MULTI-OBJECTIVE EVOLUTIONARY APPROACH FOR OPTIMIZING A DEMAND RESPONSIVE TRANSPORT

Renan José dos Santos Viana
Departamento de Informática - Universidade Federal de Viçosa
Campus da UFV, 36.570-900, Viçosa - MG - Brasil
renan.viana@ufv.br

André Gustavo dos Santos
Departamento de Informática - Universidade Federal de Viçosa
Campus da UFV, 36.570-900, Viçosa - MG - Brasil
andre@dpi.ufv.br

José Elias Claudio Arroyo
Departamento de Informática - Universidade Federal de Viçosa
Campus da UFV, 36.570-900, Viçosa - MG - Brasil
jarroyo@dpi.ufv.br

ABSTRACT

In this paper, we address the Demand Responsive Transport (DRT) services. A DRT is a flexible transportation service that provides transport on demand, being especially useful in sparsely inhabited areas, which deal with a lack of transportation service. Users formulate requests specifying desired locations and times of pickup and delivery. The vehicle routes are planned and scheduled based on these requests minimizing a set of objectives, like costs and user inconvenience, while respecting a set of constraints imposed by the passengers and vehicles, as time windows and capacity. We adapt a formulation and propose a multi-objective evolutionary algorithm (MOEA) with feasibility-preserving operators. To compare and validate our approach, a MOEA proposed in the literature was reimplemented. Computational experiments were performed on benchmark instances and the results were analyzed by quality indicators widely used for multi-objective algorithms comparison. The proposed algorithm proved to be better in all indicators for all instances.

KEYWORDS. Demand Responsive Transport, Multiobjective, Evolutionary Algorithm.

Main Area: MH - Metaheuristics; SE - OR in Services; L&T - Logistic and Transport.

1. Introduction

Demand Responsive Transport (DRT) is a transport service operated on demand by a fleet of vehicles, which are scheduled to collect and deliver passengers in accordance with their needs (Mageean and Nelson, 2003). Users formulate requests determining intended locations and times of pickup and delivery (Ambrosino *et al.*, 2004). Usually this type of transport service is shared, i.e., many passengers can be in the same vehicle at the same time (Cordeau *et al.*, 2007).

A DRT service is specially useful in sparsely inhabited areas, which deal with a lack of transportation service, given that the service providers do not want to admit the cost of a transport service insufficiently used (Chevrier, 2008). This type of service is activated only on demand. In (Ambrosino *et al.*, 2004), three main reasons of the growth in popularity of the DRT are presented: the lack of the adaptability of conventional regular bus and taxi services; shortcomings of special transport services; and new developments in community transport. According to Mageean and Nelson (2003), there is also an interest in the potential of DRT to combat social exclusion.

Managing a DRT service in an optimized way consists in grouping the largest possible number of passengers in the same vehicle and planning the routes in the best way in order to reduce the operational costs besides respecting a number of constraints like the capacity of the vehicles and the locations and schedules of requests' pickup and delivery without decreasing the service quality (Chevrier *et al.*, 2012).

In its usual form, a DRT service can be related with the Dial-a-Ride Problem (DARP) and to increase the efficiency of these services, models and optimization techniques have been proposed for this passenger transportation problem (Chevrier *et al.*, 2012; Parragh *et al.*, 2010). The DARP consists of designing a set of vehicle routes and schedules for a number of passengers, who send pickup and delivery requests between desired origins and destinations (Cordeau and Laporte, 2007). This problem is similar to the Pickup and Delivery Problem (PDP), both belonging to the class of Vehicle Routing Problems with Pickups and Deliveries (VRPPD), where goods are transported between pickup and delivery locations. Both problems operate based on transportation requests, that consist of paired pickup and delivery points. However, there is a important difference between PDP and DARP: the first deals with the transportation of goods while the second deals with passenger transportation. Thus, DARP focuses on quality of service usually expressed by additional constraints or objectives instead of or besides others related to cost (Parragh *et al.*, 2008; Parragh *et al.*, 2010).

The main difference between a DRT service and the DARP is the flexibility. The first accepts delays on the journeys, but limiting them by constraints to keep a good quality of service. Another difference is the way in which the user sends requests: in a DARP, the users often send two requests during the same day, an outbound request from a pickup to a delivery point, and an inbound request for the return trip (e.g., from home to the hospital and return to home). In a DRT, this does not necessarily occur. For surveys on the DARP and DRT, we refer to (Cordeau and Laporte, 2007) and (Parragh *et al.*, 2010), respectively.

