Applying Genetic Algorithms To The Dial-A-Ride Problem With Time Windows

CLAUDIO CUBILLOS F., FRANCO GUIDI-POLANCO AND CLAUDIO DEMARTINI
Dipartimento di Automatica e Informatica
Politecnico di Torino
C.so Duca degli Abruzzi 24, Turin, 10129
ITALY

Abstract: - The Dial-a-ride problem with time windows (DARPTW) assigns and schedules users transport requests to a fleet of vehicles enabled to fulfill the required service. The literature offers different heuristics for solving DARPTW problems, which are based on the extension of traditional graph search algorithms. On the other hand, the approach through Genetic Algorithms (GA) has been experienced in problems of combinatorial optimization. In this article we present our work and initial results with a framework to develop and test different GAs in the aim of finding an appropriate encoding and configuration, specifically for the DARPTW problem.

Key-Words: - Genetic algorithms, DARPTW, Transport systems, Optimization.

1 Introduction

Research in the field of passenger transport planning systems has received an increasing attention during last years, due to the congestion and contamination problems, and the number of accidents generated by an always increasing number of vehicles in our cities. As response, new alternatives to satisfy de transport demands of the citizens are being conceived [1].

This lead us to the problem of efficiently managing these transport requests. In literature this problem is often stated as Dial-a-Ride Problem (DARP) or Pickup and Delivery Problem (PDP). It consists in a minimal fleet of vehicles with limited capacity that must transport a set of clients from an initial pickup point to a final delivery location. Normally "Time windows" constraints are added, which specify the time intervals within which each client must be picked-up and delivered, generating the DARPTW (or PDPTW) problem.

The Dial-a-ride problem with time windows can be seen as a derivation of the Vehicle Routing Problem with time windows (VRPTW). The VRPTW has been studied more deeply as it treats a simpler planning problem; assumes a central depot for all the vehicles, considers only the picking-up of goods (not the pickup & delivery), and therefore no disutility function exists for the transported entities.

Traditional approaches in Dial-a-Ride service planning are usually implemented as heuristic procedures that extend basic graph search algorithms, acting over large collections of data that describe the entities of the domain problem (vehicles, service requests, schedules). The most commercially used transport planning algorithms correspond to extensions of the solomon's heuristic for the VRPTW [15]. An example can be found in our past research [4], where we implemented a version of Jaw's Advanced Dial-A-Ride with Time Windows (ADARTW) algorithm [8].

2 Related Work

The Vehicle Routing Problem has been investigated for the past 20. More recent research in the field tries to include newer techniques to improve the quality of the obtained solutions. In [11], Li presented a metaheuristic for the PDPTW. A tabu-search can be found in [14], implemented for real-life problems including time-window constraints. Kohout and Erol [10] showed an agent-based implementation that uses an stochastic improvement in the final solutions.

On the other hand, Genetic Algorithms (GA) have been successfully used as optimizators under different research domains, including simple route optimization as in [3]. In the Vehicle Routing Problem several works can be found. In [2] an hybrid GA is reported, that combines the genetic algorithm together with a greedy constructive heuristic. Thangiah [16] uses a genetic approach for clustering initial routes. In [9] an hybrid approach uses a GA together with dynamic programming and [13] combines GA with tabu-search. In [12], Maeda uses only GA for the VRP problem but with time

deadlines.

Under some domains, traditional GAs do not perform well [6]. These are the so called deceptive problems in which the GA is not able to converge to a near optimal solutions mainly due to a bad linking of the building-blocks. Much research can be found about this linking problem ([5], [6], [7]) and its possible solutions.

3 Objectives

No relevant work could be found on the dial-a-ride sub-problem making use of the genetic paradigm. This can be explained because of the additional requirements of the dial-a-ride (pickup & delivery constraints + users' level of service) and because the scheduling of clients has taken more followers in the recent years with the need of Demand Responsive Transport Systems (DRTS). Another reason can be found, as it is shown by our results, in the fact that DARP corresponds to a deceptive problem, so traditional GAs do not perform well.

