A HYPER-HEURISTIC APPROACH FOR MODELING ASSIGNATION PRIORITIZATION RULES ON VRPTW CONSTRUCTIVE HEURISTIC BY NEURAL NETWORKS.

Diego Crespo Pereira[(a)], David del Río Vilas[(b)], Alejandro García del Valle[(c)] , Adolfo Lamas Rodríguez[(d)]

[(a) (b) (c) (d)]Grupo Integrado de Ingeniería, University of A Coruña (Spain)

[(a)]dcrespo@udc.es, [(b)]daviddelrio@udc.es , [(c)]agvalle@udc.es , [(d)]alamas@udc.es

ABSTRACT

Heuristic constructive algorithms have been widely and successfully applied to the solution of routing problems. These algorithms generally consist of a sequential or parallel assignment of nodes to constructed routes. Prioritization rules for assignments are critic for a good performance of the algorithm. This paper presents a hyper-heuristic technique based on a neural network model for prioritization rules which is evolved using an evolutionary search. The technique is applicable to general routing problems, although this paper focuses on the VRPTW. A set of factors specific about the VRPTW for assignment evaluation is introduced. The algorithm is tested with the Solomon instances benchmark with tests oriented to the performance evaluation and the generalization capability of the evolved neural network when solving problems for which has not been evolved.

Keywords: Hyper-heuristic, prioritization rules, VRPTW, neural network.

1. INTRODUCTION

Routing problems constitute one of the most well known and characteristic problems in operations research. They play a central role in logistics due to the relatively high importance of distribution costs in supplying chain (Figliozzi 2007).

In general terms, routing problems deal with finding an optimal or quasi-optimal set of routes for rendering the corresponding services - freight transportation, repair services, passengers transportation, etc.- to a given set of customers or suppliers geographically distributed and connected by a transportation network. But most of routing problems are NP-complete, which causes high computational costs when aiming to find optimal solutions. This has lead to a widespread development of heuristic and metaheuristic approaches, more suitable for real applications.

Places where to render services and pick up or deliver freights are modeled as vertices in a graph where edges are the paths in the underlying transportation network. Different sets of constraints and types of graphs lead to the numerous specific problems in routes optimization. Some of the most classical and studied ones include the TSP, the CVRP or the one in which this work has focused, the VRPTW (Solomon and Desrosiers 1988). An important division between types of routing problems is set up by the dynamic or static nature of the problem (Larsen 2000). A problem is said dynamic if information is not completely known a priory, when the routes construction is initiated, and this information is completed as vehicles cover the routes.

Common approaches to solve routing problems include optimal algorithms, heuristic algorithms and metaheuristics. Optimal algorithms are non-polynomial time solving –and thus not so appropriate for practical applications. Heuristic algorithms exploit appropriate knowledge about the problem features and are able to find almost optimal solutions with very competitive time consumption. However, they have the drawback of depending on problem features and thus it is possible to find instances for which performance is poor. Finally, metaheuristics are usually not as competitive as the best heuristics, but more general in their definition and thus more easily adaptable to problems with different nature.

Currently, main trends in routes optimization research include the definition of new problems closer to applications in real environments (Van Woensel 2008), research of new hybrid algorithms with higher performance and generalization capabilities (Chen 2005) and the improvement of existing solution techniques.

In this paper, a hyper-heuristic technique (Bai 2007) is introduced in order to develop a suitable algorithm for general routing problems in real environments. This technique is applied to the solution of the VRPTW as an initial study to evaluate its feasibility and to identify the main aspects for its future improvement.

