Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. IV (2009), No. 2, pp. 127-136

Application of Genetic Algorithms for the DARPTW Problem

Claudio Cubillos, Enrique Urra, Nibaldo Rodríguez

Pontificia Universidad Católica de Valparaíso
Escuela de Ingeniería Informática
Av. Brasil 2241, Valparaíso, Chile
E-mail: claudio.cubillos@ucv.cl, enrique.urra.c@mail.ucv.cl

Abstract: On the Dial-a-Ride with time windows (DARPTW) customer transportation problem, there is a set of requests from customers to be transported from an origin place to a delivery place through a locations network, under several constraints like the time windows. The problem complexity (NP-Hard) forces the use of heuristics on its resolution. In this context, the application of Genetic Algorithms (GA) on DARPTW was not largely considered, with the exception of a few researches. In this work, under a restrictive scenario, a GA model for the problem was developed based on the adaptation of a generic GA model from literature. Our solution applies data pre-processing techniques to reduce the search space to points that are feasible regarding time windows constraints. Tests show competitive results on Cordeau & Laporte benchmark datasets while improving processing times.
Keywords: Dial-a-ride, Passenger Transportation, DARPTW, Heuristic, Scheduling.

1 Introduction

In the research of transport systems, the Dial-a-Ride Problem (DARP) or the customers' transportation problem is largely known [3]. It consists on searching the optimum way to transport a set of customers which are territorially distributed through a locations network, considering diverse constraints, for example the vehicle capacity and the time windows (TW) which are time intervals where a customer can be picked or delivered on the respective location, in a feasibility context.

The problem objective is to optimize the transport system factors (vehicles number, travel costs) and the quality of service for customers (waiting time, travel time). DARPTW (the time windows problem version) is considered a NP-Hard problem, especially because of the time-window constraints [10]. For this reason, the problem is usually solved through heuristics to find good solutions, under its diverse variants. In fact, the time windows restrictions make the problem highly non-convex, making it difficult to find feasible solutions.

One of the tools considered in this context are the Genetic Algorithms (GAs). After their comparison in the scientific scenario thanks to John Holland on the 70's decade [9], these algorithms have been successfully accepted by their efficiency to solve problems of diverse complexity, and to date there is a large number of proposed GA models that considers the canonical GA problems, particularly the linkage concept [8].

In this work, the application of GAs on DARPTW is extended considering two elements: on the one hand, the implementation of the LLGA model [8] with and adequate adaptation in the context and on the other hand, the use of data pre-processing techniques (namely precedence table of events and incompatible clients' list) for aiding the GA to avoid infeasible solutions from the time windows perspective.

The paper is structured as follows. Section 2 explains the DARPTW problem for then in section 3 tackling other research in the field. Section 4 details the implemented GA and section 5 the experiments and its results. The main conclusion of the work are drawn in Section 6.

Copyright © 2006-2009 by CCC Publications

2 The DARPTW Problem

DARPTW is a multiple objective optimization problem, because there are two critical factors to be optimized: on the one hand the total transportation costs and on the other, the quality of service offered to customers (minimizing their insatisfaction with the service). There is a set of transportation requests from customers that are known in advance and do not change during algorithm execution, defining the problem as static. Each request defines a time window for the customer delivery and a time window for the customer pickup. The upper bound (Latest Time) and the lower bound (Early Time) of the time window are supplied. A solution is considered infeasible when the vehicle arrives outside the time window bounds, defining these as hard time windows.

To execute the service, there is a homogeneous vehicles set with the same load capacity that cannot be exceeded. The passengers are picked and delivered by the same vehicle. A vehicle can enter on inactivity times or slacks only without passengers on board. Additionally, there is only one depot (single depot), a particular location where vehicles start and end their travels. In this work, maximum route duration is not considered.

3 Related Work

As mentioned previously, there are only few researches where the GAs are considered for DARPTW. There are more works which provide relevant elements in this context on VRP (Vehicle Routing Problem), a generic case of DARP. An example is Thangiah's work[11], which describes GIDEON, a GA based heuristic to solve VRP with time windows. This mechanism uses a cluster first - route second strategy, starting with the assignation of customers to vehicles and after improving the best solution by a post-optimizing process. For the system implementation, a GA software called GENESIS is used, where the individuals are represented by bit strings. The client clusters/sections are obtained from an individual by splitting him on K divisions of B bits. Each division is used to compute the size of a sector. The individual quality is obtained through the cost function evaluation when all computed clients are served, regarding their derived sector divisions. In this work, B = 3 is used, bigger values showed less satisfactory results. For the testing, the parameter values for the population size, crossover rate and mutation rate were 1000, 0.5 and 0.001 respectively. A set of 56 instances were tested using the Solomon's benchmark data, widely known in this context. In the results, 41 instances showed improvement in comparison to other heuristics developed by Solomon and Thompson.

