Evolutionary Algorithm for the Vehicles Routing Problem with Time Windows Based on a Constraint Satisfaction Technique[1]

Algoritmo Evolutivo para el Problema de Ruteo de Vehículos con Ventanas de Tiempo Basado en una Técnica de Satisfacción de Restricciones[1]

Marco Antonio Cruz Chávez and Ocotlán Díaz Parra
CIICAp, Autonomous University of Morelos State
Av. Universidad 1001. Col. Chamilpa, C.P. 62210.Cuernavaca, Morelos, México
mcruz@uaem.mx, odiazp@uaem.mx

Article received on April 22, 2008; accepted on November 10, 2008

Abstract
In this paper a Memetic Algorithm (MA) is proposed for solving the Vehicles Routing Problem with Time Windows (VRPTW) multi-objective, using a constraint satisfaction heuristic that allows pruning of the search space to direct a search towards good solutions that represent the individuals of the population. An evolutionary heuristic is applied in order to establish the crossover and mutation between sub-routes. The results of MA demonstrate that the use of Constraints Satisfaction Technique permits MA to work more efficiently in the VRPTW.
Key Words: Memetic algorithm (GA-PCP), Constraints Satisfaction Problem, Precedence Constraint Posting, local search, VRPTW.

Resumen
En este documento se propone un Algoritmo Memetico (MA) para resolver el problema de ruteo vehicular con ventanas de tiempo (VRPTW) multi-objetivo, usando una heurística de satisfacción de restricciones que permite podar el espacio de búsqueda para dirigir la búsqueda hacia buenas soluciones las cuales son representadas por los individuos de la población. Se aplica una heurística evolutiva para establecer el cruzamiento y mutación entre subrutas. El resultado del MA demuestra que el uso de la Técnica de Satisfacción de Restricciones permite al MA trabajar más eficientemente en el VRPTW.
Palabras clave: Algoritmo Memetico (GA-PCP), Problema de Satisfacción de Restricciones, Estableciendo Restricciones de Precedencia, Búsqueda Local, VRPTW.

1 Introduction

One of the first forerunners of genetic algorithms was John Holland in 1960 [Mitchell, 1999, Holland, 1975]. The mere structure of a GA (Genetic Algorithm) involves three types of operators: Selection, crossover and mutation [Alvarenga, et al., 2007, Goldberg, 1989, Coley, 1997, Gen, Cheng, 2000, Adeli, Hung, 1995].

By definition, the search carried out by a GA in the solution space of a problem is global (exploration in the search space). When global search is combined with local search (exploitation in the search space), a genetic hybrid algorithm, called an MA (Memetic Algorithm), is formed [Krasnogor, Smith, 1987]. This MA, because of the new characteristics contributed by the local search, is able to find better solutions than a simple GA because for each solution S obtained by the global search, the algorithm searches the neighborhood of S for the local optimum, which could turn out to be the global optimum. Researchers suggest that involving the technique of local search in GA, allows for results nearer to the global optimum to be found in combinatorial optimization problems [Tavakkoli-Moghaddam, et. al, 2006, Moscato, 1989, Moscato et al., 2003].

In this paper a Memetic Algorithm is proposed called GA-PCP, which combines two techniques of search, local and global. For the local search, the algorithm used was one of constraint satisfaction PCP (Precedence Constraint

[1] This work was supported by project 160 of the Fideicomiso SEP-UNAM, 2006-2007.

Posting) proposed by Cheng and Smith [Cheng-Chung, Smith, 1996]. For the global search, the simple crossover of a GA was used.

In order to prove the efficiency of the proposed algorithm, GA-PCP was applied to the Vehicles Routing Problem well-known like VRPTW (Vehicles Routing Problem with Time Windows) which is an NP-complete problem [Solomon. 1987, Garey, Johnson, 2003]. The VRPTW [Toth, Vigo, 2001, Thangiah, 1995, Tan, et al., 2001] is a variant of the VRP with the additional restriction of the time window associated with each client. This window defines an interval within which the client has to be assisted. The objective is to reduce the number of the vehicles, the route time sum, and the necessary wait time to provide all clients the times of attention required.

Very little research of MA exists as applied to VRPTW. In [EL Rhalibi, Kelleher, 2003], a Memetic is proposed using a GA for a constraint satisfaction model of VRPTW with rescheduling and optimization of Pareto. The algorithm includes three local searches, Route-exchange, mutation and lambda-exchange. In [Chin, et al., 1999] a Memetic is proposed that combines TS (Tabu Search) and GA. TS is used for its excellent local search execution capacity which allows for exploitation of the solutions space, while GA is able to diversify these local searches, allowing for the exploration of several regions in the search space. In [Tan, et al., 2003], a Memetic multi-objective is proposed that incorporates three heuristics of local exploration. The first heuristic, Intra_Route, generates two different numbers based on the sequence size of the route assignment of both vehicles. This heuristic chooses two routes randomly and exchanges two nodes of each route. The second heuristic, Lambda_Interchange, assumes that two routes A and B are selected, and begins by sweeping the nodes of route A and moving the feasible nodes into B route. The third heuristic, Shortest_pf, is a modification of the shortest path first method, which tries to change the order of the nodes of a particular route and uses the optimization concept of Pareto to solve multi-objective optimization in VRPTW.
In this paper, in order to apply PCP to VRPTW, the problem was treated as a CSP (Constraint Satisfaction Problem). The constraint satisfaction works with problems that have finite domains like VRPTW, which is a discreet optimization problem. A solution to a CSP is an assignment of values to all the variables such that all restrictions of the CSP are satisfied. The most common techniques in CSP management can be organized in three groups: **systematic search techniques, inference techniques and hybrid techniques** [Castillo, et al., 2005].

The **systematic search techniques** that are commonly used in CSP in order to explore the solution space of the problem are the following two: the GT method (Generate and Test) and the BT method (BackTracking). The GT method generates the possible instance tuples of all variables systematically and later tests successively each instance tuple in order to prove whether or not all of the constraints of the problem are satisfied. The instance tuple that satisfies all the restrictions is a solution to the problem. The BT method carries out an depth exploration of the search space successively instancing the variables and checking each new instance to determine whether the partial instances carried out are locally consistent. If they are consistent, it continues with the instance of a new variable. If there is conflict, it assigns a new value to the last instanced variable if possible. If it is not possible, it backtracks to the assigned variable immediately previous. BT generally suffers a combinatorial explosion in the search space and BT alone is not efficient in solving CSP.

The **inference techniques** attempt to deduce new restrictions derived from the explicitly known problem restrictions. These techniques erase inconsistent values from the domains of the variables or induce implicit constraints between variables to obtain a new CSP. Finding new solutions allows the delimitation of the solution space because of having eliminated inconsistent values.