2. HEURISTICS CONSTRUCTIVE ALGORITHMS FOR ROUTING PROBLEMS

Construction algorithms for routing problems consist of a sequential construction of the solution routes so that in every step one or a certain number of nodes are inserted into partially obtained routes based on a given criteria. Common criteria might be regarded as the selection of a local optimal insertion among a set of feasible insertions. Every insertion performed induces a change on the feasible insertions on the following steps, so that local optimal insertion decisions do not necessary lead to global optimal solutions. An insertion algorithm that only attempts to select the insertion with lowest increase on total distance of routes is a greedy algorithm. The well known problem caused by this approach is that on the first steps of the algorithm low distance insertions are selected with the eventual cost of leaving many nodes for insertion in inappropriate routes during the last steps of the algorithm.

To face this problem, most of insertion algorithms include criteria to select insertions based on more complex rules. Designed rules should provide an estimate not only for local making decision cost, but an estimation of the impact in global solution. For instance, in VRPTW, waiting time of inserted node is often considered due to the opportunity cost of being waiting to serve a node instead of serving others.

Solomon (1987) developed a heuristic for the VRPTW in which three criteria are used for the insertion selection. These include 1) Latest time for returning to the depot, 2) Lowest gap between service beginning time for a customer and customer closing time and 3) a pondered insertion gain that is based on many factors about the vehicle, node and restrictions.

Faulín and García del Valle (2007) developed an heuristic algorithm for the CVRP in which the criteria followed to sort the assignments to the routes is based on an analogy to the electrostatic charges attraction. Thus, attraction force between nodes is calculated and used to prioritize those with highest attraction.

Generally, a construction algorithm starts with a set of nodes unassigned to routes (or the equivalent situation: assigned to a route in which only that node is serviced). Then, iteratively, nodes are added to construction routes according to the prioritization rules. The order in which assignments are performed is critical since selection implies that other assignments become infeasible in the following steps. For instance, if a node with a high demand is assigned to an almost full route, no other node will be allowed in the same route due to capacity restrictions. Another example is the situation of two nodes A and B for which node C is the closest. If node A is assigned to a construction route that ends in C, then B is not allowed to be assigned to its closest in following iterations.

These prioritization rules are determinant for heuristic algorithms. In this paper we propose the substitution of predefined rules for a parameterized set of rules based on a wide set of factors specific about the routing problem.

3. ALGORITHM DESCRIPTION

In order to describe our algorithm, we first introduce the following notation. This notation is used for the VRPTW

Let $G = \{V, E\}$ be the graph of the transportation network where V denotes the set of nodes. v_0 is the depot. A route is a cycle in V that contains the depot:

$$R = \{v_0, a_{0,1}, v_1, a_{1,2}, \dots v_{|R|-1}, a_{|R|-1,|R|}, v_{|R|}, a_{|R|,0};\ v_i \in V;\ a_{i,i+1} = (v_i, v_{i+1})\}. \quad (1)$$

The distance of an edge will be a function $d: E \to \mathbb{R}$ such that $d_{ij} = d(v_i, v_j)$. No further assumption is made on distances function, thus this algorithm might be either applied to cases with Euclidean, symmetric or asymmetric distances. The distance of a route is the sum of its edges distances:

$$d(R_k) = \sum_{i=0}^{|R|-1} d(v_i, v_{i+1}) + d(v_{|R|}, v_0);\ v_i \in R \quad (2)$$

The travelling time from one node to another is also a function $\text{t}: E \to \mathbb{R}$ such that $t_{ij} = t(v_i, v_j)$. This function has to be positive for every edge, but we do not assume any further restriction or relation with the distance. This let us model routing problems for which the average traveling speed is different for every arc.

Given a route, the sequence of nodes visited from the depot defines an arrival (*at*) and a departing (*dt*) time for each node depending on its opening time (*ot*) and service time (*st*) as follows:

$$at_i = dt_{i-1} + t_{i-1,i} \quad (3)$$

$$dt_i = \max\{at_i + st_i, ot_i\} \quad (4)$$

A solution to the routing problem is a set of routes S that verifies the following constraints:

1. $\forall v \in V \to \exists R \in S$ such that $v \in R$.
2. $R_i \cap R_j = \{v_0\}\ \forall i \neq j$.
3. The load of a vehicle cannot exceed its capacity C. In this paper, we assume all the vehicles to have the same capacity. Thus, $\forall R_k \in S \to \sum_{i=1}^{|R|} q_i \leq C; q_i \in R_k$.
4. The arrival time to a node cannot exceed its closing time (*ct*). $at_i \leq ct_i\ \forall\ i \in S$.