Regarding DARPTW, in [1], a cluster first - route second strategy is also used. The mathematic model used in their work is a generalization of the one used in this work. This generalization is justified by the use of soft time windows, so the objective function considers additional elements related to quality of service. The individual is based on client clusters. A matrix is used, where the row number equals the available vehicles and the columns equals the clients and depots number. If an element on the matrix equals to 1, then the respective client is assigned to the respective vehicle. Additionally, a row (vehicle) represents a specific route. Because the absence of standardized benchmark data set in DARPTW, in contrast to VRP case, a set developed by Cordeau & Laporte is used [5], that contains 20 random instances. This set considers instances from 24 to 144 customers. The obtained results are compared also with a Cordeau & Laporte research [4], with similar improvement.

Finally, there is the Cubillos work [6], where the objective is to develop a specific GA model to solve the DARPTW problem, considering it as a deceptive problem, using the GA to solve the full problem, in contrast to a universal solver GA and other researches where the GA solves only a part of the problem. To accomplish this, an adequate framework was developed, that considers all critical GA elements. At the same time, the considered DARPTW instance was very specific in contrast to other studies, for example, there were only outbound customers. A bus-passenger representation for each gene on an individual was used. The solution decode is done by an ordered lecture, where the first occurrence of a customer is

always a pickup and the second one is always a delivery, this obligates the representation to consider only two genes per customer. The vehicle associated with the pickup is the one actually considered in the solution and the one of the delivery can be different but not considered. The final results showed an improvement on the solution quality factor, in contrast to the vehicle quantity, when compared to previous research.

It is important to highlight that none of the above solutions considers any technique for improving the search over the feasible solution space, especially regarding the time-windows constraints. This is especially true when applying the crossover and mutation operators, reason why the present work improves actual GA solutions by trying to avoid or minimize the infeasibilities and subsequent reparations after the operators through the use of pre-feasibility tables or schemas.

4 Implementation

In this section, the implemented GA framework and each of its elements will be described in the following.

4.1 Preliminary feasibility schemas

As stated before, the time window constraints make the problem search space highly non convex, meaning that it is very easy to move from a feasible solution to an unfeasible one when searching through meta-heuristics, being the only solution to roll-back or to repair the unfeasible solution.

In the present work an approach has been developed to minimize these problems by reducing a priori the feasible planning combinations considering the time windows restrictions.

When processing the dataset of requests, a preliminary precedence table of events is built, where each event (pickup or delivery) is associated a list of other events which need to be inserted before that event on a route (if assigned to the same vehicle) to preserve time windows feasibility. In Figure 1, for example, are shown 4 clients (A, B, C and D) with their time windows for pickup ($+$) and delivery ($-$). An obvious relation is that the pickup $A+$ must precede its delivery $A-$. Then, the pickup of C ($C+$) cannot be before the pickup of a ($A+$) as it would be impossible for the same vehicle to serve both events within their time window bounds as the last ends before the other stars.

In the most general case, an event Y must precede an event X when $ET_X + DRT_{XY} > LT_Y$, where ET_X corresponds to the Early Time (time windows lower bound) of event X, DRT_{XY} the direct ride time from the location of X to the location of Y and LT_Y to the Latest Time (time windows upper bound) of event Y. In this way, it is possible to build a list of precedence events for each event.

On the other hand, there is an incompatible clients' list that defines, on a level of passenger assignation (clusters), which clients cannot be transported by the same vehicle, for the time windows feasibility of a solution. This list can be obtained from the pairs of events that precede each other simultaneously: if an event X is preliminary to Y and at the same time Y is preliminary to X, then it is impossible for both to be part of the same route and, consequently, its respective clients cannot be transported by the same vehicle.

In practical terms, the case involved is when a couple of events are too far the one from the other and their time windows are too close to each other making it impossible for a single vehicle to go from one location to the other within their time windows constraints.