The **hybrid techniques** are a product of the inclusion of an inference technique and a systematic search technique. The inference technique is used in order to detect and eliminate local inconsistencies in order to delimit the search space. While searching, they use a type of systematic search. These hybrid techniques define the algorithms of search as look-Backward and look-Ahead. The look-Backward carries out the confirmation of the consistency of the previous partial instance. The look-Ahead makes the confirmation inferential for a future instance.

In this work, GA-PCP uses the **hybrid search constraint satisfaction technique** for a CSP using the PCP look-Ahead algorithm.

The PCP local search algorithm involves the calculation of the shortest path, partially and globally, between a pair of nodes and among all the nodes respectively, in the graph that represents the VRPTW model [Cruz-Chávez, et al., 2007]. PCP is applied specifically to disjunctive graphs models. PCP fixes the address of each edge based on the

execution of certain rules and converts the disjunctive graph into a digraph. The shortest path of the digraph represents a feasible solution to VRPTW. The representation of results that is obtained by PCP is coupled with the model of the VRPTW, which is modeled by means of a digraph in order to represent the routes, clients, demand for the client and times of attention required by the client (time window). PCP carries out a series of transformations in order to establish the address of edges in a graph, the set of transformations that is carried out to change an edge is small, since every time that it returns only a small change in the address of an edge is made. PCP behaves similarly to the ramification of a tree and a bounded solution space, which carries out a local search, where each transformation is considered near. These transformations are called local transformations and the method is known as local search [Aho, et al., 1988].

The result obtained in this research is that the combination of PCP with GA applied to VRPTW improves the results for several benchmarks, depending on the percentage of PCP applied to the population used in GA.

The structure of the paper is as follows; section one is the introduction, section two explains the procedures of the proposed GA-PCP algorithm and the conventional GA algorithm for the Vehicles Routing Problem with Time Windows, section three shows the experimentation and comparison of results generated by the GA-PCP algorithm compared with the results obtained by another GA that use constraints satisfaction techniques for the Solomon benchmarks, section four presents conclusions.

2 GA-PCP algorithm for VRPTW

The genetic algorithm in its two versions, the GA-PCP algorithm and the GA algorithm, when applied to the vehicles routing problem with time windows, uses individuals. Each of these individuals represents a set of chromosomes, each one of these is formed by a set of genes. A gene represents a node (client) of a positive integer, including zero. A set of a group of genes represents a vehicle, such that a solution for the VRPTW represents an individual in a genetic algorithm [Tavares, et al., 2003]. The following is an explanation of the two versions of the algorithms and their main differences.

2.1 GA-PCP algorithm

Figure 1 is a general outline of the proposed algorithm called GA-PCP (Genetic Algorithm with Precedence Constraint Posting) for VRPTW.

The GA-PCP algorithm consists of the following general steps:

Step 1. Creation of the initial population comprised of individuals with route information and individuals with Time windows information. The initial population is created with 50% of individuals with the PCP procedure. It is completed 100% by individuals generated of randomly form (Figure 2).

Step 2. Apply the tournament selection to the initial population. The tournament selection consists basically of taking the list of the population called P, and copying that list and order according to the aptitude of each individual. This resulting list is called M. With these two lists, the comparison is begun between two individuals from the P list with two from the M list. The one that has lesser fitness from the P list taken as a candidate and is combined with the individual of lesser fitness from the M list. This continues until the end of the lists. This procedure can be seen in Figure 3(A).

Step 3. Apply crossover *k*. The crossover is shown in Fig. 3(B). A pair of individuals is chosen randomly from the population. A pair of genes is selected randomly, which are the same for both individuals but which are located in inverse positions. For example, if the pair (4,3) is chosen from individual one, the pair chosen from individual two would be the same pair but in inverse position, (4,3). Next the crossover of the pair of individuals begins, as shown in Figure 3(B). This procedure is a permutation of a pair of genes for each individual. This is a Flip-bit on two individulas.

Step 4. Apply flip bit mutation. La mutación flip-bit, consiste en el intercambio de un par de genes en un individuo. El intercambio se genera aleatoriamente [Lahoz-Beltra, 2005].

Step 5. Construction of the following generation with migrant quality individuals (with PCP) and individuals of the population obtained of a convencional GA procedure (Figures 2, 3). In order to create the initial population, a certain percentage *x* of the population generated by the genetic phase of the GA is taken, and a certain percentage *y = 100-x* of another migrant population generated with the PCP is taken [Cruz-Chávez, et al., 2007]. The sum of the (*x*, *y*) percentage is 100% of the new generation to be evaluated.

a) Construction of the population x: There are different types of genetic operators applied in the procedure of generation of *x* population for the genetic phase. One is the tournament selection method operator, which consists of taking the list of the population (INDIVIDUAL LIST P), making a copy of that list and putting it in order according to the fitness F of the individual, from smallest to greatest (INDIVIDUAL LIST M). After making these two lists, two individuals from the INDIVIDUAL LIST P and two individuals from THE INDIVIDUAL LIST M are compared. The one that has the least fitness on the INDIVIDUAL LIST P becomes a candidate to be combined with the individual which has the least fitness from the INDIVIDUAL LIST M. For the case in which the fitness is equal for both individuals, either of the two individuals can be chosen. This process is repeated until the ends of the lists are reached (see Figure 3A). The crossover proposes a new method that consists of finding two points randomly in individual RANDOM1 and looking for the corresponding genes to make the crossover in individual RANDOM2. This guarantees the fulfillment of one of the restrictions of the VRPTW which is not passing twice through the same node (see Figure 3B). For the mutation, the Flip-Bit method is used which consists of taking two genes randomly from the same individual, with *gen1* being different than *gen2*, and proceeding to exchange the places of gen1 and *gen2* (see Figure 3C). Figure 3, illustrates the way in which this process is carried out by each one of these operators.
b) Construction of the population *y*: The PCP procedure is applied for the construction of feasible individuals. This procedure is explained in extended form in section 2.1.1.

Step 6. Evaluate the fitness with objective function that is shown in equation (1), for the case of the VRPTW problem, two primordial objectives are used: the demand and attention time to each client, trying to minimize the cost implied by these two objectives.