In the literature, there are a number of different versions of the DARP. Most papers consider a static variant of the problem, in which all requests are known in advance of the planning. In (Cordeau and Laporte, 2003), a tabu search heuristic is proposed for solving the static DARP that minimizes total routing costs, considering time windows, a maximum user ride time limit, and a maximum route duration limit as constraints. In (Chevrier *et al.*, 2010) a DRT service is addressed as a special case of the DARP adding some of its specificities (DARP applied to a DRT service) and a multi-objective formulation. To solve the problem, an evolutionary approach was proposed as well as new solution representation and variation operators. Such mechanisms were integrated in three algorithms state of the art: Non dominated Sorting Genetic Algorithm II (NSGA-II) (Ded *et al.*, 2002), Strength Pareto Evolutionary Algorithm 2 (SPEA-2) (Zitzler *et al.*, 2001) and Indicator Based Evolutionary Algorithm (IBEA) (Zitzler e Künzli, 2004). In order to intensify the search process in the solution space, Chevrier *et al.* (2012) solve the DRT problem by proposing a hybrid multi-objective evolutionary approach based on the algorithms used in (Chevrier *et al.*, 2010). The routes are improved by a Local Search strategy based on the metaheuristic Iterated Local Search (ILS) together with the local search 2-opt algorithm within the mutation operator.

Another variant of the DARP is the dynamic one, in which the routing process is done in real time. In other words, new requests may come during the planning of the routes and have to be scheduled into existing routes. A number of parallel implementations of the tabu search heuristics created by Cordeau and Laporte (2003) were proposed by Attanasio *et al.* (2004) to solve a dynamic DARP. Berbeglia *et al.* (2012) introduced a hybrid approach that is used to solve the dynamic variant of the problem combining an exact constraint programming algorithm and a tabu search heuristic.

In this paper, we present a new mathematical model adapted for a DRT service based on a static DARP model proposed in the literature and also propose a new multi-objective evolutionary algorithm (MOEA) that performs a search in the feasible search space instead of exploring the entire search space. Our goal is producing a large set of well spread non-dominated solutions close to the Pareto-optimal set. A construction heuristic to create the initial population, as well as mutation and crossover operators to produce diversity in the population and new solutions, respectively, without generating infeasible solutions were created. These modules are integrated in the Non-dominated Sorting Genetic Algorithm II (NSGA-II). The proposed approach is compared to an algorithm of the literature by means of two different solution quality indicators.

The remainder of this paper is organized as follows: Section 2 presents a formal definition of the DRT and a mathematical formulation of the problem. An overview of the concepts of multi-objective optimization, evolutionary algorithms and the NSGA-II are presented in Section 3. Section 4 shows how the NSGA-II modules were defined in our approach. Finally, the computational results and conclusion are presented in sections 5 and 6, respectively.

2. Problem Definition

The operation of a general DRT service can be formulated as a Demand Responsive Transport Problem (DRTP) and is classified as a multi-objective combinatorial optimization problem (MOCOP). The mathematical model proposed in this paper is a DARP (Cordeau, 2006) adaptation that introduces the flexibility (e.g. delays tolerance) and multi-criteria decision. In this work, the problem is addressed considering three objective functions. Given that the problem under study is a generalization of the DARP, it is also a NP-hard problem due to its complexity. Preliminary tests discourage the use of the model in exact methods for instances test with more than twenty requests.

The DRTP is modeled on a complete direct graph $G = (V, A)$, where $V = V^+ \cup V^- \cup \{0, 2n+1\}$ is the set of all vertices and A the set of all arcs. The subsets $V^+ = \{1, \ldots, n\}$ and $V^- = \{n+1, \ldots, 2n\}$ contain all pick-up and drop-off vertices, respectively. A total of n users (or requests) to be served consist of a pickup vertex i and a delivery vertex $n+i$. The nodes 0 and $2n+1$ represent the origin and destination depots.

Let K be an homogeneous fleet with k identical vehicles, each with a capacity Q. To each user $i = 1 \ldots n$ is associated a number of passengers q_i and a service duration d_i for loading or unloading operations at each vertex. For pickup and drop-off vertex we have $q_i = -q_{n+i}$ ($i = 1, \ldots, n$) and $d_i > 0$, and for the depots $q_0 = q_{2n+1} = 0$ and $d_0 = d_{2n+1} = 0$. To each arc $(i, j) \in A$ is associated a travel time t_{ij}.

