for the attendance. In the type 2 classes, this work obtained 17 new best results when compared to the ones found in the literature on minimization of the total traveled distance, and equaled other 10 best results (obtaining success in all but one instance (RC208)) out of the 28 tested. With respect to the classes C1, R1 and RC1, (where the solutions for each instance contain many routes and few customers for attendance), this work only paired the best results of the C1 class and the instance R101, being inferior in 19 out of the 29 tested instances.

It is important to remark the small standard deviation generated in the 15 executions of this work, where it showed regularity in most of the instances of the Solomon's benchmark, when compared to the system presented in the work of [2].

A general comparison with other works that focus on the total traveled distance of the VRPTW ([20],[21],[3] and [2]) shows the superiority of this work in the R2 and RC2 classes, and a general advantage when considered the sum of the distances achieved for all the Solomon's instances. These results encourage a further continuation of this research to refine the algorithm towards a performance improvement for the general classes of the VRPTW and to examine the use of this algorithm in real world operational situations where practical restrictions to attend real needs of routing problems have to be considered.

REFERENCES

[1] G. B. Alvarenga. G. R. Mateus. A Two-Phase Genetic and Set Partitioning Approach for the Vehicle Routing Problem with Time Windows. Fourth International Conference on Hybrid Intelligent Systems (HIS04). IEEE Computer Society Press. (2004).

[2] G. B. Alvarenga. Um algoritmo híbrido para os problemas de Roteamento de Veículos Estático e Dinâmico com Janela de Tempo. PhD Thesis. Department of Science Computer. Federal University of Minas Gerais - Brazil (2005).

[3] De Backer, B. and V. Furnon, "Meta-heuristics in Constraint Programming Experiments with Tabu Search on the Vehicle Routing Problem", presented at the Second International Conference on Metaheuristics (MIC´97), Sophia Antipolis, France. (1997)

[4] T. H. Cormen; C. E. Leiserson; R. L. Rivest; Introduction to algorithms. The MIT Press. (1999).

[5] G. B. Dantzig. R. H. Ramser; The Truck Dispatching Problem Management Science. 6. 80 (1959).

[6] E. Eiben; J. E. Smith. Introduction to Evolutionary Computing. Natural Computing Series. MIT Press. Springer. Berlin. (2003).

[7] G. F. King; C.F. Mast; Excess Travel: Causes. Extent and Consequences; Transportation Research Record 1111. 126-134. (1997).

[8] Kirkpatrick. S.. Gellat. D. Vecchi. M. P.. *Optimizations by Simulated Annealing*. *Science* v. 220. pp. 671-680. 1983.

[9] J. Larsen; Parallelization of the vehicle routing problem with time windows. Phd Thesis. Department of Mathematical Modeling. Technical University of Denmark. (1999).

[10] J. Lenstra; A. Rinnooy Kan; Complexity of vehicle routing and scheduling problems. Networks 11. 221-227. (1981).

[11] H.C.B. de Oliveira. M.M. de Souza. G.B. Alvarenga. R.M.A. Silva. "Uma adaptação do Algoritmo Genético no Problema de Roteamento de Veículos com Janela de Tempo". Infocomp Journal of Computer Science. pp. 51-58. (2004).

[12] H.C.B. de Oliveira. G.C. Vasconcelos. G.B. Alvarenga. "Uma Abordagem Evolucionária para o Problema de Roteamento de Veículos com Janela de Tempo". XXXVII SBPO - Simpósio Brasileiro de Pesquisa Operacional. Gramado - Brazil (2005).

[13] Papadimitriou. C. H; Steiglitz. K. "Combinatorial Optimization – Algorithms and Complexity". Dover Publicationsm INC. (1982).

[14] Stuart J. Russell and Peter Norvig. editors. *Artificial Intelligence: A Modern Approach*. Prentice Hall. 2003.

[15] M. M. Solomon; Algorithms for the Vehicle Routing Problem and Scheduling Problem with Time Window Constraints. (1987).

[16] S. R. Thangiah. I. H. Osman. T. Sun. Hybrid Genetic Algorithm Simulated Annealing and Tabu Search Methods for Vehicle Routing Problem with Time Windows. Technical Report 27. Computer Science Department. Slippery Rock University. (1994).

[17] Y. Rouchat and E. D. Taillard; Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. Journal of Heuristics 1. 147-167. (1995).

[18] P. Shaw; Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In Principles and Practice of Constraint Programming - CP98. Lecture Notes in Computer Science. 417-431. M. Maher and J.-F. Puget (eds). Springer-Verlag. New York. (1998).

[19] J. F. Cordeau. G. Laporte. and A. Mercier; A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. Working Paper CRT-00-03. Centre for Research on Transportation. Montreal. Canada. (2000).

[20] Kilby, P., P. Prosser and P. Shaw, "Guided Local Search for the Vehicle Routing Problem with Time Windows", in META-HEURISTICS Advances and Trends in Local Search Paradigms for Optimization, S. Voss, S. Martello, I.H. Osman and C. Roucairol (eds), 473−486, Kluwer Academic Publishers, Boston. (1999)

[21] Riise, A., M. Stølevik, "Implementation of Guided Local Search for the Vehicle Routing Problem", SINTEF Internal report STF42 A99013, SINTEF Applied Mathematics, Norway. (1999)

[22] Bräysy, O. and M. Gendreau. Metaheuristics for the Vehicle Routing Problem with Time Windows. Internal Report STF42 A01025, SINTEF Applied Mathematics, Department of Optimization, Oslo, Norway. (2001)

[23] Bräysy, O. and M. Gendreau, "Route Construction and Local Search Algorithms for the Vehicle Routing Problem with Time Windows", Internal Report STF42 A01024, SINTEF Applied Mathematics, Department of Optimisation, Norway. (2001)

[24] Bräysy, O., W. Dullaert and M. Gendreau. Evolutionary Algorithms for the Vehicle Routing Problem with Time Windows. Internal Report STF90 A04406, SINTEF ICT, Department of Optimization, Norway. (2004)