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} X_{ijk} \tag{1}$$

Subjet to:

$$a_i \sum_{j \in \Delta^+(i)} X_{ijk} \leq w_{ik} \leq b_i \sum_{j \in \Delta^+(i)} X_{ijk} \qquad \forall k \in K, i \in N \tag{2}$$

In equation (1), *c* represents the cost of transporting of an *i* origin to a *j* destination, *X* represents the journey of an *i* origin to a *j* destination in a *k* vehicle. In order to complete the cost objective, the minimum number of vehicles assigned to each journey is searched for, while fulfilling the capacity constraint of the vehicle and the time window. For the journey, the attempt is to find the shortest distance. In equation (2) w_{ik} represents the time window for the node *i* taken care of by vehicle k. The vehicle k is taken from the fleet K of vehicles, for any i node that belongs to set N of the natural numbers. In order to exemplify the representation of the equation (2), suppose that for the node i = 3 that is assisted by the vehicle k = 1 has a time window $w_{3,1} = 4$ and one interval $[a_i,b_i] = [1,5]$ where a_i represents the arrival time *i* node. It does not have to be minor to 1, and b_i that represents the departure time of *i* node does not have to exceed to 5, in order to fulfill the restriction of time of attention the client.

Step 7. Verify whether the stop criterion is satisfied. The stop criterion is set based on the execution time and the generation number. If some of these criterions are satisfied, the GA-PCP execution is finished.

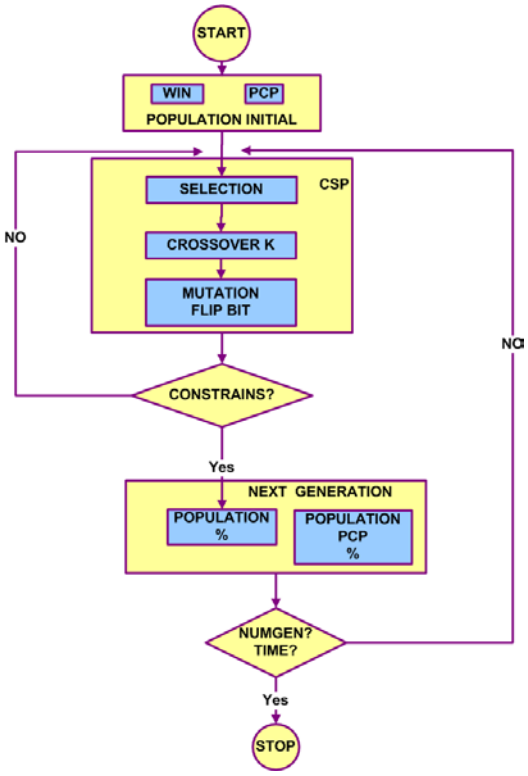

Fig. 1. Flow Diagram of GA-PCP algorithm

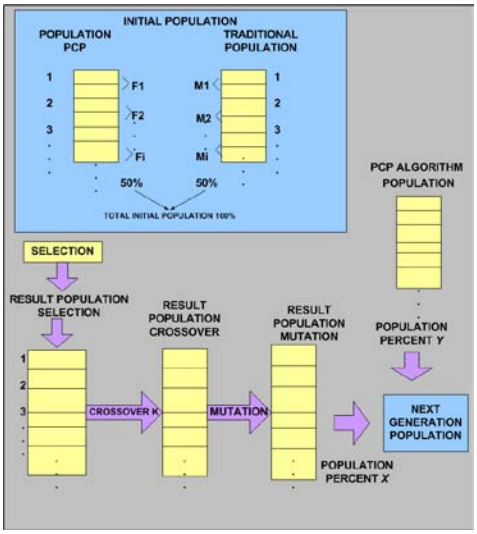

Fig. 2. Procedure general that builds the GA-PCP population

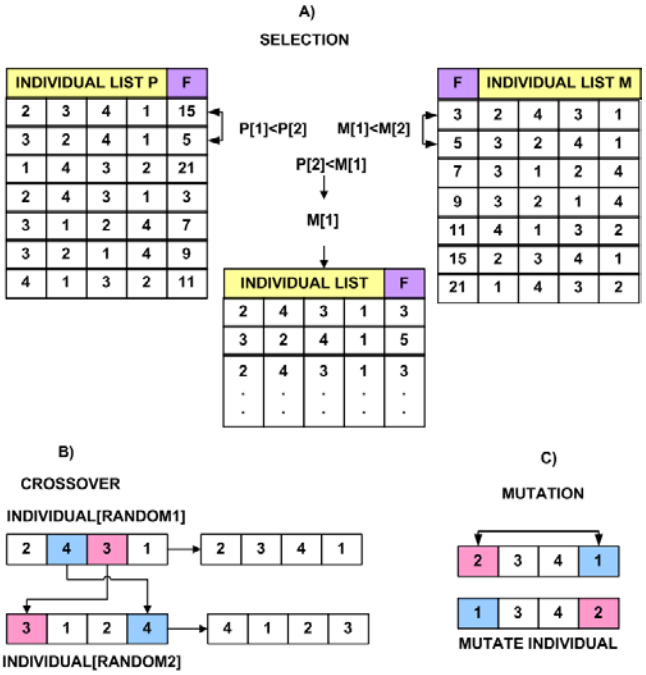

Fig. 3. Procedure for the generation the population *x* by applying the genetic phase of the GA-PCP algorithm. Figure was taken from [Tan, et al., 2001] [Zhu, 2000]

2.1.1. Construction of the PCP population

Within the local search procedure, PCP builds the solution through Depth First using partial assignments of Ω (the set of pair of nodes *i, j* of the VRPTW disjunctive graph). The PCP algorithm carries out a pruning of the search space early on and provides a heuristic for the assignment of values of the $Ordering_{ij}$ variables.

PCP consists of a series of cases in which it should be true that if the shortest path (*sp*) between a pair of nodes (*i, j*) that represent the $Ordering_{ij}$ variable then it has a value that fulfills **some** of the PCP cases. According to the result obtained upon evaluating the shortest path, the value of $Ordering_{ij}$ is designated. The evaluation of *sp* is calculated from *i* to *j* (sp_{ij}) and from *j* to *i* (sp_{ji}).

The PCP algorithm applies the disjunctive graph model of VRPTW. PCP obtains a digraph as a result, and with the help of a greedy algorithm, finds the shortest path from a node of origin to any node of the digraph, evaluating all the paths between nodes and choosing the shortest of these paths. In order to evaluate the shortest path, the Dijkstra algorithm called minimum route [Smith, et al., 1982] is used, the number of optimum routes is obtained that satisfies the capacity restriction of each vehicle used in optimum form that represents a feasible solution to the problem.