Each user $i = 1, \ldots, n$ defines a desired pick-up time h_{i^+}. The time window duration w_{i^+} of a pick-up point is proportional to the journey duration $t_{i,n+i}$ from i to $n+i$, defined as: $w_{i^+} = k_w \cdot t_{i,n+i}$, being k_w a coefficient that indicates the percentage of the duration allocated to the time window. The theoretical arrival time at the delivery point h_{i^-} is the sum of the desired pick-up time and journey duration from i to $n+i$, i.e., $h_{i^+} + t_{i,n+i}$. The maximal delivery time h'_{i^-} is defined as: $h'_{i^-} = h_{i^+} + (k_r \cdot t_{i,n+i})$ being k_r a coefficient of relaxation.

The DRTP can be formulated as the following 3-index mixed integer program. The main decision variables are:

- x_{ij}^k: binary variable set to 1 if vehicle k travels arc (i, j), 0 otherwise
- v^k: binary variable set to 1 if vehicle k is used, 0 otherwise

Other auxiliary variables are used to model the constraints and objectives:

- t^k: total travel time of vehicle k if it is used, 0 otherwise
- D_i: delivery delay of customer i
- H_i^k: arrival time of vehicle k at node i, 0 if not served by the vehicle
- Q_i^k: number of passengers on the vehicle k after visiting node i, 0 if not served by the vehicle

Objective functions

$$F_1 = min \sum_{k \in K} v^k \tag{1}$$

$$F_2 = min \sum_{k \in K} t^k \tag{2}$$

$$F_3 = min \sum_{i \in V^-} D_i \tag{3}$$

Subject to:

$$\sum_{k \in K} \sum_{j \in V} x_{ij}^k = 1, \qquad \forall i \in V^+ \tag{4}$$

$$\sum_{j \in V} x_{ij}^k - \sum_{j \in V} x_{n+i,j}^k = 0, \qquad \forall i \in V^+, k \in K \tag{5}$$

$$\sum_{j \in V} x_{0j}^k = v^k, \qquad \forall k \in K \tag{6}$$

$$\sum_{j \in V} x_{ji}^k - \sum_{j \in V} x_{ij}^k = 0, \qquad \forall i \in V^+ \cup V^-, k \in K \tag{7}$$

$$\sum_{i \in V} x_{i,2n+1}^k = v^k, \qquad \forall k \in K \tag{8}$$

$$\sum_{j \in V} x_{ij}^k \leq v^k, \qquad \forall i \in V^+, k \in K \tag{9}$$

$$H_j^k \geq H_i^k + d_i + t_{ij} - M_{ij}^k(1 - x_{ij}^k), \qquad \forall i \in V, j \in V, k \in K \tag{10}$$

where $M_{ij}^k \geq max\{0, l_i + d_i + t_{ij} - e_j\}$

$$Q_j^k \geq Q_i^k + q_j - W_{ij}^k(1 - x_{ij}^k), \qquad \forall i \in V, j \in V, k \in K \tag{11}$$

where $W_{ij}^k \geq min\{Q, Q + q_i\}$

$$t^k = H_{2n+1}^k - H_0^k, \qquad \forall k \in K \tag{12}$$

$$D_i \geq max\{0, H_i^k - h_{i^-}\}, \qquad \forall i \in V^-, k \in K \tag{13}$$

$$v^{k-1} \geq v^k, \qquad \forall k \in K \backslash 1 \tag{14}$$

$$v^k \in \{0, 1\}, \qquad \forall k \in K \tag{15}$$

$$x_{ij}^k \in \{0, 1\}, \qquad \forall i \in V, j \in V, k \in K$$

$$t^k \geq 0, \qquad \forall k \in K$$

$$D_i \geq 0, \qquad \forall i \in V^-$$

$$h_{i^+} \leq H_i^k \leq h_{i^+} + w_{i^+}, \qquad \forall i \in V^+, k \in K$$

$$h_{i^-} \leq H_i^k \leq h'_{i^-}, \qquad \forall i \in V^-, k \in K$$

$$max\{0, q_i\} \leq Q_i^k \leq min\{Q, Q + q_i\}, \qquad \forall i \in V, k \in K$$