4.2 Initial Population Generation

It is based on the client incompatibility previously described. The mechanism is concerned that incompatible clients will not be assigned to the same vehicle, first generating a base feasible solution that

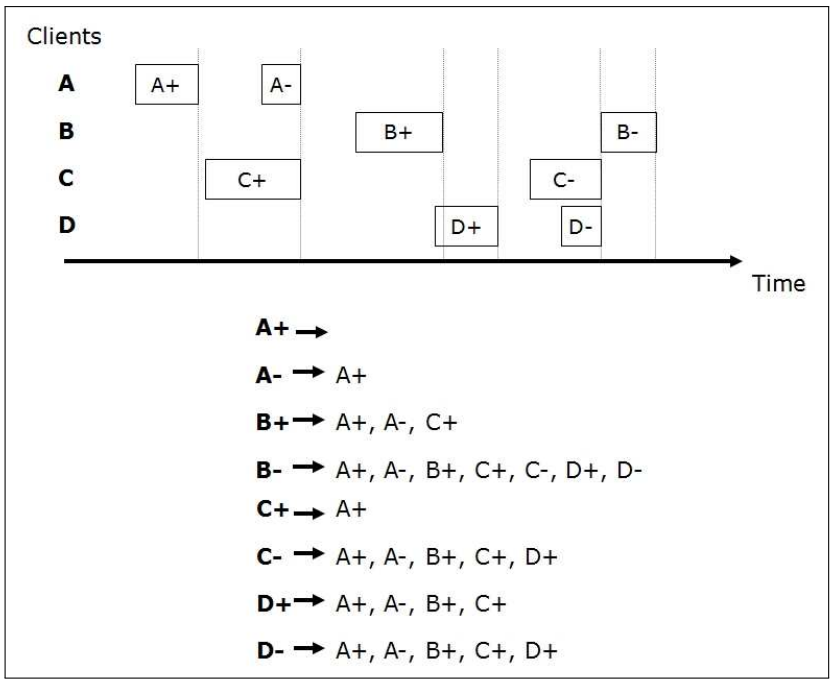

Figure 1: Example of preliminary precedence table of events, where each event has other events that must precede it when they are on the same vehicle

contains all conflicting clients on separated vehicles and afterwards generating a final feasible solution as a result of the insertion of the remaining clients over the base feasible solution.

The implemented mechanism does not assure the generation of a feasible solution on the first try. In this context, the randomness on the insertion heuristic and the vehicle selection for the clients, facilitates the possibility of restart the process when the incapacity of continue generating a feasible solution is detected.

4.3 Genotype and Crossover

A model like LLGA [7] has been considered. The incorporation of the locus on the genes allows to order them in different ways representing an identical solution. A gene is composed by the locus, that corresponds to the client number on this implementation, and the vehicle assigned to him. Figure 2 shows the clients A, B, C, D and the vehicles V1, V2, V3 to which are assigned.

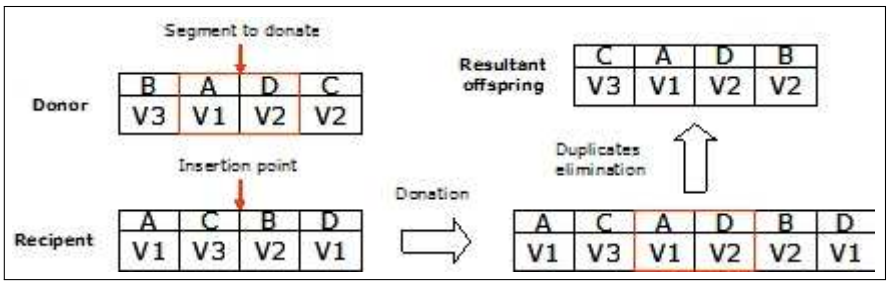

Figure 2: Crossover operation for the implemented genotype

This representation tackles the assignment of clients to vehicles (clustering) while the scheduling itself (route construction) is carried out through a greedy insertion heuristic explained in the next subsection.

On crossover, (see Figure 2) a father assumes the donor role, giving an own gene segment of its structure, on a random insertion point of the other father that assumes the recipient role, where the duplicated genes are deleted. In this case, the duplicated clients from the recipient are deleted. This generates a change on the clusters from the chromosome external layer, but also triggers the modification of the routes from the chromosome internal layer.