The local search PCP algorithm applied to VRPTW for the CSP consists of the following four steps:

Step 1.- Find the shortest path for each unordered pair of nodes sp_{ij} and sp_{ji}.
Step 2.- Classify the decision of ordination of the pairs not ordered with four cases
Case 1. If $sp_{ij} >= 0$ and $sp_{ji} < 0$ then $O_i \prec O_j$ should be selected.
Case 2. If $sp_{ji} >= 0$ and $sp_{ij} < 0$, then $O_j \prec O_i$ should be selected.
Case 3. If $sp_{ji} < 0$ and $sp_{ij} < 0$, then the partial solution is inconsistent.
Case 4. If $sp_{ji} >= 0$ and $sp_{ij} >= 0$, then no relationship of order is possible
Step 3.- Existence of cases
Does either case 1 or case 2 exist?
If one exists, go to step 4

If neither exists, go to step 1

Step 4.- Fix new precedence for unordered pairs.

In order to better understand the algorithm, an example is shown of a small instance of five nodes and a vehicle with a capacity of 200 packages. The disjunctive graph model that is obtained is presented in Figure 4.

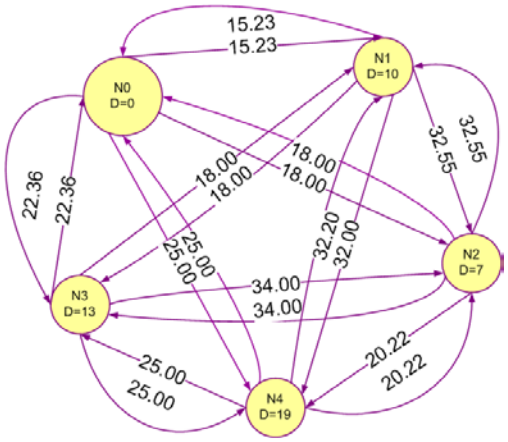

Fig. 4. Disjunctive graph

Applying the shorter path algorithm between pairs of nodes and evaluating the PCP cases, the graph shown in Figure 5 is obtained. The resulting graph does not generate a feasible solution, this means that a route from the initial node to the final node does not exist through which each node is passed only once.

Because the resulting graph does not generate a solution, backtracking is applied in the nodes with an enter zero and exit zero, leaving fixed the nodes that have at least one entrance and one exit. The PCP algorithm is applied in order to find a route, if a feasible solution is generated, it is taken as a solution. A solution of the problem is shown in Figure 6. Lastly, a greedy algorithm is used which divides the shortest path presented in Figure 5into a set of routes in order to satisfy the demand constraints of the client and capacity of the vehicles.

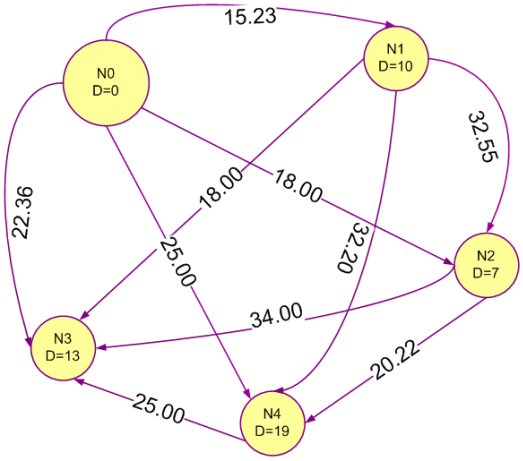

Fig. 5. Conjunctive graph

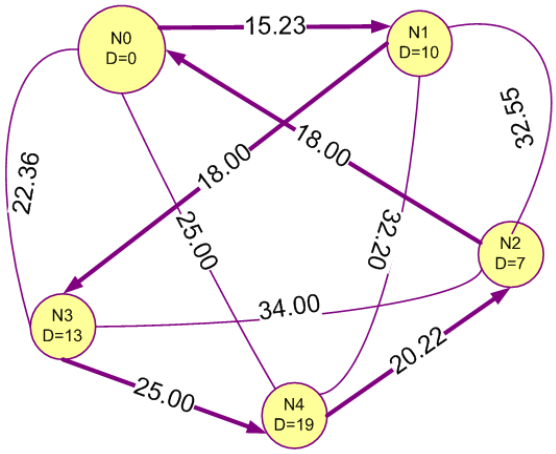

Fig. 6. Solution graph

There are three criteria for stopping the algorithm, it is carried out: (1) for an established amount of time, (2) until a certain range of an optimum is reached, or (3) for a certain number of generations. In this work is used only the (1) and (3).

2.2 GA algorithm

Figure 7, presents the GA algorithm. This algorithm begins with a randomly generated population, and the subsequent generations of the population are built conventionally by applying tournament selection, crossover and mutation flip-bit. The difference that exists between the GA-PCP algorithm and the GA algorithm mainly resides in the construction of each population. In the GA algorithm the initial population is generated randomly. In the GA-PCP algorithm the initial population is formed by mixing individuals that from a randomly generated population and individuals generated with the PCP technique. In the following generations, the corresponding population for the GA algorithm is built conventionally, as was already explained. For the case of the GA-PCP algorithm, the population for the following generation is formed by the mixture of a percentage of feasible individuals generated by the application of the PCP method (see section 2.1.1) and a percentage of individuals generated conventionally as is done in the GA algorithm.

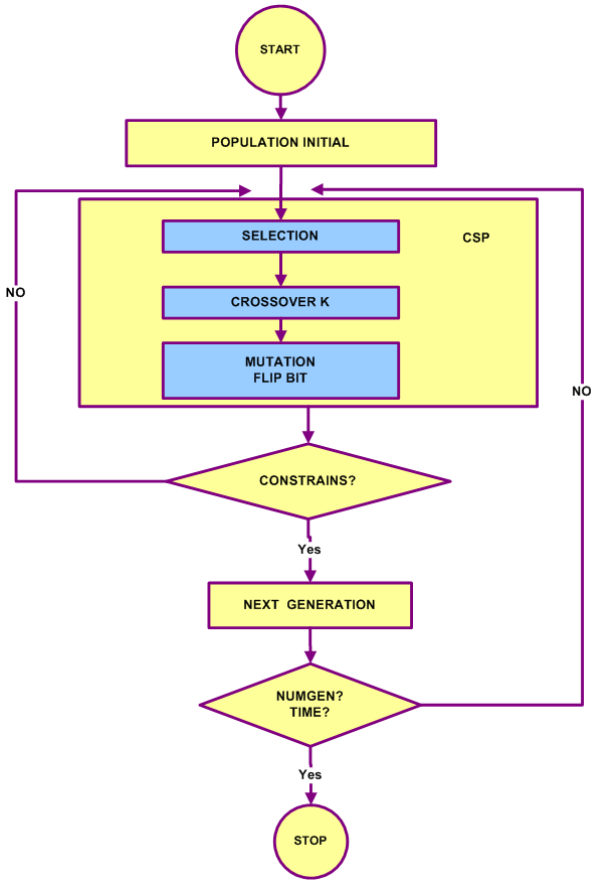

Fig. 7 GA-Algorithm

2.3 GA-PCP algorithm complexity

In order to calculate the temporary complexity of the GA-PCP algorithm, an analysis of the temporary function that represents it was carried out, based on the proposed method by Aho [Aho, et al., 1988]. Being of this analysis a poly-