The objective functions (1, 2 and 3) minimize respectively the number of vehicles used, the journeys durations and delays. The constraints (4) and (5) guarantee that each request is served exactly once and that the same vehicle visits the origin and destination nodes. The constraints (6), (7) and (8) ensure that if the vehicle k is used, its route starts at the origin depot and ends at the destination depot and that if the vehicle visits a node, it must leave that node. The constraint (9) ensures that the vehicle will be defined as used if it serves any request. The constraints (10) and (11) guarantee the consistency of time and occupancy of a vehicle while it travels through its route. The equalities (12) and (13) define the route time of each vehicle and the delay at each delivery node. Constraint (14) remove symmetry in the use of vehicles. Finally the set of constraints (15) defines the binary variables and the bounds of each other variable, ensuring that each user is serviced within its defined time window and that vehicle capacity is respected at each node.

3. Multi-Objective Optimization

In a multi-objective optimization problem (MOOP), there are a number of objective functions, which are to be minimized or maximized. These objectives are usually in conflict with each other. Furthermore, a feasible solution must satisfy both a number of constraints and variable bounds imposed by the problem. The MOOP can be defined by a set of m objective functions $f = (f_1, f_2, \dots f_m)$, a set X of feasible solutions in the decision space and a set Z of feasible points in an objective space $Z = f(X)$. A solution $x = (x_1, x_2, \dots x_n)$ is a vector of n decision variables in the decision space. For each solution $x \in X$, there exists a point $z \in Z$, denoted by $f : X \to Z$ with $z = f(x) = (f_1(x), f_2(x), \dots, f_m(x))$. The main goal of a MOOP is finding solutions in the decision space optimizing (minimizing or maximizing) m objectives (Chevrier *et al.*, 2012; Coelho *et al.*, 2007; Deb, 2001; Dhaenens *et al.*, 2010).

Generally, multi-objective optimization algorithms use the concept of dominance to define that one solution x is better than other solution x'. This occurs if two conditions holds:

- The solution x is no worse than x' in all objectives.
- The solution x is strictly better than x' in at least one objective.

If any of the conditions is violated, the solution x does not dominate the solution x'. A solution $x \in X$ is called Pareto-optimal when it is not dominated by any other solution of the decision space. Such solution is also called efficient. The set of all efficient solutions is the efficient set or Pareto-optimal set. The set of all non-dominated vectors is the Pareto Front.

Instead of finding a good approximation of the global optimum solution like in a single-objective optimization, in a MOOP there are two goals: find a set of solutions as close as possible to the Pareto-optimal front and find a set of solutions as diverse as possible. An efficient multi-objective optimization algorithm must work on satisfying both of them, although these tasks are somewhat complicated to do simultaneously (Deb, 2001). Generating the entire efficient set is usually a difficult task due to the complexity of the MOOPs. Therefore, often the main goal is finding a good approximation of it with good diversity.

3.1. Evolutionary Algorithms

Evolutionary algorithms (EA) are optimization methods, which are based on natural evolutionary principles to compose search and optimization procedures (Deb, 2001). These methods iteratively simulate the evolution of a set of solutions, utilizing the notion of competition, where the individuals with better fitness survive for next generations (Talbi, 2009). Based on biology, the EAs are composed by mechanisms of reproduction, mutation, combination, and selection. Their success is motivated by the ability in solving difficult optimization problems in various domains and has been successfully applied to many real and complex problems (Talbi, 2009). Among these problems, there are the multi-objective ones, with more than one objective that have to be simultaneously optimized. In this paper, the multi-objective evolutionary algorithm (MOEA) Non-dominated sorting genetic algorithm II (NSGA-II) was chosen to solve the DRTP.

3.2. NSGA-II

The NSGA-II (proposed in (Deb *et al.*, 2002)), is one of the most widely used algorithms for solving multi-objective optimization problems. This algorithm uses the fast non-dominated sort method to rank the solutions of a population into several non-domination classes called fronts. To ensure the diversity in the population, the NSGA-II uses a crowded-comparison approach that calculates an estimate of the density of solutions surrounding each solution in the population. Both these approaches allow the NSGA-II to assign to each solution the rank and the crowding distance values which represent the quality of the solution in terms of convergence to the Pareto-optimal set and in terms of diversity, respectively. Given these two values, one solution x is chosen over another solution y if it has a better rank value. If both have the same rank value, x is chosen over y if it has a better crowding distance.