Because this process is not exempt from unfeasibility problems, in the basis of the randomness of the internal processes that support the crossover (insertions, eliminations, etc.) and the own crossover factors (insertion point, donated segment), a restart of the whole process is considered, until a feasible crossover has been done.

4.4 Route Scheduling

It regards the greedy insertion heuristic used for route generation, plus feasibility evaluation procedures. This mechanism is based on "MADARP" model shown on [6], applying the concept of time windows intersection and the use of pre-calculated data for a direct evaluation.

Figure 3 (a) shows a portion of a schedule, containing the pickups and deliveries of clients A and B on a first block, then a slack (that is, vehicle idle time without passengers onboard) and a second block serving clients C and D. Within the first block, the evaluation of client X is carried out by evaluating all the possible permutations of the new client in the sequence, that is, (n+1)(n+2)/2 with n the number of events already present in the block.

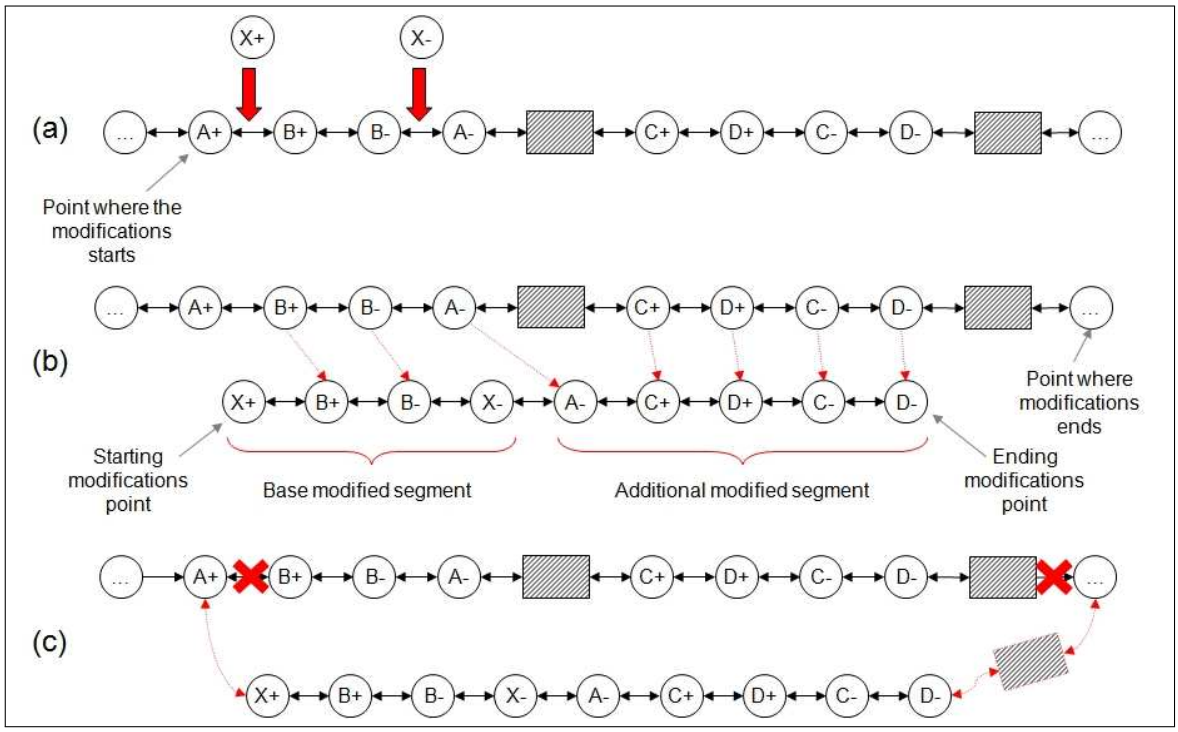

Figure 3: Crossover operation for the implemented genotype

In addition, the heuristic also considers the possibility of merging blocks as Figure 3 shows. The insertion of client X makes the slack time after $A-$ to disappear, resulting in a bigger block.

4.5 Phenotype or Evaluation

It is worth highlighting that there are two generic elements evaluated on a solution: The transport system efficiency and the quality of service offered to the clients. On the first element, the considered factors are the vehicle travel time, the slacks length and the vehicle quantity. On the second element, the excess ride time and the wait time are considered. A weight is assigned to each of these factors that directly influence the evaluation result.

