A new evolutionary hybrid algorithm to solve demand responsive transportation problems

Roberto Carballedo, Eneko Osaba, Pablo Fernández and Asier Perallos

Deusto Institute of Technology - DeustoTech, University of Deusto, Avda. Universidades 24, 48007 - Bilbao, Spain
{e.osaba, roberto.carballedo, perallos, pablo.fernandez}@deusto.es

Abstract. This paper shows the work done in the definition of a new hybrid algorithm that is based on two evolutionary techniques: simulated annealing and genetic algorithms. The new algorithm has been used to solve the problem of finding the optimal route for a bus in a rural area where people are geographically dispersed. The result of the work done is an algorithm that (in a reasonable time) is able to obtain good solutions regardless of the number of stops along a route.

Keywords: meta-heuristics, simulated annealing, genetic algorithm, demand responsive transport.

1 Introduction

Nowadays public transport systems have some drawbacks to meet the demand for all passengers. The most obvious one is the limitation of the resources. Although each transport system has its own characteristics, there are some limitations shared by all of them: the capacity of the vehicles, the frequency and schedules of the services, and the geographical area of coverage. As a result of this arises the concept of transportation on demand. This concept aims to adapt transportation systems to passenger demand in an efficient manner. Many of the techniques used to solve these problems do not yield an exact solution. This is because the types of problems to be solved are classified as NP-hard [1]. For this reason, heuristics techniques are used for obtaining good approximations.

This paper is divided in 6 sections. Section 2 presents the main types of well known problems related to the work done. Section 3 illustrates the most commonly used strategies in the field of route optimization. Section 4 presents the approach followed to define our hybrid algorithm. Section 5 presents the results of the tests done to validate our algorithm and finally conclusions and future work is detailed.

2 Transportation on demand

Transportation-On-Demand (TOD) [2] is concerned with the transportation of passengers or goods between specific origins and destinations at the request of users. Most TOD problems are characterized by the presence of three often conflicting objectives: maximizing the number of requests served, minimizing operating costs and minimizing user inconvenience. As is common in many combinatorial optimization problems, these objectives are conflicting and it is needed to sort them by importance.

2.1 Well know transportation problems

Most of the problems arisen in transportation on demand topic have similar characteristics, which means that they can be framed as instances of other generic and well know problems. In this section, we present the most common traditional problems in the field of transportation on demand.

Traveling Salesman Problem (TSP) [4]: The Travelling Salesman Problem (TSP) is an NP-hard problem in combinatorial optimization studied in operations research and theoretical computer science. Given a list of cities and their pair-wise distances, the task is to find a shortest possible tour that visits each city exactly once. This type of problem is used as a benchmark for many optimization algorithms.

Vehicle Routing Problem (VRP) [5]: The vehicle routing problem (VRP) is a generalization of the TSP. The aim of the problem is to service a number of customers with a fleet of vehicles. Often the context of this type of problem is related to deliver goods located at a central depot to customers which have placed orders for such goods. Implicit is the goal of minimizing the cost of distributing the goods. Many variants of the VRP are described in the literature [6]. These problems include the addition of variables and constraints. One of the most popular variants includes time windows for deliveries. These time windows represent the time within which the deliveries (or visits) must be made. [7]

Demand Responsive Transport (DRT): Demand Responsive Transport or Demand-Responsive Transit (DRT) or Demand Responsive Service is an advanced, user-oriented form of public transport. It is characterized by flexible routing and scheduling of small/medium vehicles operating in shared-ride mode between pick-up and drop-off locations according to passengers needs. DRT systems provide a public transport service in rural areas or areas of low passenger demand, where a regular bus service may not be economically viable. DRT systems are characterized by the flexibility of the planning of vehicle routes. These routes may vary according to the passenger' needs in real time. This is the type of problem that we used to benchmark the algorithm proposed in this paper.