Then, the VRPTW aims at finding the solution S^* that minimizes the sum of distances of its routes. Let Ψ be the set of all the solutions to the problem. Thus:

$$\sum_{R \in S^*} d(R) \leq \sum_{R \in S} d(R)\ \ \forall\ S \in \Psi \quad (5)$$

An assignment is defined as a pair node-route. $A_{ki} = A(R_k, v_i) = \{R_k, v_i | v_i \notin R_k\}$. Every assignment defines an extended route of R_α as the route formed by adding the node at the end of the route. $R'_\alpha = \{v_0, v_{k1}, \dots, v_{kn}, v_i; v_{kj} \in R_k\}$. The cost of an assignment is a function that represents the difference between the distance of the initial and extended routes. $CF(A_{\alpha i}) = d(R'_\alpha) - d(R_\alpha)$. Every route can thus be obtained as a sequence of assignments starting from a route that only contains the depot.

$$R_\alpha = \{v_0, v_{\alpha 1}, \dots, v_{\alpha n}\} = \{\ A(\emptyset, v_{\alpha 1}), \dots, A(\{v_{\alpha 1}, \dots, v_{\alpha n-1}\}, v_{\alpha n})\ \} \quad (6)$$

An assignment is said feasible if the extended route verifies the problem constraints. It can be easily proved that a route is feasible if and only if all the assignments necessary to obtain it are so.

Since every route can be decomposed into a sequence of assignments, a solution can be also regarded as a set of feasible assignments. A partial solution is a set of assignments such that at least one solution exists containing all of them. The feasibility of an assignment given a partial solution depends on the latter one.

The constructive algorithm that we propose consists of an iterative performance of assignments such that the obtained solution is intended to have a distance close to the optimal solution S^*. On the *n*-th step, there is a set of feasible assignations (FA_n) that can be added to the temporary partial solution constructed and one of them has to be performed. Thus, the key of the algorithm are the rules followed to determine which one of the feasible assignments is chosen. These prioritization rules can be modeled as a function of a set of factors that characterize the assignment: $p_{ki} = PR(f_1(A_{ki}), \ldots, f_{NF}(A_{ki}))$, where $f_j(A_{ki})$ is each one of the assignment factors and *NF* the total number. Ties in priority might be broken randomly or by a proper selection of the model.

In order to increase algorithm's speed, the priority is not calculated for all the feasible assignments. A first selection of studied assignations (SA_n) is made according to one of the factors, the selecting factor (SF). Only the *NK* assignments with highest SF to each route in the partial solution are selected for prioritization.

The procedure is the following, where:

Let L_0 be the set of not yet assigned vertices.
PS is the partial solution.

1. $PS = \{v_0\}$.
2. $L_0 = V - \{v_0\}$.
3. Obtain $SA_{n,1}$.
4. $j = 1$.
5. Calculate priorities for routes in $SA_{n,j}$ by the prioritization model.
6. Search the assignation $A_{i^*} \in SA_{n,j}$ such that $p_{i^*} \geq p_i\ ; A_i \in SA_{n,j}$.
7. If the route $R_{i^*} \in A_{i^*}$ only contains the depot $R_{i^*} = \{v_0\}$ then create a new route containing only the depot.
8. Add vertex $v_{i^*} \in A_{i^*}$ to route R_{i^*}.
9. Remove v_{i^*} from L_0.
10. Remove from $SA_{n,j}$ all the assignments that contain either the node or route of A_{i^*}: $SA_{n,j+1} = SA_{n,j} - \{A_{k,i} | (k = k^*) \cup (i = i^*)\}$.
11. If $j \leq ND$, then go to step 4 and make $j = j + 1$. Otherwise continue.
12. If $L_0 \neq \emptyset$ return to 3.