nomial of third degree, it is represented as $T(n) = c_1 n^3 + c_2 n^2 + c_3 n + c_4$. The complexity of the proposed algorithm is $O(n^3)$, where n is the number of (nodes) clients in the problem. In the case of the GA-Algorithm, the temporary function that represents it is $T(n) = c_1 n^3 + c_2 n^2$, which is also a polynomial of third degree, but with a smaller number of terms. The asymptotic complexity of the two algorithms is the same, that is, $O(n^3)$, which indicates that the efficiency of both algorithms for instances of big size will be very similar. With regard to the efficacy of both algorithms, this will depend on the quality of exploration and exploitation that they contribute for the search of the global optimum in the solution space of the problem. In this case, both algorithms use the same procedure for the exploration (crossover). For the exploitation of the GA-PCP, the procedure includes a local search PCP carried out in each iteration of the algorithm for the construction of new generations. The conventional GA-Algorithm does not have a procedure that permits a good exploitation of the solution space. Results of another GA of CSP type are also presented in order to carry out comparisons of efficiency and efficacy with the proposed GA-PCP.

3 Experimental results

The VRPTW problems used in the experiment are taken from the Solomon benchmarks [Solomon, 1987]. The instances for VRPTW are classified by type and by class. Two types of instances exist; type 1 manages narrow windows of time and small vehicle capacity, type 2 manages large windows of time and large vehicle capacity. Three classifications exist, C, R and RC. The C classification includes the instances that have a territorial distribution for clients bunched together. The R classification has the clients evenly distributed in a territorial area. The RC classification is the combination of territorial bunched together and distributed distribution is. The Solomon benchmarks for VRPTW used in this experimentation are types C1, R1, RC1, C2, R2 and RC2. Figure 8 shows the characterization of these instances.

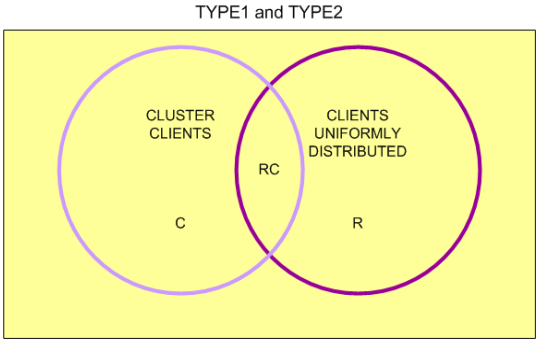

Fig. 8. Characterization of VRPTW instances

The results of the proposed GA-PCP algorithm were compared with the results of the GA algorithm, a conventional version of GA, which does not use the local search PCP. The search in GA only applied the techniques of tournament selection and point crossover. Also the proposed algorithm GA-PCP is compared with others that use CSP to evaluate the efficiency and efficacy of the proposed algorithm. The stop criterion of both algorithms is defined with regard to the execution time. The purpose of this is to be able to evaluate efficacy within the same amount of time. For all the tests carried out in this investigation, the time of execution was one hour.

The results that are reported were obtained in a computer with the following characteristic: Pentium processor (R) M to 1.60 GHz, 1GB RAM, operating system XP Windows, and compiler visual C+ 6.0.

The instances used in the experiment were C104, R104, RC108, C204, R208, RC208, for 25 nodes. C104, R104 and RC108 were used as instances of 100 nodes. The algorithms of CSP type used as comparison for the proposed algorithm are the GGA (Generic Genetic Algorithm) [Affenzeller, 2002], SSGA (Steady-State Genetic Algorithm)

[Chafekar, et al., 2003] and SXGA (Sexual Genetic Algorithm) [Wagner, Affenzeller, 2005]. In all the tests carried out in this investigation, for all the algorithms, the size of the initial population was fixes at 1000 individuals and the number of generations was 20. The efficacy and efficiency of the GA-PCP algorithm are explained later, based on the obtained results presented in Tables 3 and 4.

Table 1 presents the results obtained with the GA-PCP algorithm. The results that have the ** label represent the better well-known value in the instances R104-100 and RC108-100. OP* is the optimum reported in literature, V is the number of vehicles found by the GA-PCP algorithm, V* is the optimum number of vehicles reported in literature and RE is the Relative Error. The RE for the experimentation involves the best known value in literature for the problem and the best found value; it is defined for the equation (3). The RE is a measure in percentage of the distance of the result generated by the experimentation to the best known value for the instance that is being evaluated.

$$RE = \frac{Best - Op^{**}}{Op^{**}} * 100 \tag{3}$$

Thirty executions were carried out for each benchmark; the reported results include the results of the executions, the best, and medium values as well as the standard deviation. The time of each execution was one hour. This stop criterion was chosen in order to be able to make a good measure of the efficacy of the algorithm proposed with regard to the others with which it is compared. Table 1 shows that for the problems of 25 nodes, the best result is near the global optimum, for C104-25 and RC108-25 it was reached with regard to the distance, but for C204-25 the relative error (RE) was large. The results for 25 nodes show that GA-PCP works acceptably if the problem is R and/or C classification, when the clients are evenly distributed or bunched together respectively in a territorial area, and when the time windows are narrow and vehicles with a small capacity (type 1) are used, see the results for C104-25, R104-25 and RC108-25. When the problem is of R and C classification but has a big time window and the vehicles have a large capacity (type 2), the results begin decrease in quality, see the result for R208-25 and RC208-25. When the problem only has the C property but has a large time window and the vehicles have a large capacity (type 2), the results are very poor, see the result for C204-25. The GA-PCP algorithm generated good results for big instances of 100 nodes with regard to the well-known bouds [Dorronsoro, 2007].

Table 2 presents the results obtained by the GA-PCP and GA algorithms. When GA algorithm is used the solution could be of even poorer quality as can be observed for problems of 25 nodes. It can be observed that GA-PCP obtains results near the global optimum (less than 1% RE) for instances of type 1 (small time window and small vehicle capacity) with C and RC classification. With GA algorithm the results were obtained which were very near the global optimum (less than 1% RE) for instances of type 2 (large time windows and large vehicle capacity) with C and RC classification. For R classification type 1 result were not near the global optimum (greater than 1% RE) for both algorithms. For R classification type 2 results were near the global optimum (less than 1.5 % RE) for both algorithms.