3 Artificial intelligence techniques and algorithms

In the literature we can find many attempts to find an exact solution to the problems explained in the previous section. For most routing problems is not possible to find the optimal solution, for that reason, there have been a number of strategies to find an acceptable solution, taking care of the basic criteria of the computational complexity: time needed to obtain the solution and consumption of computational resources. This section details the most commonly used techniques for solving the problems explained in the previous section.

3.1 Local search algorithms

Most of solution methods begin the resolution process by generating an initial solution that does not have to be correct. From this, and iteratively, these algorithms "search" for a better/good solution. These techniques use an objective function that measures the quality of the solutions obtained during the search process. In this scope we can find the local search techniques:

Simulated Annealing [8]: This is one of the most popular local search techniques. It is based on the physical principle of cooling metal. Using that analogy, it generates an initial solution and the process proceeds by selecting new solutions randomly. The new solutions are not always better than the initial solution, but as time passes and the temperature decreases (the metal becomes stronger), each new solution must be better than previous solutions.

Tabu search [9]: This technique is similar to Simulated Annealing, but with a different approach when selecting the successive solutions. In this case, several memory spaces are used, in which solutions found and discarded during the search process are stored.

Ant Colony [10]: This algorithm simulates a colony of artificial ants working in groups and communicating through artificial pheromones trails. Each artificial ant builds a solution to the problem and the path to reach that solution. When all ants have completed a trip to a solution and all trips are reviewed, the traces are stored. The process of paths constructing is repeated until almost all ants follow the same trip in each cycle.

3.2 Evolutionary algorithms [11]

These methods include algorithms inspired by the laws of natural selection and the evolution of the animal species. In most cases, an initial population of solutions is defined. This initial population consists of a number of individuals (solutions of the problem.) Then, with the combination and evolution of these individuals, the algorithm tries to get a better solution. The most popular technique in this field is genetic algorithms, which are inspired by the biological evolution of species.

3.3 Hybrid local search [12]

This is one of the most used strategies. This approach attempts to solve the problems faced by traditional strategies. To this end, several strategies are combined (usually 2) in a single process. This allows grouping the advantages of each strategy and solving its individual problems. As explained below, this is the approach we used for the design of our algorithm.

4 Proposed algorithms

Having defined the main types of problems related to route optimization, and techniques used for resolution, we will specify the algorithm that we designed, and the problem we used to validate it. The algorithm we have designed allows us to model and solve any combinatorial optimization problem. Nevertheless, we have defined an instance of a DRT problem, to illustrate the operation and performance of the algorithm.

4.1 Description of our DRT problem

To verify our new algorithm, we have defined an instance of a DRT problem. Our problem refers to a bus on demand system. The passengers make requests for travel from one stop to another. There are 15 stops. Five of the stops are mandatory and the rest are optional. The position of all stops is fixed and known, but the passage of buses by an optional stop depends on the passenger demand. The bus will pass the optional stops, if passenger demand exceeds a certain threshold. If the bus does not go through any optional stop, the route between the mandatory stops is always the same. If the bus has to pass more than an optional stop, the route between two mandatory stops should be calculated dynamically to minimize the distance traveled by the bus. The optimization problem we have to solve is based on the calculation of the optimal path between two mandatory stops, through a series of optional stops.

To solve the problem, we designed a hybrid algorithm that combines simulated annealing methods and genetic algorithms. Then we explain the details of each technique separately.

4.2 Simulated annealing

As explained above, this is a meta-heuristic algorithm based on the physical principle of metals cooling. The most important characteristic of this algorithm are:

Concept of state: A state of a problem, define a specific situation of the problem. This situation is defined by the fundamental elements that make up the problem. In our problem, a state is defined by the order in which the bus travels through the optional stops between two mandatory stops. Then the state of our problem is a path between several stations.

Evaluation function: This function measures the quality of a state. This quality is usually associated with a numerical value that allows us to compare states and determine which is better. In our problem, the evaluation function is the sum of the distances between the stations that make up a state. The evaluation function is the criterion for determining that a solution is better than another.