A group of assignment factors has been defined based on VRPTW properties judged relevant. In order to introduce these parameters, we first introduce the following notation:

Given a sorted list L and its element e_i, then the order of the element is its position in the list sorted from minimum to maximum and it is noted as $O(e_i, L)$.

The assignation with subscript α for which the parameters are defined is the pair route-node (R_k, v_i).

The route head node (RH) is the last node of a given route.

The direct gain of an assignation is the gain obtained by performing an assignation compared to the second best possible assignation to the route. Thus:

$$dg_{k,i} = \min_{\mathrm{j}}\{d_{RH_k,j}; d_{RH_k,j} \neq \min_{\mathrm{l}}\{d_{RH_k,l}\}\} - d_{RH_k,i} \quad (7)$$

The total gain of an assignation is the direct gain obtained by performing an assignation compared to the second best direct gain to assign the node.

$$g_{k,i} = dg_{k,i} - \max_{\mathrm{j}}\left\{dg_{RH_j,i}; dg_{RH_j,i} \neq \max_{\mathrm{l}}\{dg_{RH_l,i}\}\right\} \quad (8)$$

The normalization waiting factor (WNF) is used to normalize waiting times and can be set up according to characteristic time units in the problem.

Direct degree of a node δ(v_i) is the number of feasible assignations to a route ended by it.

Inverse degree of a node δ'(v_i) is the number of feasible assignations to a route ended by it.

The selection factor used in our algorithm is the total gain. Thus, in the SA_n calculation step, the total gain is evaluated for each assignation in (FA_n). For each route in the partial solution, the first *NK* assignations with highest total gain are added to SA_n. The nodes that can only be accessed from the depot are also added to SA_n.

The rest of the factors for the prioritization rules model have been chosen with the aim of providing enough information about the assignment to make possible good prioritization rules. The parameters are transformed into normalized values between 0 and 1. Thus they do not depend on problem size or units.

For the VRPTW we employ the following sixteen factors:

1. Relative total gain between the maximum and the minimum gain for all the selected assignments. If no more than one assignment is feasible for a node or a route, its value is assumed to be 1. Expected behavior for evolved rules is to prioritize routes with high gains. In case of no difference between top and lowest gain, this factor equals 1.

$$RTG_{k,i} = \frac{g_{k,i}}{\max_{\mathrm{A \in LA}} \mathrm{g(A)}} \quad (9)$$

2. Relative total gain to route. It is the relative total gain between the maximum and the minimum gain

for all assignments which correspond to the same route as evaluated assignment. If no more than one assignment is feasible for the route, its value is assumed to be 1. Expected behavior for evolved rules is to prioritize routes with high gains. In case of no difference between top and lowest gain, this factor equals 1.

$$\mathrm{RGR}_{k,i} = \frac{g_{k,i}}{\max_j g_{k,j}} \tag{10}$$

3. Waiting factor at the visited node. It is defined to prioritize assignments that do not cause high waiting times in the route.

$$\mathrm{wt}_{k,i} = \begin{cases} \frac{1}{1+\frac{to_i - ta_i}{\mathrm{WNF}}} \\ 1 \text{ otherwise} \end{cases} \tag{11}$$

4. Arrival time factor. The aim of this factor is to provide different prioritization rules based on whether is a node early assigned to route or lately.

$$\mathrm{TAF}_{k,i} = \frac{ta_i}{\max_j tc_j} \tag{12}$$

5. Centrality factor. It is the relative distance from node to depot related to the highest distance to depot.

$$\mathrm{CNF}_i = \frac{d_{0,i}}{\max_j d_{0,j}} \tag{13}$$

6. Centrality order. It is the position in the ordered list of distances to depot. Together with centrality factor is expected to provide combined prioritization rules as a function of the proximity to depot compared to the rest of nodes in the problem.