Table 1. Results of the GA-PCP algorithm

Benchmark	V	Best	Average	σ	Op*	V*	RE
C104-25	3	186.9	190.14	2.7	186.9	3	0
R104-25	5	449.3	470.2	21.6	416.9	4	7.77
RC108-25	5	294.70	304.8	7.5	294.5	3	0.07
C204-25	2	282.05	288.3	2.9	213.1	1	32.36
R208-25	2	328.5	331.3	3.4	328.2	1	0.12
RC208-25	2	270.9	288.5	7.9	269.1	2	0.66
C104-100	12	942.9	1381.2	315.1	822.9	10	1.45
R104-100	14	1001.3	1414.1	218.9	982.01	10	1.96
RC108-100	12	1149.9	1851.3	260.5	1139.82	10	0.88

Table 2. Results of GA-PCP and GA algorithms

Benchmarks	GA-PCP	RE	GA Algorithm	RE	Optimum
C104-25	**186.9**	0	241.5	29.21	**186.9**
R104-25	449.3	7.77	453.3	8.73	416.9
RC108-25	**294.7**	0.07	306.5	4.08	**294.5**
C204-25	282.0	32.36	**213.2**	0.05	**213.1**
R208-25	**328.5**	0.12	332.0	1.16	328.2
RC208-25	270.9	0.66	**269.3**	0.074	**269.1**

Figures 9 and 10 illustrate the percentages of population *y* with PCP for the instances RC108 and C204 respectively. The increments in percentage of PCP in order to guarantee the mixture of populations began as increases of 5% of PCP population, beginning with a 5% increase, and finishing with a 95% increase. The remaining amount in each one of the increments is completed with the randomly generated population in order to arrive at 100% of the total population. Figure 9 shows the global optimum in RC108-25 with $y = 60\%$ for the population of individuals in the GA-PCP. In this figure it can be observed that when *y* increases from 0 to 60%, GA-PCP tends to improve the solution of RC108-25. It can also be seen that when *y* increases from 60 to 95%, GA-PCP tends to worsen the solution of RC108-25. Figure 10 shows a value near the global optimum for C204-25 with $y = 95\%$ of the population of individuals in the GA-PCP. In this figure it is observed that when *y* increases from 5 to 95%, GA-PCP tends to improve the solution of RC204-25.

According to the results reported in Figures 9 and 10, for each instance proven in this paper, the appropriate percentage of the PCP population required in order to improve the efficiency of GA-PCP will be different and will need tuning according to the properties and type of the problem. What is observed in Figures 9 and 10 is that in order for GA-PCP to work efficiently, the percentage *y* of the PCP population should be greater than 50%.

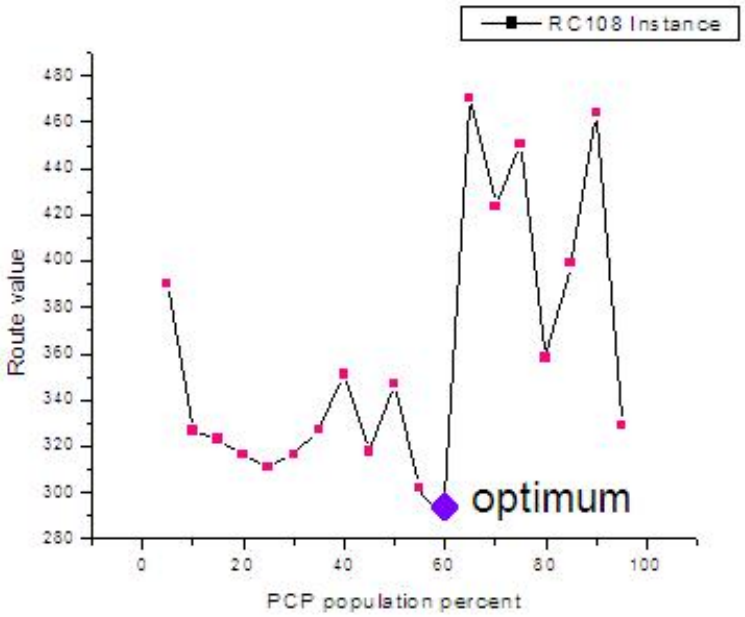

Fig. 9. Percentage of PCP population for instance RC108-25

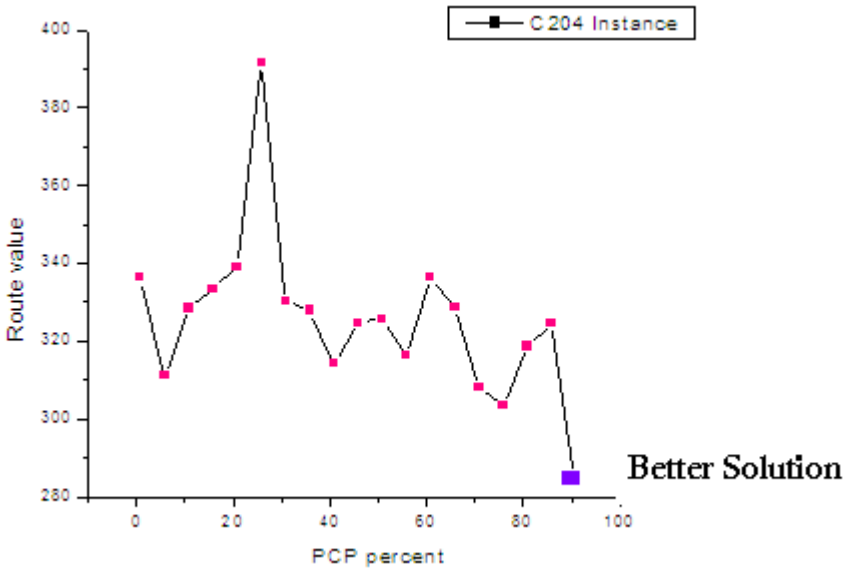

Fig. 10. Percentage of PCP population for instance C204-25

The following is a comparison of the results of the GA-PCP algorithm with another GA that uses the constraints satisfaction techniques. The heuristics laboratory [Wagner, Affenzeller, 2004] implemented the GA used for the comparison. These comparison algorithms are the GGA (Generic Genetic Algorithm) [Affenzeller, 2002], SSGA (Steady-State Genetic Algorithm) [Chafekar, et al., 2003] and SXGA (Sexual Genetic Algorithm) [Wagner, Affenzeller, 2005]. The GGA algorithm uses the constraints satisfaction technique of **systematic search**. the SSGA algorithm uses the constraints satisfaction technique of **inference** by means of a substitution strategy. The SXGA algorithm uses the constraints satisfaction technique of **systematic search** by means of a selection algorithm. These genetic algorithms of the heuristics laboratory report their best results using the following tuning of their entry variables: overload penalty = 50.00, Tardiness penalty = 20.00, Route time penalty = 0.05, Travel time excess penalty = 50.00, Distance penalty = 1.00. The selection operator was tournament. The number of generations and population size is the same as used for GA-PCP, 1000 and 100 respectively. With this tuning the GGA algorithms, SSGA and SXGA were executed, giving the results presented in Tables 3 and 4.