Successor function: The objective of this function is to obtain a new state based on the current state and the temperature. For this, it takes a random exchange in the order of the stations of the current state, changing the path also. The successor function is designed to create a new state from another. In our problem, the successor function performs a random change of the position of two stops. With this change, a new state is created. This new state represents a new route and it has a new value of evaluation function, usually different from the previous state's value.

As explained previously, the process of simulated annealing algorithm is based on the generation of successor states iteratively. In each iteration, if the value of the evaluation function of the new state is better than the current state value, the successor state becomes the new current state. Otherwise, the successor state will be the new current state with a certain probability that decreases as temperature decreases. Therefore, the temperature is used to select the successor states that do not have a better evaluation function as the new current state.

The temperature function is a mathematical function, which is updated each iteration, and allows controlling the selection of "bad" successor states. In the first iteration, the value of the temperature function will be high and the probability of choosing "bad" successor states will be great, but as the temperature value decreases, the probability of choosing "bad" states, will also decrease.

4.3 Genetic algorithm

Genetic algorithms are based on the principles of natural selection of species. For this reason, these algorithms work with concepts of chromosomes, genes, genetic combination and mutation.

One of the most important tasks when working with genetic algorithms is the definition of the concept of state. The states of a genetic algorithm (also known as chromosome) are composed by genes. Each gene is a property or a characteristic of the problem. In our problem, a gene represents a stop, and a chromosome is defined by a sequence of stops.

The operation of a genetic algorithm is based on the evolution of an initial population of chromosomes through a series of iterations. The chromosomes evolve through the crossover and the mutation of genes. The basic operation of a genetic algorithm can be defined as follows:

1. Creation of the initial population. In our problem, we create a series of random routes, which represent the initial population.
2. Evaluation of each of the individuals (chromosomes) using an evaluation function. In our problem, this evaluation function is based on the sum of the distance between the stops that define a chromosome.
3. Start an iterative process until it reaches the threshold of generations

4. Selection of the best chromosomes to be parents. The selection process was carried out based on a fitness function.
    4.1. Generation of new chromosomes from the cross between parental chromosomes. The creation of new chromosomes is done using a crossover function.
    4.2. Once new chromosomes are generated, a process of mutation of some genes of the new chromosomes is performed.
    4.3. Selection of chromosomes that form part of the new population. After performing the process of crossover and mutation, the resulting chromosomes are evaluated by fitness function, and the best ones are selected to be part of the new population.
5. Once the process of generating new populations, the solution is the best chromosome of the current population.

Fitness function: This is the function used to measure the quality of the chromosomes. The quality depends on the order of genes, since the value is the sum of the distances between the genes (stops) of a chromosome.

Crossover function: This is the function used to perform the reproduction process. Usually each crossover generates two children. Each child is formed from fragments of each of their parents. In our problem this process is complex, since stations cannot be repeated. This is the simplest reproduction process but there are other ways to make the crossover process. [13].

Selection criteria: The selection criterion is used twice in the process of the algorithm: the selection of the parents of the new population and the selection of the best individuals after a full iteration. There are multiple criteria, from which selected all individuals, even those who selected only the best individuals (according to fitness function). In our algorithm Stochastic Remainder Criteria was used. This selection criterion selects all individuals whose probability of selection is above the average probability of selection of the entire population (according to the value of the fitness function). If this criterion is not reached the target number of individuals to choose, other individuals were selected randomly.

5 Test and solution proposed

As indicated above, for the design of our hybrid algorithm, separate versions of simulated annealing and a genetic algorithm have been implemented. In addition, we have implemented a "brute force" algorithm, to find out the optimal solution for small instances of the problem (with few intermediate stops).

With these 3 algorithms, there have been a series of tests to measure the performance of each algorithm and the ability of each one to solve the problem. As a result of these tests, we have obtained several conclusions:

1. The “brute force” algorithm is optimal because it always finds the best solution. Even so, it has the disadvantage that the execution time is unacceptable when the number of stations increases to more than 9 (for a large number of stations cannot even get a solution). This algorithm cannot be used in a real scenario.