$$\mathrm{CNO}_i = O\left(d_{0,i}, \{d_{0,j}; j \in 1, \dots, N\}\right) \tag{14}$$

7. Closing time factor. It is the relative closing time compared to the latest closing time in the problem.

$$\mathrm{CTF}_i = \frac{tc_i}{\max_j tc_j} \tag{15}$$

8. Closing time order. It is the position in the ordered list of closing times to depot. Together with closing time factor is expected to provide combined prioritization rules as a function of the closing time compared to the rest of nodes in the problem.

$$CTO_i = O\left(ct_i, \{ct_j; j \in 1, \dots, N\}\right) \tag{16}$$

9. Inverse degree factor. It is defined as the fraction of non assigned nodes from which assignment node can be accessed.

$$\mathrm{IDF}_i = \frac{\delta\prime(i)}{N} \tag{17}$$

10. Direct degree factor. It is defined as the fraction of non assigned nodes that can be accessed from assignment node.

$$\mathrm{DDF}_i = \frac{\delta(i)}{N} \tag{18}$$

11. Assignment demand factor. It is defined as the degree of completeness of the vehicle capacity if the node was assigned to the route.

$$\mathrm{ADF}_{k,i} = \frac{\sum_{j \in R_k} q_j + q_i}{C} \tag{19}$$

12. Assigned rate. It is the fraction of nodes that have already been assigned at the considered step. It is expected to provide different rules depending on the algorithm completion rate.

$$\mathrm{AR} = \frac{|L_0|}{N} \tag{20}$$

13. Centrality factor of route. It is the relative distance from route head to depot related to the highest distance to depot.

$$\mathrm{CNFR}_{CR_k} = \frac{d_{0,CR_k}}{\max_j d_{0,CR_k}} \tag{21}$$

14. Centrality order of route. It is route head position in the ordered list of distances to depot.

$$\mathrm{CNOR}_{CR_k} = O\left(d_{0,CR_k}, \{d_{0,CR_k}; j \in 1, \dots, N\}\right) \tag{22}$$

15. Route load factor. It is the fraction of vehicle capacity already served by the route.

$$\mathrm{RLF}_k = \frac{\sum_{j \in CR_k} q_j}{C} \tag{23}$$

16. Direct degree fraction of the route. It is the fraction of non assigned nodes that can be accessed from the route.

$$\mathrm{DDF}_k = \frac{\delta(CR_k)}{N} \tag{24}$$

Our approach consists of defining a parameterized prioritization function model which to optimize. Thus the routing problem is turned into a parameters (γ) optimization problem which is solved by means of a search algorithm. The objective function for the search algorithm is the total distance of the solution obtained for a given set of parameters.

$$p_{ki} = \mathrm{PR}(f_1(A_{ki}), \dots, f_{16}(A_{ki}), \gamma_1, \dots, \gamma_{NP}) \tag{25}$$

NP is the number of parameters in the model.

It can be defined a function of the parameters on the solutions set such that:

$$d(\gamma_1, \dots, \gamma_{NP}) = d\left(S(\gamma_1, \dots, \gamma_{NP})\right) \tag{26}$$

This is the function we attempt to minimize by finding the parameters vector.

4. PRIORITIZATION RULES MODEL

The model considered for the prioritization rules is a neural network. Specifically, the neural network used is a multilayer perceptron with sigmoid activation function due to its capability to approximate every real function, including non continuous functions if it has more than one hidden layer (Hornik 1989). The multilayer perceptron used has one hidden layer with 4 neurons. This means that a set of 78 parameters are needed to define the net weights, bias and slopes. Maximum and minimum weight values and bias are limited to 8 and -8 respectively. The slopes are in the range from 0.2 to 1.8.