For this reason, the aim of this work is not to develop a genetic algorithm to obtain a universal optimizer (blackbox optimization). On the contrary, we pursue the objective of developing a specific GA encoding for the DARPTW problem and obtain non deceptive results.

As we have shown above, many works make use of the GA together with other techniques for improving the final solution. Others use the GA as a clusterization technique for the assignment of the clients to the vehicles but not for the scheduling of the trips itself.

Our intention is to use the GA for the whole DARPTW problem, that is, the assignment and scheduling of the clients, without post-optimization procedures and to find the best combination of chromosome's encoding and evolutionary operators (selection, crossover and mutation) for the dial-a-ride problem. For that reason, we have developed a GA framework that allows us to test several GAs with different configurations of genetic operators, chromosome's encoding and parameters.

4 The Dial-A-Ride Problem

The problem we are treating consists of a set C of geographically distributed transportation requests, coming from customers that should be served by a set of identical vehicles V.

The service can be defined as picking-up the client from the origin node ns_i, and conducting it to a destination node nd_i, where $\{ns_i, nd_i\} \subset N$, with N the entire set of nodes that represent the network.

The service must be executed considering a *time window for delivery* constraint defined for each customer, expressed in terms of an earliest delivery time and a latest delivery time, the pair (edt_i, ldt_i) with $edt_i < ldt_i \ \forall i \in C$. In this a way, a vehicle serving the customer i must reach nd_i, neither not before the edt_i time, nor after the ldt_i time.

Two functions, $DRT(N \times N) \rightarrow \Re$ and $MRT(N \times N) \rightarrow \Re$, define the *direct ride time* (optimistic time) and the *maximum ride time* (pessimistic time) required to reach the node nd_j from $ns_i \ \forall i \neq j$. Delivery times define a *time window for pick-up*, the pair (ept_i, lpt_i), where $ept_i = edt_i - MRT(Ns_i, Nd_i) \wedge lpt_i = edt_i - DRT(Ns_i, Nd_i)$ (see Figure 1).

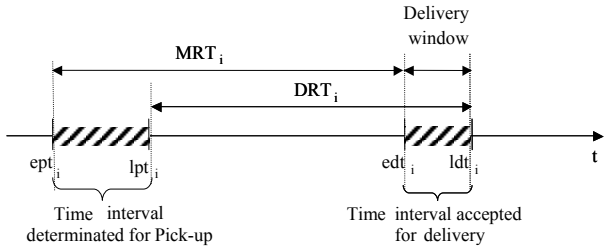

Fig.1 – Users Time Windows.

In practical terms the customer is supposed to attend the vehicle at the pick-up point not after the time ept_i. Vehicles are not allowed to wait for a client, so they have to be scheduled to reach the point ns_i for serving the request i not before the time ept_i. On the other hand, the passenger has to be picked-up not after the time lpt_i otherwise his request (theoretically) will not be satisfied.

Service requests have to be assigned to Vehicles and scheduled according to the time restrictions. There exists no restriction about the minimum number of passengers to serve, but the maximum capacity of the vehicles must never be exceeded.

In our model we consider the possibility of a multi-depot scenario, that is, the vehicle i starts from the depot D_{Si} and after serving their last clients they turn back to the depot D_{Fi}, where $\{D_{Si}\ D_{Fi}\} \in N$. The objective function pursues the minimization of a disutility function that considers the fleet operator (number of vehicles required, fixed and variable travelling costs) and the served users (effective waiting time, effective ride time).

5 The GA Framework for DARPTW

We have developed a GA framework for the assignment and scheduling of customers in the Dial-

a-Ride problem. The GA should tackle the whole DARPTW problem, so each individual of the population corresponds to an entire solution of the problem, that is, a set of vehicles and their routes that fulfil all customer requests. In our model the chromosomes evolve tempting to minimize the fitness function, through a process that finishes when the evolution reaches a pre-specified number of generations. The evolution of the chromosomes is the result of the continuous application of the genetic operators (selection, reproduction, and mutation) over the population.