In the NSGA-II, firstly, a random initial population P_0 of size N is created. Generically, all solutions of the parent population P_t are sorted into different non-dominated classes. An offspring population Q_t of size N is created using binary tournament selection, recombination and mutation operators applied to P_t. The two populations are combined together to create R_t of size $2N$, then the population R_t is classified using the non-dominated sorting. The construction of the new population P_{t+1} of size N starts with the best non-dominated front of R_t, continues with the second non-dominated front, and so on. This process ends when the current front can not be entirely accommodated with previous fronts in N slots, in other words if there are more solutions in the current front than remaining slots in the new population. Then, the solutions of the current front with higher crowding distance values are chosen to complete the new population P_{t+1}.

4. NSGA-II for the DRTP

In this paper the multi-objective evolutionary algorithm NSGA-II is used to solve the DRTP. A new population initialization and operators are proposed here for compose a new NSGA-II approach. To validate the proposed approach, the NSGA-II described in (Chevrier et al., 2012) was reimplemented. In the following, the NSGA-II modules proposed in this paper are detailed. Figure 1 shows a flowchart of how the proposed NSGA-II works.

4.1. Solution Representation

The solution representation choice is an important step that influences all the remainder modules design of an optimization method. In this paper the same representation proposed by Chevrier *et al.* (2012) is used. A solution is composed by a set of routes that starts and ends at the depot point. Generally, a route is a set of visited points. Here the vehicle journey is represented by a sequence of request identifiers. The first occurrence of a request identifier represents the pick-up point r^+ and the second one is the delivery point r^-. The sequence of request identifiers of a route literally represents the sequence of respective points of pickup or delivery that the vehicle will visit. Since the solutions are vectors of routes, which are vectors of request identifiers, the best solution encoding is a vector of vectors. Note that each individual of the population is a problem solution.

4.2. Population Initialization

To generate the initial population, a slightly more sophisticated procedure than proposed by (Chevrier *et al.*, 2012) is used. Here, the initialization strategy considers the desired pickup time and closeness to the depot during the procedure while (Chevrier *et al.*, 2012) constructs randomly the initial population. Since the vehicles are initially in the depot, all requests are firstly sorted according to the desired pickup time and the distance between the depot and the request pickup point. The nearest and the most urgent requests will have priority. Then, the first m requests on the list are assigned to a distinct vehicle. After that, the other requests are sorted according only to desired pickup time and if possible, randomly added to an existing vehicle route. Otherwise, a new vehicle route is created to serve the request. The number of vehicle routes may vary between m and the number of requests received.