4.6 Selection

For the proposed models, a tournament selection is used, from where a population subset is obtained, and an individual from these are selected as a winner. On this operator, the selection pressure can be managed through the tournament size. In this implementation, when a winner is obtained from the tournament, it is not removed from the original population, allowing them be the winner on future tournaments and facilitating the convergence.

4.7 Mutation

In contrast from other models described in literature, two mutation operators have been developed in this implementation: a cluster mutation and a route mutation. Each of these has its own probability, hence an individual can be affected by one, both or none of them on the generation advance. The cluster mutation consists in moving a client from one vehicle to another and the probability is applied to each client in the solution as Figure 4 (a) shows.

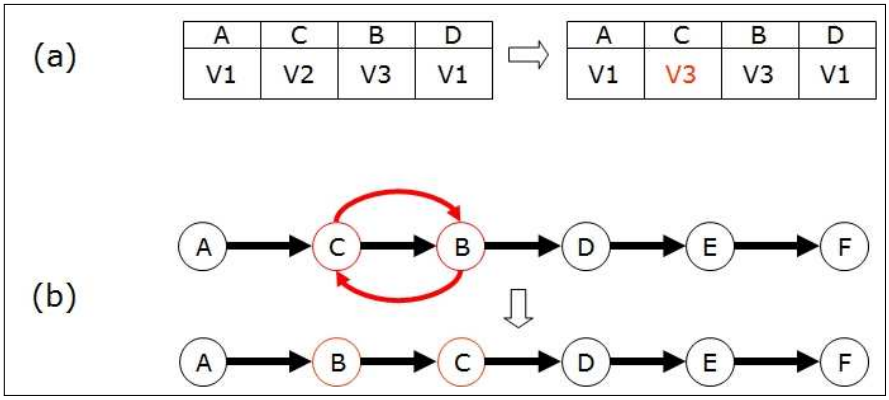

Figure 4: Crossover operation for the implemented genotype

The route mutation consists on interchanging the order of an event's pair from the route and the probability is applied to each route in the solution (see Figure 4 (b)). Because this operator is not exempt from unfeasibility problems and the mutation probabilities are small, when all possibilities has been evaluated and a feasible result has not been obtained, the process is discarded for a client or a route according to the respective mutation.

5 Experiment Settings

In literature, a common benchmark dataset used to evaluate heuristics for the DARPTW problem corresponds to the ones provided by Cordeau & Laporte [5]. These sets are divided on two subsets: One is used in [3] for an Branch-and-Cut algorithm, while the other one is used in [4] for a Tabu Search algorithm and in [1] for a cluster-first route-second GA. These are used on this research both for the GA

calibration stage and final testing stage. For the evaluation, the benchmark data was divided on three groups according to the number of clients: small sets (pr01, pr02, pr11, pr12 and pr17), medium sets (pr03, pr05, pr15 and pr19) and big sets (pr16).

On the calibration stage, the following parameters were considered: 3 sets (a small one, a medium one and a big one); Population size: 100; Maximum generation number: 10000; Crossover ratio: 0.35, 0.45 and 0.75; Cluster mutation ratio: 0.0025, 0.005 and 0.0075; Route mutation ratio: 0.025, 0.05 and 0.075; Tournament size: 2.

These give a total of 108 possible execution combinations. For each combination, 3 runs were done, generating a total of 324 executions on the calibration stage. The problem parameters used were: Route duration factor: 8; Slack time factor: 1; Vehicle quantity factor: 0; Excess ride time factor: 2; Wait time factor: 0; Ride time factor: 4.

Finally, the parameters associated to the best results were: Crossover ratio: 0.45, cluster mutation ratio: 0.005 and route mutation ratio: 0.075 for the Small set; Crossover ratio: 0.35, cluster mutation ratio: 0.075 and route mutation ratio: 0.075 for the Medium set; and Crossover ratio: 0.75, cluster mutation ratio: 0.025 and route mutation ratio: 0.025 for the Big set.

With these parameters, the final tests were executed. In this stage, a bigger number of runs per test and a bigger generation number were used. For each instance set, 20 runs and 15.000 generations were considered. A total of 200 tests with the fore-mentioned characteristics were carried out.