As mentioned earlier, we want to consider in the GA framework different kinds of genetic operators and chromosome's encoding. In particular we are taking into consideration the following:

a) *Genotype*: The models for encoding the problem into a chromosome representation. The idea is to find the most suitable for the DARPTW problem.
b) *Phenotype*: The definition of a fitness function according to the given genotype.
c) *Initial population*: The procedures for the construction of the initial population (building blocks).
d) *Selection*: Models for the parents selection for reproduction: tournament, roulette-wheel, etc.
e) *Crossover*: The different types of crossover operators: PMX, Edge recombination, cycle crossover, position crossover, and Merge crossover.
f) *Mutation*: Different mutation operators: bit-level mutation and 2-opt, among others.

6 Implementation

In order to perform the tests with the different configurations, several procedures have to be implemented. Up to now we have programmed the following encodings and genetic operators:

Genotype: For the application of GA to a problem, is required to model a representation of the solution in a chromosome. As first encoding we choose a representation in which the chromosome is made up of a *bus-passenger* list, where each bus-passenger pair corresponds to a gene as shows the Figure 2. A simple reading template was provided to decode the chromosome, which consists in interpreting the first passenger's occurrence always as a pick-up, and the second one, always as a delivery. In this way the chromosome must have exactly two genes for each passenger, which determines the length of the chromosome. Note that these 2 genes associated to a customer (pickup/delivery) can specify different buses. Therefore, the bus finally assigned to the customer is the one specified by the pick-up. The bus information in the delivery gene is used as a sort of recessive genetic material useful upon recombination.

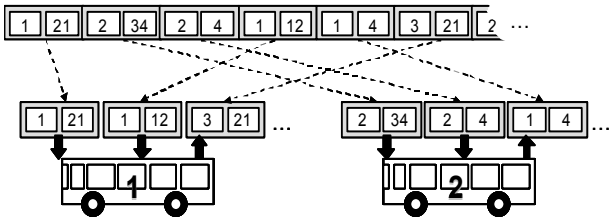

Fig.2 – Decoding the chromosome.

Other encodings are considered to be implemented, like the *locus-bus-passenger* gene, that will allow putting the information about the relative position of the gene in the chromosome inside the gene itself, enabling the adoption of a linkage learning model.

Phenotype: In order to evaluate the "quality" of a chromosome and to decide its reproduction probability, a fitness function was defined. In the case of the DARP problem we adopted as fitness function the disutility function already defined in Section 2.

Initial Population: The generation of an initial population that covers the hole problem's state-space is crucial for the final performance and convergence of a given Genetic algorithm. In this sense, we have implemented two alternative procedures for the population initialization: the first is the common random generation of the chromosome's genes at bit-level.

The second initialization procedure is based on an basic insertion heuristic for scheduling the passengers into the available vehicles. The heuristic picks one by one the clients from a list and tries to schedule them in the first possible vehicle. Different solutions (population's individuals) are generated by applying a random ordering to the list of clients and vehicles each time the insertion heuristic is to be used.

Selection: Firstly, a *tournament* selection has been used to choose the individuals(mates) for reproduction. It works by choosing a number of t_s (tournament size) individuals from the population in a random way. Then the individual with the lowest fitness value is selected (the fitness function in our case is a cost function, so lower values are better than higher values). The number of tournaments applied in a generation depends on the number of individuals required to reproduce.

Crossover: The reproduction is the operation that produces a new individual pertaining to a new generation, due to the combination of the genetic material of its parents. For the framework we have already implemented the basic bit-level one-point crossover and a modified version of the Partial Match Crossover(PMX).

Mutation: The mutation allows an individual to slightly change the inherited genetic material. We have already implemented a bit-level mutation and the 2-opt operator.