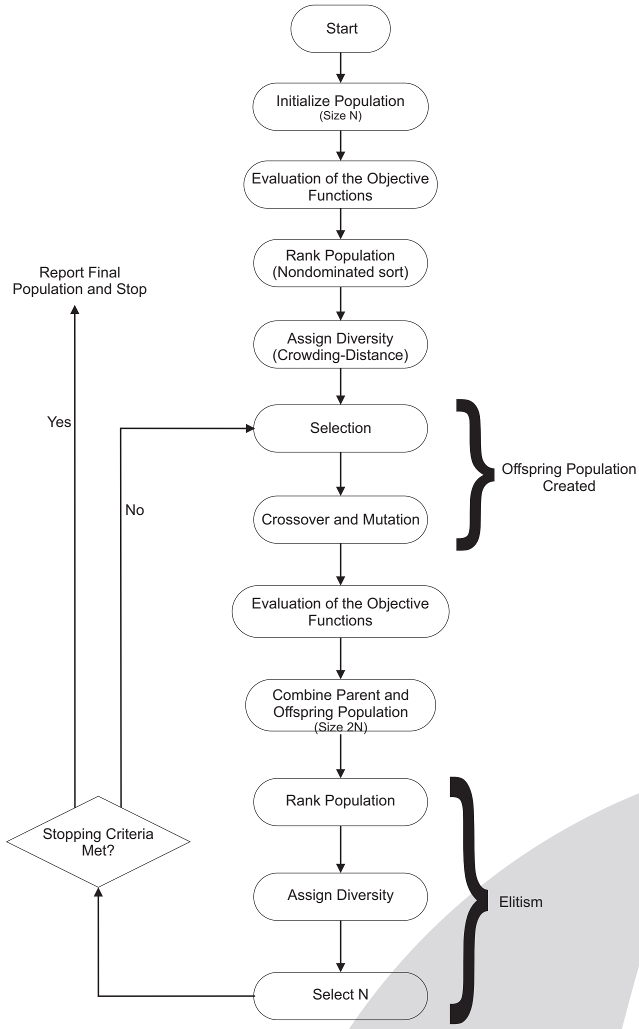

Figure 1: Flowchart of the NSGA-II algorithm

4.3. Selection Operator

The selection operator, also called reproduction operator, performs a selection process, that imitates natural selection by giving to good solutions higher opportunity to breed. These solutions are placed in a memory called mating pool, in which the variation operators (crossover and mutation) are applied. Here, the adopted strategy is the binary tournament selection that performs a contest between two randomly chosen solutions from the population. A solution wins if it has a smaller rank value (better convergence), or in case of equality, if it has a larger crowding distance value (better diversity).

4.4. Crossover Operator

The crossover operator produces new solutions called offspring by the exchange of information between two parents of the current population. The operator proposed here, firstly chooses two parents ($P1$ and $P2$) through the selection operator and makes a copy of both solutions to create the offspring ($C1$ and $C2$). In order to modify $C1$, a route in $P2$ is randomly chosen, and to avoid duplicated data, all requests of this route are removed from $C1$. If any route of $C1$ becomes empty, it is deleted. The second step is performed in a way that the offspring solutions do not become infeasible. The route chosen from $P2$ is inserted in the same position in $C1$ and the counterpart route and all subsequent routes are moved forward. The production of $C2$ follows the same process by inverting the parent solutions. To control the possibility of performing a crossover, a probability pc called crossover rate is defined. The algorithm 1 shows how the operator was designed.

Algorithm 1 Crossover Operator (population)

```
P1 ← RandomSolution (population)
repeat
    P2 ← RandomSolution (population)
until P1 ≠ P2
C1 ← CopySolution(P1)
C2 ← CopySolution(P2)
randomRoute1 ← RandomRoute (P2)
for all requests of randomRoute1 do
    for all routes of C1 do
        if Contains(route, request) then
            RemoveRequest (route, request)
            break
InsertRoute (C1, randomRoute1)
randomRoute2 ← RandomRoute (P1)
for all requests of randomRoute2 do
    for all routes of C2 do
        if Contains(route, request) then
            RemoveRequest (route, request)
            break
InsertRoute (C2, randomRoute2)
```

4.5. Mutation Operator

The main goal of a mutation operator is to introduce diversity in the current population. In this paper we propose a mutation operator that performs a swap of a single request between different routes, keeping the solution feasibility. Randomly, a single request is selected and removed from a route. After that, a different route is chosen to insert the removed request. If the swap results in an infeasible solution, a subsequent route is tried until the swapping is feasible. If no route can accommodate the request, a new route is created to assign the costumer. Like in the crossover operator, to control the possibility of performing a mutation, a probability pm called mutation rate is defined. The algorithm 2 shows how the operator was designed.

Algorithm 2 Mutation Operator (population)

```
solution ← RandomSolution (population)
route1 ← RandomRoute (solution)
repeat
    route2 ← RandomRoute (solution)
until route1 ≠ route2
request ← RemoveRandomRequest (route1)
i ← RouteIndex (solution, route2)
repeat
    if i ≠ RouteIndex (solution, route1) then
        if FeasibleInsert(solution, i, request) then
            break
    i ← i + 1
until i > RouteNumbers (solution)
if i > RouteNumbers (solution) then
    NewRoute(solution, request)
```

Given the route index and the request identifier, the Feasible Insert function tries to insert the request in the route so that it continues feasible. Both in the insertion of the pickup point as in the delivery point, the scan is started in the last position of the route and proceeds backwards up to the start of the route. The insertion of the two points are performed on the first viable position found, so that it does not cause big delays on subsequent visits and therefore, bring on less impact on the service quality. The algorithm 3 describes how the operator was designed.

5. Computational Experiments

In this section, we discuss the results obtained using two sets of test instances. The proposed and the literature approaches were coded in C++. The mathematical programming model was run in CPLEX 12.6 and implemented using Concert C++ library. The computational tests were run on a 3.40 GHz Intel Core i5 computer, with 16 GB RAM running Windows Seven.