In order to optimize neural network parameters an evolutionary strategy is used. Genes in the chromosome are the net weights, bias and slopes. Also the number of assignments selected for each route (*NK*) and the number of performed assignments (*ND*) are included in the chromosome. The objective function is the total distance calculated solving a problem or a set of problems. Notice that once a priority model is evolved, it can be used to solve problems different to the evolving set. The capability to obtain good solutions when solving problems different to the evolving set is the most desired feature of this technique.

Two search algorithms have been used in this work. One is the Differential Evolution strategy as defined in (Rainer Dtorn and others, 1997). The parameters used for the evolution are F = 0.7 and CR = 0.3. Since it is an algorithm to evolve genes being real numbers, ND and NK are coded as real numbers between -1 and 1. Then they are transformed into integer values by setting top and lowest values.

The other search algorithm employed is the Macroevolutionary Algorithm (MA) developed by J. Marín and R. V. Solé (Marín 1999). Both evolutionary strategies are compared and the MA is selected due to its highest performance.

Each objective function for the evolutionary algorithm is the distance of the solution to a VRPTW instance by using the algorithm described in the previous section. In each evaluation, the neural network employed is the given by the individuals chromosome coding.

5. EXPERIMENTATION

The algorithm has been coded in JAVA and makes use of the Evolutionary Algorithm Framework Library and the Artificial Neural Network Library developed by the Grupo Integrado de Ingeniería of the University of A Coruña.

As benchmark of problems to evaluate algorithm's performance, we employ the Solomon's Instances. In order to test the algorithm's capability to find rules suitable for many problems, a subset of these instances was used for evolving the neural networks. It is the set of problems derived from the Solomon's benchmark by selecting only the first 26 nodes in each one. As benchmark for testing the neural networks evolved, we employ the 26, 51 and 101 nodes instances. The best solutions to this benchmark can be found in Solomon's website. The performance measure employed is the relative increase in distance to the optimal solution.

Four types of tests have being conducted in order to test algorithm's performance. The first is to compare both evolutionary strategies by evolving a neural network for each problem in the Solomon's 26 nodes instances. Results presented in table 1 show that the MA's performance is higher. In both cases the stop criteria for the algorithm is the same: 10,000 objective function evaluations as a top. The difference is maximum on random cluster type problems and minimum on cluster.

Table 1: Comparison between MA and DE.

26 Nodes Solomon's Instances				
Problem	**MA**	**DE**	**Rel. Diference**	**Max. Rel Diference**
R1	5,995	6,247	4.20%	8.53%
R2	4,615	4,938	7.00%	11.38%
C1	1,759	1,768	0.51%	4.31%
C2	1,733	1,743	0.58%	3.70%
RC1	2,989	3,106	3.90%	12.74%
RC2	2,846	3,191	12.13%	27.38%
Total	***19,937***	***20,992***	***5.29%***	***27.38%***

The second test attempts to find the best solution that can be obtained by evolving a neural network for each problem. Thus, a neural network is evolved for every Solomon's instance and compared to the optimal solution. This is accomplished for 26 nodes instances by means of the MA and setting the maximum number of objective function evaluations to 60,000. The population size is 1,500 and the number of generations 40. Results obtained are good since *quasi*-optimal solutions are found (table 2). Algorithm's performance is higher for cluster type problems than for random ones.

Table 2: Best solutions found with the MA.

26 Nodes Solomon's Instances				
Problem	**Optimal Distance**	**Distance**	**Rel. Diference**	**Max. Rel Diference**
R1	5,560	5,788	4.09%	8.75%
R2	4,204	4,439	5.60%	11.52%
C1	1,716	1,733	0.99%	4.84%
C2	1,716	1,733	1.00%	4.30%
RC1	2,802	2,872	2.51%	3.86%
RC2	2,554	2,690	5.33%	10.54%
Total	***18,552***	***19,256***	***3.79%***	***11.52%***

The third test consists of using the best neural networks obtained for the 26 nodes instances and apply them to solve the 51 and 101 cases. Thus the generalization capability of the evolved neural networks for solving bigger problems is checked. The results are shown in tables 3 and 4 and show as the average relative distance to the optimum increases with the problem size. The obtained solutions are bad compared to the solutions achieved for the problem size that they have been evolved for. However, these problems are rather different and hence this result might be expected.