On the other hand, the tests consider the evaluation of different values for several parameters. The considered parameters are: Problem size (number of transport requests), temporal and spatial distribution of the demands (i.e. uniform, clustered, mixed), population size, crossover and mutation probability, together with selection parameters. As reference we will compare the tests outputs of the GA model with the results obtained with the application of ADARTW algorithm [4] for different problem conditions.

7 Results and Analysis

The GA model was implemented as a parametrizable C++ program that can be configured to execute different runs with different treatment for problem parameters. Firstly, we have implemented a traditional bit-level GA model with the following characteristics: bit-string chromosome using the *bus-passenger* gene, tournament selection, and bit-level population initialization, crossover and mutation. This GA model was unable to solve small problems (no feasible solution found) of 10 requests.

Then, a second implementation changed the bit-string by an integer chromosome representation. The crossover and mutation operators together with the population initialization procedure were modified to operate over the integer representation. This GA behaved well for small request sizes, up to 25 requests. From 30 requests on, the GA was not able to arrive to any feasible solution. This happened because the used population initialization step (random integer generation) produced only chromosomes with unfeasible solutions. This happened with population sizes of 50, 100 and 200 chromosomes.

Another GA implementation was done (the 3rd one), which included an insertion heuristic for the population initialization procedure together with the PMX crossover and 2-opt operator. The introduction of an insertion heuristic for the population initialization procedure, allowed to start the GA with a population with some individuals having feasible solutions. This allowed us to use the GA with request sizes bigger than 50 but lower than 100 and to obtain better results.

A possible explanation of what happened above can be the fact that in traditional Vehicle Routing problems is assumed only one depot. Therefore all the routes start and end on the same depot, allowing the adoption of certain simplifications when encoding the routes into the chromosome. On the contrary, the dial-a-ride problem we are considering assumes a multi-depot scenario. This, together with the additional requirements (explained in Section 3), makes this problem behave like a deceptive one when using a traditional GA.

Additionally, with this last GA implementation (the 3rd one), we decided to perform a test varying some of the parameters. At first view, the GA seems to work more efficiently and quickly with small sizes of requests (Under 100). For that reason we decided to use a small number of requests (5 scenarios of 25 trip requests). In particular we have performed an initial test of 640 runs, considering the combination of the following parameters:

- *Scenarios*: 5 demand scenarios, each with 25 trip requests distributed uniformly in a two hours horizon.
- *Maximum number of available buses*: 25
- *Population size*: 50 - 100 - 150 - 200 individuals.
- *Number of generations*: 1.000 and 11.000 generations
- *Crossover probability*: 0,7 – 0,8 – 0,9 – 1.0
- *Mutation probability*: 0,0015 and 0,0095
- *Tournament size*: 50% and 60% of the entire population.

Table 1 – Comparison of best solutions obtained.

Scena-rio	Genetic Algorithm						ADARTW			Notes
	Best solution in terms of minimum fitness value			Best solution in terms of minimum number of vehicles			Fit-ness	Vehi-cles	Time [sec]	
	Fit-ness	Vehi-cles	Time [sec]	Fit-ness	Vehi-cles	Time [sec]				
1	74.7	11	876	103.7	8	393	56.2	9	1	(*)
2	92.6	11	234	125.2	9	1220	9.0	11	1	(*)
3	75.7	9	1013	77.0	7	717	39.8	8	1	(*)
4	85.1	9	930	91.7	8	945	27.8	8	1	
5	88.0	9	461	92.5	8	681	42.1	9	1	(*)

(*) Better results in (minimun) number of vehicles given by the GA program.

A general conclusion from the analysis of this first study is that the ADARTW heuristic provides much better results than the implemented GA model, in terms of the fitness function value. However, in terms of the number of required vehicles the GA model has produced better results than the heuristic in 4 of the five scenarios with 25 trip requests. Table 1 summarizes the bests solutions obtained by the GA model in terms of the fitness value, and in terms of number of vehicles.