Table 3. Comparative results of efficiency of GA-PCP vs. other algorithms that apply constraints satisfaction technique

Benchmark	**GA-PCP**		**GGA**		**SSGA**		**SXGA**		**Op**
	UB	t, sec	UB	t, sec	UB	t, sec	UB	t, sec	
C104.25	**186.9**	**42.2**	189.3	78.8	187.7	59.4	189.7	79.0	186.9
R104.25	449.3	47.8	416.9	79.0	**417.6**	**0.9**	418.0	79.6	416.9
RC108.25	**294.6**	49.9	295.1	79.0	294.9	0.7	295.4	78.9	294.5
C204.25	279.0	47.9	**221.1**	79.8	223.3	0.9	**223.3**	79.7	213.1
R208.25	**329.1**	49.1	**329.1**	78.6	**329.1**	0.7	**329.1**	79.2	328.2
RC208.25	270.1	52.9	270.6	87.8	**269.3**	0.8	270.0	87.3	269.1

Table 4. Comparative results of the efficacy of GA-PCP with other algorithms that apply the costraints satisfaction technique

Results	Algorithm			
	GA-PCP	**GGA**	**SSGA**	**SXGA**
Problem	C104, OPTIMUM = 186.9			
Best*	**186.9**	189.3	187.7	189.7
Worst	229.2	227.1	206	199.9
Average	190.6	199.8	195.2	196.3
σ	5.09	16.11	5.88	3
RE *	0	0.01284	0.00428	0.01498
Problem	R104, OPTIMUM = 416.9			
Best *	449.3	**416.9**	417	417
Worst	521	424.3	435.1	425.5
Average	470.2	420.4	425.3	420.3
σ	21.6	4.42	7.12	2.54
RE *	7.77	0	0.00023	0.00023
Problem	RC108, OPTIMUM = 294.5			
Best *	**294.6**	294.9	294.9	294.7
Worst	457.9	297	297.9	298.7
Average	302.1	296.3	295.9	296.3
σ	15.08	2.01	1.74	1.19
RE *	0.00033	0.00135	0.00135	0.00067
Problem	C204, OPTIMUM = 213.1			
Best *	278	**219.3**	221.1	220.2
Worst	317.6	223.4	224.7	223.4
Average	293.1	222.1	223.7	221.3
σ	5.11	1.95	1.78	1.74
RE *	0.30455	0.02909	0.03754	0.03331
Problem	R208, OPTIMUM = 328.2			
Best *	**328.8**	**328.8**	328.9	**328.8**
Worst	493.1	331.2	335.7	333.7
Average	335	329.5	334.1	331.9
σ	4.01	0.45	2.16	0.918
RE *	0.00243	0.00152	0.00213	0.00152
Problem	RC208, OPTIMUM = 269.1			
Best *	270.1	270	**269.3**	270
Worst	495.6	279.4	289.5	279.9
Average	290.5	275.4	278.9	275.8
σ	11.19	3.13	7.86	3.55
RE *	0.00371	0.00334	0.00074	0.00334

The tuning percentage of PCP per population depends on the problem. For the instance C104, $x = 0.1$ and $y = 0.9$. For instances R104 and C204, $x = 0.05$ and $y = 0.95$. For instances RC108, R208 and RC208, $x = 0.2$ and $y = 0.8$.

Table 3 presents the times that correspond to the time of the best solution obtained in 30 tests executed by each algorithm in each instance of VRPTW. The population of 1000 individuals and 20 generations was used, with the number of generations defining other stop criterion. Table 3 shows that the efficiency of GA-PCP is better than GGA and SXGA because it obtains better results with regard to the tuning of the entry parameters for each algorithm. It is observed that SSGA is better in efficacy because the times of execution are the shortest.

Table 4 presents results of 30 tests executed by each algorithm in each problem. It shows the best (Best*) and worst results, the average value, the standard deviation, and the relative error (RE*). These results demonstrate that GA-PCP is competitive with these three algorithms that also use the constraints satisfaction technique. One could observe that the proposed algorithm obtains the best results in three of the six problems, that is, for 50% of the revised benchmarks.

4 Conclusions

The results reported in this research indicate that using the PCP local search algorithm in GA improves the results in VRPTW only for small problems of type 1 (small window and small vehicle capacity) with C and RC classification. It is not efficient if type 2 is used (big windows and big vehicle capacity) in problems for instances with C and RC classification.

The initial population is formed of feasible individuals; a randomly selected population is not used. Instead, the initial population is selected in such a way that it consists of feasible individuals that contain route information and time information. For the next generations, a certain percentage of population of PCP is worked with in order to form the total population of the following generation. It was proven that when the population is formed in great part by PCP individuals, the generated results are near the global optimum for small problems of type 1. When small problems of type 2 are used, it is better not to use PCP in GA.

It is demonstrated that for the revised benchmarks, the GA-PCP proposed algorithm is competitive in efficiency and efficacy to comparison algorithms used in this investigation that also apply the constraints satisfaction technique. The GA-PCP obtains the best results in 50% of the problems with competitive times of execution.

The GA-PCP algorithm generates good results for instances of 25 nodes, however for instances of 100 nodes it does not generate good results. It can be seen through the results of this experiment that applying a greater percentage of PCP population improves the result of the solution of the GA.

According to the obtained results, which depended on the type of problem and its coherence to the type of characterization of the instance, it is proposed that future work analyze more deeply the characterization of the instance in order to find possible improvements to the structure of the GA-PCP algorithm, according to the proven instance and therefore make the algorithm more efficient for large instances.