Table 3: Best neural networks for the 26 nodes case applied to 51 nodes instances.

51 Nodes Solomon's Instances				
Problem	**Optimal Distance**	**Distance**	**Rel. Diference**	**Max. Rel Diference**
R1	9,194	11,768	28.00%	53.83%
R2	6,967	9,219	32.33%	51.39%
C1	3,255	4,853	49.10%	147.62%
C2	2,860	4,063	42.07%	71.48%
RC1	5,852	8,464	44.65%	71.18%
RC2	4,671	7,350	57.36%	78.97%
Total	***32,798***	***45,717***	***39.39%***	***147.62%***

Table 4: Best neural networks for the 26 nodes case applied to 101 nodes instances.

101 Nodes Solomon's Instances				
Problem	**Optimal Distance**	**Distance**	**Rel. Diference**	**Max. Rel Diference**
R1	14,146	19,851	40.34%	64.80%
R2	10,362	13,649	31.72%	45.70%
C1	7,440	11,621	56.18%	121.43%
C2	4,699	7,225	53.75%	116.35%
RC1	10,731	16,857	57.08%	110.81%
RC2	8,392	12,325	46.87%	81.10%
Total	***55,770***	***81,527***	***46.19%***	***121.43%***

The fourth test performed consists of evolving the neural network for each instance with a randomly modified version of it. All the node coordinates, demand values, opening times, service times and closing times are randomly modified with an uniform distribution between its value minus and plus a specified factor. Then the evolved neural network is applied to solve the original problem and thus its capability to solve similar problems can be studied. The factor levels for the modification are set between -5% to 5% and -10% to 10%. The results are shown in tables 5 and 6. The increase in relative gap to optimal solution increases with the modifying level. The performance becomes worse, but it should also be noted that the modified problems are very different to the original ones, since the modifications in coordinates, demands and time windows change the feasible edges between nodes.

Table 5: Neural networks evolved for Solomon's instances randomly modified by a 5%.

26 Nodes Solomon's Instances				
Problem	**Optimal Distance**	**Distance**	**Rel. Diference**	**Max. Rel Diference**
R1	5,560	7,169	28.93%	46.76%
R2	4,204	5,736	36.47%	60.93%
C1	1,716	2,241	30.58%	73.96%
C2	1,716	2,219	29.36%	69.26%
RC1	2,802	3,925	40.09%	95.04%
RC2	2,554	3,622	41.82%	122.80%
Total	***18,552***	***24,914***	***34.29%***	***122.80%***

The neural networks for the modified problem are evolved with the MA, a population size of 1,500 individuals and 40 generations.

Table 6: Neural networks evolved for Solomon's instances randomly modified by a 10%.

26 Nodes Solomon's Instances				
Problem	**Optimal Distance**	**Distance**	**Rel. Diference**	**Max. Rel Diference**
R1	5,560	7,431	33.63%	55.93%
R2	4,204	5,783	37.58%	52.12%
C1	1,716	2,647	54.25%	140.92%
C2	1,716	2,533	47.66%	138.75%
RC1	2,802	4,264	52.15%	113.41%
RC2	2,554	3,847	50.61%	84.50%
Total	***18,552***	***26,505***	***42.87%***	***140.92%***

6. CONCLUSIONS

A hyper-heuristic algorithm with complex rules for the prioritization of assignments has been introduced for the VRPTW and tested with the standard benchmark Solomon's instances. This algorithm might be either applied to cases with Euclidean, symmetric or asymmetric distances. It also allows model routing problems for which the average traveling speed is different for every arc. The Macroevolutionary Algorithm (MA) has shown a good performance evolving the rules modeled by a multilayer perceptron.