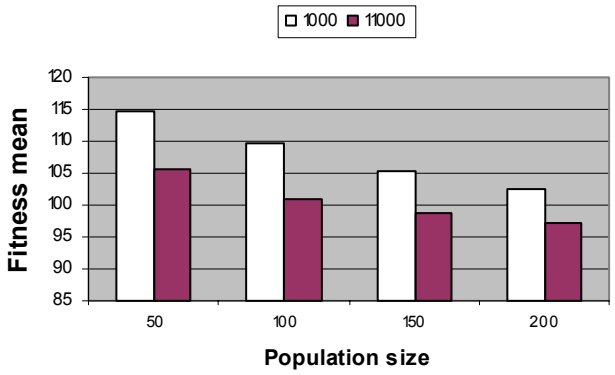

Fig.3 – Fitness value versus population size for two generation sizes (1.000 and 11.000 generations).

Figure 3 shows the tendency observed on the fitness function when the population size increases. Better results are given for bigger population sizes, and this can be explained because a bigger population size allows to examine a larger portion of the solutions space on each generation. On the same graphic can be observed that a bigger number of generations allows to obtain better results for the fitness function, independently from the considered population sizes.

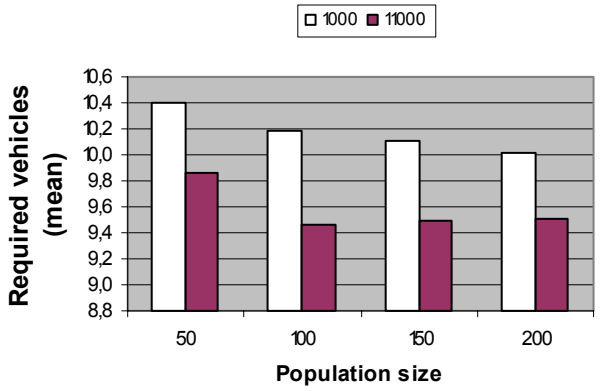

Fig.4 – Number of vehicles versus population for two generation sizes (1.000 and 11.000 generations)

On the other hand, Figure 4 shows that the search for the optimum number of vehicles produce better results with large number of generations, and good results can be obtained even in the case of relatively small population sizes.

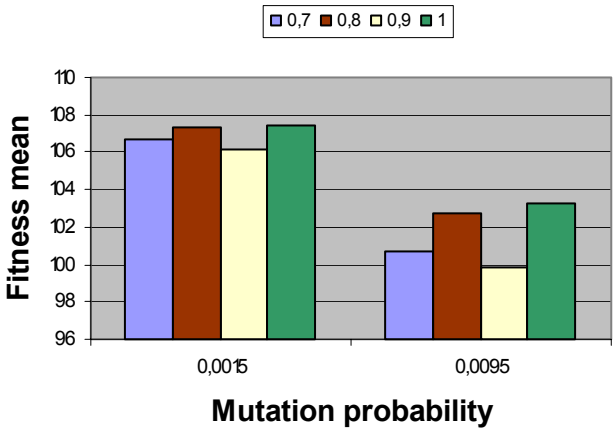

Fig. 5 – Fitness function versus mutation probability for different crossover probabilities (0,7 – 0,8 – 0,9 – 1,0).

Figure 5 shows the influence of the mutation probability in the value obtained for the fitness function. The graphic shows that independently from the evaluated crossover probability values, the higher mutation probability produces the lower fitness value (cost). In the same graphic can be appreciated that the implemented model doesn't produce significant differences of fitness function values when varying the crossover probability and the mutation probability is low, but this seems to multiply the effect in a non predictable way when the mutation probability is higher.

8 Conclusions

Our research aimed to determine the conditions in which GA can be an useful approach to deal with Dial-a-Ride problems. We have tested an initial configuration with a bit-string chromosome, tournament selection, and bit-level population initialization, crossover and mutation. It clearly performed poorly, being almost always unable to arrive to a feasible solution. Then, an integer representation was adopted for the bus-passenger gene, noticing an immediate improvement.