5.1 Obtained Results

The results were compared with Jorgensen et al. [2] research (cluster-first route-second GA) and Cordeau & Laporte [4] research on Tabu Search. Table 1 shows the results obtained by our work while Tables 2 and 3 show the results obtained by the previously mentioned researches. Because the compared models do not have the same characteristics, the comparison was done on the basis of time units of two critical factors: On the one hand, the total route duration that is associated with the transport system resources optimization, and on the other hand, the total client travel time that is associated with the offered quality of service.

LLGA Model					
Set	Route duration		Ride time		CPU Time (Min.)
	Best	Avg.	Best	Avg.	
pr01	955,25	1012,32	524,59	546,56	1,36
pr02	1839,06	1836,05	838,41	889,05	4,08
pr03	2787,18	2810,59	1597,95	1634,1	7,96
pr05	4068,05	4118,03	2935,48	3007,97	18,43
pr11	902,18	908,09	449,91	454,4	1,58
pr12	1503,34	1503,71	744,93	766,21	4,49
pr15	4057,08	4118,34	3152,67	3160,79	22,09
pr16	4658,35	4665,77	2348,48	2377,45	17,48
pr17	1223,68	1227,46	612,4	612,4	3,13
pr19	3427,06	3441,71	2515,53	2597,1	25,42

Table 1: Summary of the results obtained by our LLGA model

It is important to mention that our solution considers two restrictions not covered by both, Jorgensen et al. and Cordeau & Laporte researches. These are the time windows as hard constraints and the incapacity of vehicles to transport clients when they are on slack times. Despite this more restrictive scenario our solution performed well compared to them.

Jorgensen et al.					
Set	Route duration		Ride time		CPU Time (Min.)
	Best	Avg.	Best	Avg.	
pr01	1039	1041	310	447	5,57
pr02	1994	1969	1330	1367	11,43
pr03	2781	2779	2894	3081	21,58
pr05	4274	4250	4837	5099	58,23
pr11	928	907	549	630	5,46
pr12	1710	1719	1300	1214	11,72
pr15	4336	4296	4720	4615	58,93
pr16	5227	5309	6397	6134	81,23
pr17	1316	1299	784	990	8,29
pr19	3676	3679	5358	5362	44,66

Table 2: Summary of the results obtained by cluster first- route second GA of Jorgensenet al. [2]

Cordeau & Laporte			
Set	Route duration	Ride time	CPU Time (Min.)
pr01	881	1095	1,9
pr02	1985	1977	8,06
pr03	2579	3587	17,18
pr05	3870	6154	46,24
pr11	965	1042	1,93
pr12	1565	2393	8,29
pr15	3596	6105	54,33
pr16	4072	7347	73,7
pr17	1097	1762	4,23
pr19	3249	5581	51,28

Table 3: Summary of the results obtained by Tabu Search of Cordeau & Laporte [4].

In terms of solution quality, our LLGA model clearly performed better than the Jorgensen et al. solution. A closer look at the tables shows that for all the evaluated datasets our LLGA model presented lower average times for vehicles' route duration, clients' ride times and also CPU time, being this last one specially important.

Regarding Cordeau & Laporte results, based on a Tabu Search solution, on most cases their solution presented best times for the route duration but not for the ride times or the CPU times. In addition, on three cases (pr02, pr11 and pr12) our LLGA solution performed better also for the route duration.

5.2 Results Discussion

As exposed in the results section, our LLGA implementation presents better results for the clients' travel time with respect the other two studies. This is mainly due to the use of time-windows as hard constraints, making it easier to enforce the optimization on this factor. Then, when focusing on the vehicles' route duration our solution showed better times regarding the cluster-first route-second GA of Jorgensen et al. while obtaining worse results when compared to the Tabu-based solution. This can be understood as the cost of having good average ride times for clients, as there is a trade-off relation among both variables.

The improvement obtained in our solution is mainly explained because of the use of the so-called precedence table of events and the incompatible clients list. Both elements provide a way to reduce the search-space when considering the time-windows restriction.

In this sense, it is important to remember that the DARPTW problem behaves as a deceptive problem when solved through GAs, mainly because of the difficulty for the GA operators to find the appropriate building blocks for constructing feasible solutions. This fact is especially true when facing the DARP with TWs, as the time-windows do impose additional restrictions on the clients' requests that make the region of feasible solutions highly non-convex.

This means that it is very easy to move within this region from a feasible to an infeasible solution. Furthermore, depending on the tightness or looseness of the time-windows such region may look like a small group of feasible points spread over a big region of infeasible solutions.