Algorithm 3 FeasibleInsert ($solution, routeIndex, request$)

```
route ← GetRoute(solution, routeIndex)
i ← RouteEnd(route)
while i >= 0 do
    if FeasibleInsertPickupPoint(i, request) then
        j ← RouteEnd(route)
        while j ≠ i do
            if FeasibleInsertDeliveryPoint(j, request) then
                return true
            j ← j − 1
    i ← i − 1
return false
```

5.1. Benchmark Test Instances

In order to evaluate the proposed and literature approaches, computational experiments were realized using two sets of test instances introduced by Chevrier *et al.* (2012). The first set, called "`Rnd100`" has 10 instances with an almost homogeneous distribution of customers, which contain 100 requests randomly generated. The second set, denoted "`Gravit100`" contains 10 instances with a non-homogeneous distribution of customers, that have 100 requests generated using a geographical model of people or freight flows.

5.2. Performance Assessment

For the purpose of evaluating the quality of the approximation sets obtained in the computational experiments and of comparing the approaches, the proposed and the literature algorithms were run 10 times and the 10 outcomes of each algorithm were stored in sets A and B, respectively. After that, two reference sets (Ra, Rb) were created with all non-dominated points of the sets A and B, respectively. Also, a reference set Ref containing all non-dominated points from the union of the sets Ra and Rb was created.

5.2.1. Solution Quality Indicators

The performance assessment of the two approaches was performed using two indicators frequently employed in the literature, which Zitzler *et al.*(2003) states that are the best suited, given that they provide compatibility and completeness to most of the dominance relations. The first is the unary additive ϵ-indicator I_{ϵ^+} proposed by Zitzler *et al.*(2003), which is based on the binary additive I_{ϵ^+}. It represents the minimum factor ϵ that any objective vector in a reference set has to be added to obtain a set that is dominated by the analyzed approximation set. Due the complexity of the problem, the Pareto front for all test instances is not known. Therefore, the reference set Ref is used instead. For each instance, both $I_{\epsilon^+}(A, Ref)$ and $I_{\epsilon^+}(B, Ref)$ were calculated. Before that, a normalization of the objective function values was performed in order to adjust the scale and provide an equal interval for all objective functions. The second quality indicator is the set coverage metric C (Zitzler e Thiele, 1998). The C metric maps the ordered pair (X, Y) to the interval $[0, 1]$ (Zitzler, 1999). The $C(Ra, Rb)$ calculates the proportion of solutions in the reference set Rb, which are weakly dominated (covered) by solutions of the reference set Ra. The two directions have to be considered, since $C(Ra, Rb)$ is not necessarily complementary to $C(Rb, Ra)$. Note that both indicators values are to be minimized. Other information provided is the number of solutions in the sets Ra and Rb on each test instance, which represents the cardinality of the approximation set achieved by each algorithm.

5.2.2. Parameters Setting

For both approaches, the population size was defined as 100. All runs were performed during 1 minute, given that decision makers on real-time services usually need information in a short time. In order to calibrate the mutation and crossover rates, while the rate of a operator was fixed (50%), three different rates (20%, 50% and 80%) were tested and compared for the other operator. This process was performed for both operators.

Table 1: $I_{\epsilon+}$ for different crossover and mutation probabilities

Rate	0.2	0.5	0.8
Crossover	0,0958	0,1025	0,1097
Mutation	0,1228	0,0989	0,0794

The table 1 shows the performance of the proposed approach (regarding the unary additive ϵ-indicator) for all mutation and crossover rates compared. We can see that the lowest averages $I_{\epsilon+}$ were obtained for crossover probability 20% and mutation probability 80%. Despite the fact that mutation probability is usually small in the literature, our probability is high as stated by the calibration. The parameter m, used to generate the initial population, was defined on 10% of the requests. This value must be low in order to avoid solutions with a large number of unnecessary routes.

5.2.3. Computational Results

Performing the mathematical model, no exact results were found for any instance within 24 hours of execution, not even when the model was run for a single objective. However, the model was able to find and prove the optimality of solution for smaller instances, with a subset of 20 out of 100 requests, when the objective function was set to minimize the number of vehicles. In this case, the non-dominated solution sets found by the two NSGA-II approaches include solutions with this minimum number of vehicles in all runs. The quality of the other objectives could not be attested because the model did not find or prove optimal values. Then, for now on the approaches are compared to each other.