Afterwards, the PMX crossover and 2-opt were included together with an insertion heuristic for the population initialization, resulting in even better behaviour and results. These solutions were comparable to the ones obtained using the

ADARTW algorithm implementation[4] in relation with the number of used vehicles but not with respect to the value of the fitness (disutility) function.

9 Future Work

The idea is to continue the implementation of the other genetic operators and specially the *locus-bus-passenger* gene encoding to then adapt and implement a linkage learning module as the one presented by Harik [7].

With this modifications we will be able to obtain better results in terms of the solutions' disutility function and to use the genetic algorithm with bigger problem sizes over 100 clients.

References:

[1] Ambrosino, G. et al, EBusiness Applications to Flexible Transport and Mobility Services. 2001. Available online at: http://citeseer.nj.nec.com/ambrosino01ebusiness.html.

[2] Blanton, J; Wainwright, R. Multiple Vehicle Routing with Time and Capacity constraints Using Genetic Algorithms. *In Proc. of the 5th International Conference on Genetic Algorithms*. 1:452-459, 1993.

[3] Chang Wook, A.; Ramakrishna, R.S. A Genetic Algorithm For Shortest Path Routing Problem And The Sizing Of Populations. *In IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 6, 2002, pp. 566 -579.

[4] Cubillos, C. et al. On user requirements and operator purposes in Dial-a-Ride services. *Proceedings of the 9th Meeting of the EURO Working Group on Transportation.* 2002, pp. 677-684.

[5] Goldberg, D. et al. Toward a Better Understanding Of Mixing in Genetic Algorithms. *J. Soc. Instrument and Control Engineers*. Vol. 32, No 1, pp. 10-16, 1993.

[6] Harik, G. Learning Gene Linkage to Efficiently Solve Problems of Bounded Difficulty Using Genetic Algorithms. *PhD Thesis*. University of Michigan. 1997.

[7] Harik, G; Goldberg, D.E. Learning Linkage. *In foundations of Genetic Algorithms*. Vol. 4, 1997, pp. 247-262.

[8] Jaw, J. et al. A heuristic algorithm for the multiple-vehicle advance request dial-a-ride problem with time windows. *Transportation Research*. Vol. 20B, No 3, 1986, pp. 243-257.

[9] Jih, W; Hsu, J. Dynamic Vehicle Routing using Hybrid Genetic Algorithms. In Proc. Of the IEEE Int. Conf. on robotics & Automation. Detroit, May, 1999.

[10] Kohout, R; Erol, K. Robert C. In-Time Agent-Based Vehicle Routing with a Stochastic Improvement Heuristic. In Proc. Of the AAAI/IAAI Int. Conf. Orlando, Florida, 1999, pp. 864-869.

[11] Li, H; Lim, A. A Metaheuristic for the Pickup and Delivery problem with Time Windows. *In 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'01)*. Texas, November, 2001.

[12] Maeda, O. et al. A Genetic Algorithm Approach to Vehicle Routing Problem with Time Deadlines in Geographical Information Systems. *Presented at the IEEE International Conference on Systems, Man, and Cybernetics*. Vol.II, Tokyo, Oct. 1999, pp.595-600.

[13] Ombuki, O. et al. A Hybrid Search Based on Genetic Algorithms and Tabu Search for Vehicle Routing. *Presented at the 6th IASTED International Conference on Artificial Intelligence and Soft Computing*. Banff, Canada, July 2002, pp. 176-181.

[14] Rochat, Y; Semet, F. A Tabu Search Approach for delivering Pet food and flour in Switzerland. *Journal of Operation Research Society*. 1:1233-1246, 1994.

[15] Solomon, M. Algorithms For The Vehicle Routing And Scheduling Problems With Time Window Constraints. *Operations Research*, No. 35. 1987, pp. 254-265.

[16] Thangiah, S. Vehicle Routing with Time Windows using Genetic Algorithms. *Application Handbook of Genetic Algorithm: New Frontier*, 2:253-277, 1995.