Under such an scenario, what the precedence table does is to make such region more convex by pre-processing which sequences of events (pickup & delivery) are feasible within the "sea" of infeasible sequences due to incompatibilities in their respective time windows, making it impossible for a vehicle to serve both events in that sequence and satisfy their time intervals.

A similar situation happens with the incompatible clients' list. In this case, both events of the client are considered, his pickup and delivery, detecting the cases in which it is not possible for a vehicle to serve both clients and their time-windows constraints no matter the sequence used and no matter which other clients are assigned. This is important for identifying requests that are too close in time (either at pickup or delivery) while being too far geographically, avoiding putting them together.

In this way, the initial population and GA operators move more around feasible solutions. This reduction follows a similar principle as in Constraint Satisfaction Problems and their techniques to reduce the domain of variables.

From other point of view, it can be seen as a search with memory, as in Tabu search. However, in this case the memory is used for making the algorithm to "remember" which portions of sequences are feasible in order to reduce effort instead of remembering the solutions found so far to avoid local optima. Another important issue is that the CPU time in all cases is lower, being sometimes even the half of it or even less (e.g. pr5, pr15, pr16 and pr19). It is worth highlighting that the LLGA tests were done with a 2.66 GHz Intel Pentium 4 CPU, while the Cordeau & Laporte tests were done with a 2.0 GHz Intel Pentium 4 CPU and the Jorgensen et al. tests were done with a 2.0 GHz Intel Celeron CPU. Although the hardware configurations are dissimilar, they do not completely justify the time improvement. Undoubtedly, the precedence table of events and the incompatible clients' list have caused this

diminishing.

6 Conclusions

The proposed model has shown interesting results according to the comparisons made, highlighting the times lowering, undoubtedly the weaker GA factor. This work will allow researchers to develop comparisons of highly restrictive models, unlike most works in literature where a bigger number of constraints are relaxed.

There are two important elements of the developed model on this work that must be remarked. On the one hand, the use of pre-feasibility schemas has represented an interesting support tool for the GA behavior. Although it depends strictly on the problem, the GA behaves better when the search space is bigger and complex, as in many NP-hard problems while having a convex feasible region of solutions.

Bibliography

[1] K. Bergvinsdottir, *The Genetic Algorithm for solving the Dial-a-Ride Problem,* Master's thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, 2004.

[2] R. M. Jorgensen, J. Larsen, K. B. Bergvinsdottir, Solving the Dial-a-Ride problem using genetic algorithms, *J. Oper. Res. Soc.*, Vol.58, pp. 1321-1331, 2006.

[3] J. Cordeau, A Branch-and-Cut Algorithm for the Dial-a-Ride Problem, *Operations Research*, Vol. 54, pp. 573-586, 2006.

[4] J. Cordeau, G. Laporte, A tabu search heuristic for the static multi-vehicle dial-a-ride problem, *Transportation Research Part B*, Vol. 37, Issue 6, pp. 579-594, 2003.

[5] J. Cordeau, G. Laporte, *Benchmark datasets* accessed on march-2007, available online at: http://neumann.hec.ca/chairedistributique/data/darp/. 2007.

[6] C. Cubillos, *MADARP: Multi-Agent Framework for Transportation Systems,* Thesis for the academic degree of Dottore di Ricerca in Ingegneria Informatica e dei Sistemi (ciclo XVII), Politecnico di Torino, 2005.

[7] C. Cubillos, N. Rodriguez, B. Crawford, A Study on Genetic Algorithms for the DARP Problem, *Springer Berlin-Heidelberg LNCS*, vol. 4527, pp. 498-507, 2007.

[8] G. Harik, D.E. Goldberg, Learning linkage, *Foundations of Genetic Algorithms*, vol. 4, pp. 247-262, 1996.

[9] J.H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to biology, control and artificial intelligence,* MIT Press, ISBN 0-262-58111-6, 1998.

[10] M.W.P. Savelsbergh, The General Pickup and Delivery Problem, *Transportation Science*, Vol. 29, pp. 17-29, 1995.

[11] S. Thangiah, Vehicle Routing with Time Windows using Genetic Algorithms, *Application Handbook of Genetic Algorithms: New Frontiers*, Vol. II. Lance Chambers (Ed.). CRC Press, 1995.