2. The simulated annealing algorithm only finds the optimal solution when the first and last station does not vary during the resolution process. Running time is always the same regardless of the number of stops.
3. In the case of genetic algorithm, the execution time is constant if the number of generations is also constant. An advantage of this algorithm is that the probability of finding a good solution is independent of the number of stops.

After preliminary analysis of algorithms separately, we came to the conclusion that the results of runtime and solution quality were not good. For this reason we decided to combine the two heuristics.

5.1 Our hybrid algorithm

Our hybrid algorithm came up with the aim to combine the advantages of genetic algorithms and simulated annealing:

- Rapid and constant execution time (simulated annealing).
- Probability of finding a good solution for the problem instances with many stops (genetic algorithm).

The solution would avoid the main drawback of the two algorithms:

- The solution should be optimal or very close to it.

With all these goals, it thought about making the hybrid. By nature of the two algorithms, it is appropriate to insert the execution of simulated annealing algorithm in the execution of genetic algorithm. That is because the first algorithm is focused on only one solution and the second works simultaneously with different solutions.

Having decided the model of integration, there were two options to do the integration:

- Integrate the simulated annealing in the process of creating the initial population.
- Integrate the simulated annealing in the process of reproduction, right after generating the new population.

After several tests, we concluded that the most effective solution was to apply the simulated annealing algorithm just after the reproduction process. Below is a table showing the results of the tests. The table shows the number of stops, the number of generations used in the genetic algorithm, runtime, and the percentage of times the algorithm finds the optimal solution.

Table 1 Results of the tests.

N. of stations	N. of generations	T. of execution	% of optimal solution
9	5	3 seconds	80%
9	10	5 seconds	100%
10	5	3 seconds	80%
10	10	5 seconds	100%
11	5	3 seconds	80%
11	10	5 seconds	100%

Comparing the proposed alternative with each of the separate algorithms, we can ensure that the execution time is right, regardless of the number of stops. Moreover, in situations where the optimal solution is not found, the average deviation for the optimal solution does not exceed 3% of the value of the optimal solution.

6 Conclusions and further work

The work presented is the result of a research project funded by the Basque government. The aim of the project was the optimization of on-demand bus transport systems. Our algorithm is integrated into a Web application that allows passengers to make requests via a mobile device. With these requests, using the algorithm described, we construct the bus route dynamically. In addition, if a request will not be met, the system notifies the passenger the nearest station in which he can take the bus.

During the implementation of the algorithm different software design patterns have been used. This has allowed the generation of a library for modeling and solving problems of route optimization, which may be used in future developments.

At present, we are working on the design of a methodology that facilitates the modeling of route optimization problems to take into account constraints associated with vehicles (capacity and cost of travel) and passenger preferences (time restrictions).

References

1. Garey, M. R. and Johnson, D. S: Computers and Intractability; a Guide to the Theory of Np-Completeness. W. H. Freeman & Co. (1990).
2. R M Jorgensen, J Larsen and K B Bergvinsdottir: Solving the Dial-a-Ride problem using genetic algorithms. (2004)
3. Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J. The Traveling Salesman Problem. ISBN 0691129932. (2006)
4. Dantzig, G.B.; Ramser, J.H. The Truck Dispatching Problem. Management Science Vol. 6, No. 1, October 1959, pp. 80-91.
5. D Pisinger, S Ropke: A general heuristic for vehicle routing problems. (2005)
6. Repoussis, P.P.; Tarantilis, C.D.; Ioannou, G.: Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. (2009)
7. Rutenbar, R.A.: Simulated Annealing algorithms: an overview. (2002)
8. M Gendreau, A Hertz, G Laporte: A tabu search heuristic for the vehicle routing problem. (1994)
9. Marco Dorigo, Luca Maria Gambardella: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. (1997)
10. Panagiotis P. Repoussis, Christos D. Tarantilis, George Ioannou: An Evolutionary Algorithm for the Open Vehicle Routing Problem with Time Windows. (2009)
11. L Zhang, M Yao, N Zheng: Optimization and improvement of Genetic Algorithms solving Traveling Salesman Problem (2009)