The results are good when solving problems for which the rules have been evolved with the MA. It is able to reach *quasi*-optimal solutions with the advantage that the found solution is not just a set of routes that minimize the distance, but the neural network which contains the constructive routes that lead to them.

Another advantage of the introduced algorithm is that, since few assumptions have been made on problem constraints, it might be applied to VRPTW variations without big effort on adaptation expected.

However, the generalization capabilities of the evolved rules have shown to be poor either at solving problems with bigger size or randomly modified from the original instance. Input factors for the neural network are critic on this matter. If the factors values present different ranges on different instances, the rules evolved for one will perform badly when applied to another. Hence, the proposed group should be carefully studied and improved in order to enhance these results.

ACKNOWLEDGMENT

We wish to express our gratitude to the Xunta de Galicia, which has funded this work through the research project "Análisis y desarrollo de una aplicación para la gestión, planificación y optimización de rutas de transporte en entornos dinámicos" (PGIDIT06DPI000PR10).

REFERENCES

Bai, R., Blazewicz, J., Burke, E.K., Kendall G., McCollum, B. 2007. A simulated annealing hyper-heuristic methodology for flexible decision support. Technical report, School of CSiT, University of Nottingham, UK.

Chen, A., Yang, G., Wu, Z. 2005. Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. Journal of Zhejiang University - Science A, 7 (4), 607-614.

Faulin, J., García del Valle, A. 2007. Solving the capacitated vehicle routing problem using the ALGELECT electrostatic algorithm. Journal of the Operational Research Society, 1-11.

Figliozzi, M.A., Mahmassani, H.,S., Jaillet, P. 2007. Pricing in Dynamic Vehicle Routing Problems. Transportation Science, 41 (3), 302-318.

Hornik, K., Stinchcombe, M., White, H. 1989. Multilayer feedforward networks are universal approximators, Neural Networks, 2, 359-366.

Larsen, A., 2000. The Dynamic Vehicle Routing Problem. Ph.D.-thesis. IMM-PHD-73, IMM.

Marín, J., Solé, R.V. 1999. Macroevolutionary Algorithms: A New Optimization Method on Fitness Landscapes. IEEE transactions on evolutionary computation, 3 (4).

Rainer, D., Kenneth, P. 1997. Differential Evolution. A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, 11, 341–359.

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

Solomon, M.M., Desrosiers, J. 1988. Time window constrained routing and scheduling problems. Transportation Science, 22(1), 1- 13.

Van Woensel, T., L. Kerbache, L., Peremans, H., Vandaele, N. 2008. Vehicle routing with dynamic travel times: A queueing approach. European Journal of Operational Research, 186 (2008), 990-1007.

AUTHORS BIOGRAPHY

Diego Crespo Pereira holds an MSc in Industrial Engineering (Ingeniero Industrial) and he is currently studying for a PhD. He works full-time in the GII of the University of A Coruna as a research engineer since 2008. He is mainly involved in the development of R&D projects related to industrial and logistical processes optimization. He is especially interested in the field of dynamic vehicle routing problems.

David del Rio Vilas holds an MSc in Industrial Engineering (Ingeniero Industrial) and has been studying for a PhD since 2007. He works full-time in the research group GII of the University of A Coruna as a research engineer since 2007. He is mainly involved in R&D projects development related to industrial and logistical processes optimization. His area of major interest is the use of simulation in industrial and logistical environments, especially in maritime container terminals.

Alejandro Garcia del Valle is an Industrial Engineer by the Polytechnic University of Madrid and holds a PhD in Industrial Engineering. He is Professor and one of the founders of the GII at the University of A Coruna. He has led many research projects about simulation and optimization of industrial, logistical and service systems.

Adolfo Lamas Rodriguez graduated from the University of Vigo in 1998. He holds an MSc and a PhD in Industrial Engineering. He combines his research activities in the GII and his position as a senior engineer in the Spanish leading shipbuilding company *Navantia*. He is also Associate Professor in the University of A Coruna.