References

1. **Adeli, H. & Hung, S. (1995)**. *Machine Learning: Neural Networks, Genetic Algorithms, and Fuzzy Systems*. New York: John Wiley & Sons
2. **Affenzeller, M. (2002).** A Generic Evolutionary Computation Approach Based Upon Genetic Algorithms and Evolution Strategies. *Journal of Systems Science*, 28(2), 59-72.
3. **Aho, A.V., Hopcroft, J.E. & Ulllman, J.D. (1988).** *Structure of data and algorithms*. New Jersey: Adisson-Wesley
4. **Alvarenga, G.B., Mateus, G.R. & DeTomi, G. (2007).** A genetic and set partition two-phase approach for the vehicle routing problem with time Windows. *Computers & Operations Research*, 34(6), 1561-1584.
5. **Castillo, L., Borrajo, D. & Salido, M.A. (2005).** Planning, Scheduling and Constraint Satisfaction: From Theory to Practice. *Frontiers in Artificial Intelligence and Applications*, 117(1), 1-10.
6. **Chafekar, D., Xuan, J. & Rasheed, K**. **(2003).** Constrained Multi-objective Optimization Using Steady State Genetic Algorithms. *The Genetic and Evolutionary Computation Conference* (GECCO'2003). Athens, Greece, 2723, 813-824.
7. **Cheng, C. & Smith, S.F. (1996)**. A Constraint Satisfaction Approach to Makespan Scheduling. *3th International Conference on Artificial Intelligence Planning Systems*, Edinburgh, Scotland, 29-31.
8. **Chin, A., Kit, H. & Lim, A. (1999).** A new GA approach for the vehicle routing problem. *11th IEEE International Con-ference on Tools with Artificial Intelligence*, Illinois, USA, 307 – 310.

9. **Coley, D.A. (1997)**. *An Introduction to Genetic Algorithms for Scientists and Engineers*. Singapore: World Scientific Publishing Company.
10. **Cruz-Chávez, M. A., Díaz-Parra, O., Hernández, J. A., Zavala-Díaz, J. C., Martínez-Rangel, M. G. (2007)**. Search Algorithm for the Constraint Satisfaction Problem of VRPTW. *CERMA2007*, 0(1), 336-341.
11. **Dorronsoro-Diaz, B. & Alba, E. (2008).** *Cellular Genetic algorithms.* Luxembourg: Springer-Verlag.
12. **ELRhalibi, A. & Kelleher, G. (2003).** An approach to dynamic vehicle routing, rescheduling and disruption metrics. *IEEE Interntional Conference on Systems, Man and Cybernetics*, 4, 3613 – 3618.
13. **Garey, M.R. & Johnson, D.S. (2003).** *Computers and intractability, A Guide to the theory of NP-Completeness*. New York: W.H. Freeman and Company.
14. **Gen, M. & Cheng, R. (2000).** *Genetic Algorithms and Engineering Optimization*. Canada: John Wiley and Sons.
15. **Goldberg, D.E. (1989).** *Genetic Algorithms in Search, Optimization, and Machine Learning*. Canada: Addison Wesley Professional.
16. **Holland, J. (1975).** *Adaptation in Natural and Artificial Systems*. Michigan: The University of Michigan Press.
17. **Krasnogor, N. & Smith, J. (2000).** MAFRA a Java Memetic Algorithm Framework. *Workshops Proceedings of the 2000 International Genetic and Evolutionary Computation Conference (GECCO2000)*, Las Vegas, USA, 125–130.
18. **Lahoz-Beltra, R. (2004).** *Bioinformática: Simulación, vida artificial e inteligencia artificial.* Madrid: Ediciones Díaz de Santos.
19. **Mitchell, M. (1996)**. *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press.
20. **Moscato, P. (1989)**. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. *Technical Report Caltech Concurrent Computation Program*, Report. 826, California, USA.
21. **Moscato, P. & Cotta, C. (2003).** An Introduction to Memetic Algorithms. *Revista Iberoamericana de Inteligencia artificial* 19(2), 131-148.
22. **Potvin, J. Y., Duhamel, C. & Guertin, F. (1996).** Genetic Algorithms for vehicle routing with backhauling. *Applied Intelligence*. 6(4), 345-355.
23. **Prins, C. (2004).** A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research.* 31(12), 1985-2002.
24. **Smith, D. & Wiley, J. (1982).** *Network Optimization Practice. A Computational Guide*. New York: John Wiley &Sons.
25. **Solomon, M.M. (1987).** Algorithms for Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2), 254-265.
26. **Tan, K.C., Lee, L.H., Ou, K. (2001).** Artificial intelligence heuristics in solving vehicle routing problems with time windows constraints. *Engineering Applications of Artificial Intelligence*, 14(6), 825-837.
27. **Tan, K.C., Lee, L.H., Zhu, Q.L., Ou, K. (2001).** Heuristics methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering*, 15(3), 281-295.
28. **Tan, K.C., Lee, T.H., Chew, Y.H., Lee, L.H. (2003).** A multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *IEEE International Conference on Systems*, 1(1), 361 – 366.
29. **Tavakkoli-Moghaddam, R., Saremi A.R., Ziaee M.S. (2006).** A memetic algorithm for a vehicle routing problem with backhauls. *Applied Mathematics and Computation*, 181(2), 1049–1060.
30. **Tavares, J. and Pereira, F. B. and Machado, P. and Costa, E. (2003).** Crossover and Diversity: A Study about GVR. *Analysis and Design of Representations and Operators (ADoRo'2003)*, Genetic and Evolutionary Computation Conference (GECCO-2003), Chicago, Illinois USA, 27-33.
31. **Thangiah, S.R. (1995).** Vehicle Routing with Time Windows using Genetic Algorithms. In L.Chambers, (Ed.), *Practical Handbook of Genetic Algorithms, Vol. 2. New Frontiers*, 253-277. Boca Raton, Fla.: CRC Press.
32. **Toth, P. & Vigo, D. (2001).** *The Vehicle Routing Problem. Monographs on Discrete Mathematics and Applications*. Philadelphia: SIAM.

33. **Wagner, S. & Affenzeller, M**. **(2005).** SexualGA: Gender-Specifc Selection for Genetic Algorithms. *9th World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI)*, 4, 76-81.
34. **Wagner, S. & Affenzeller, M**. **(2004).** *The HeuristicLab Optimization Environment*. Linz, Austria: Johannes Kepler University.
35. **Zhu, K.Q. (2000).** A new Algorithm for VRPTW. *Proceedings of the International Conference on Artificial Intelligence IC-AI 2000*. Las Vegas, USA, 311-320.

Marco Antonio Cruz Chávez *received the Doctor degree in Computer Sciences from Tec de Monterrey in 2004. He works from 2004 like professor-researcher in the Research center in Engineering and Applied Sciences (CIICAP) of the Autonomous University of the Morelos State (UAEM). He is a National Researcher of Mexico (SNI). He has 19 international publications and 11 nationals, from the 2005 he is reviewer of International Journal of Production Research.*

Ocotlán Díaz Parra *is Engineer in Computer Systems, Master in technologies of Information and Doctor in Applied Sciences. She is professor-researcher in the Autonomous University of the Morelos State (UAEM-FCAeI). Their areas of interest are: combinatorial optimization, evolutionary algorithms, software engineering, and computer science security, to mention some. She has given classes in UPEMOR, UAEM and in other diverse Mexican Universities. She has developed projects under contract for PEMEX in the telephony area. For publications and more information see www.diazparra.net.*