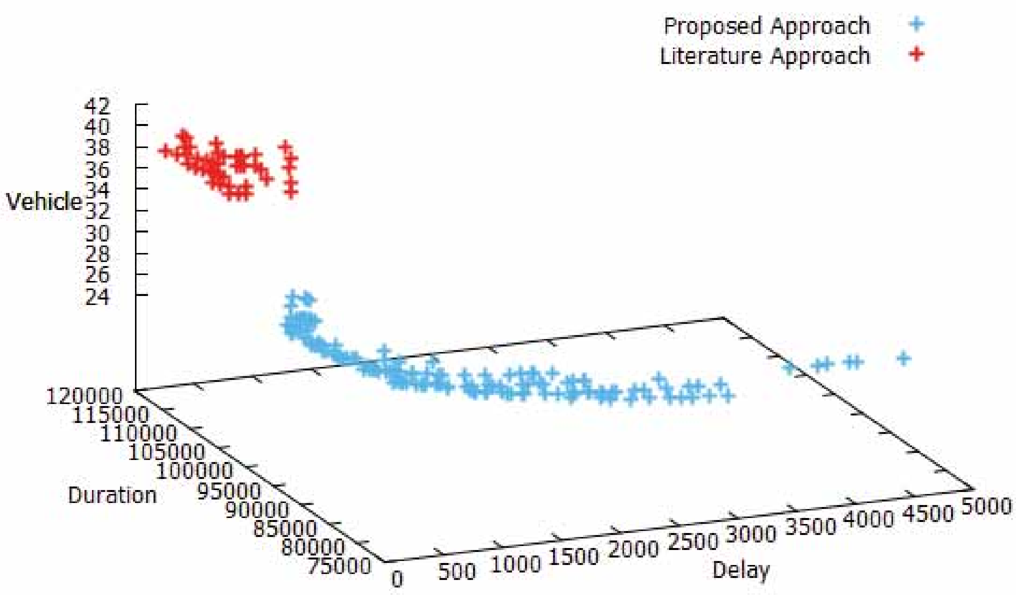

Figure 2: Proposed NSGA-II and literature NSGA-II reference sets of the Gravit100_0 instance test

The comparison of the two approaches is performed according to the previously explained methodology. Every test instance was run 10 times during 1 minute. The table 2 shows that in all test instances the proposed approach obtained average $I_{\epsilon+}$ values smaller than the ones of the literature approach. We can conclude that the proposed approach found better approximation sets, given that

Table 2: Average of the $I_{\epsilon+}$ and C results and cardinality of the reference sets

Instance	$I_{\epsilon+}$		C		$\|R\|$	
	Proposed	Literature	Prop/Lit	Lit/Prop	Prop	Lit
Gravit100_0	0,1717	0,7790	1,0000	0,0000	135	43
Gravit100_1	0,0999	0,7378	1,0000	0,0000	166	39
Gravit100_2	0,1004	0,7575	1,0000	0,0000	136	32
Gravit100_3	0,1067	0,8265	1,0000	0,0000	150	23
Gravit100_4	0,1415	0,7888	1,0000	0,0000	173	19
Gravit100_5	0,0949	0,7734	1,0000	0,0000	188	20
Gravit100_6	0,1169	0,7184	1,0000	0,0000	201	57
Gravit100_7	0,1193	0,7366	1,0000	0,0000	175	40
Gravit100_8	0,1044	0,7631	1,0000	0,0000	171	69
Gravit100_9	0,1790	0,7914	1,0000	0,0000	127	25
Rnd100_0	0,0519	0,6373	1,0000	0,0090	221	45
Rnd100_1	0,0303	0,6250	0,9770	0,0045	220	87
Rnd100_2	0,0340	0,6310	1,0000	0,0049	204	55
Rnd100_3	0,0297	0,6531	1,0000	0,0145	207	42
Rnd100_4	0,0329	0,6621	1,0000	0,0049	205	62
Rnd100_5	0,0325	0,6026	0,9663	0,0141	213	89
Rnd100_6	0,0478	0,6508	0,9762	0,0123	243	84
Rnd100_7	0,0477	0,6809	1,0000	0,0117	256	63
Rnd100_8	0,0484	0,6734	1,0000	0,0250	240	67
Rnd100_9	0,0346	0,6570	1,0000	0,0046	218	65

the indicator values are closer to zero. The results $C(Ra, Rb)$ and $C(Rb, Ra)$ show that in all "`Gravit`" instances Ra covers Rb. For the "`Rnd`" instances, the $C(Ra, Rb)$ and $C(Rb, Ra)$ were next to 1 and 0, respectively, indicating the superiority of our approach. By the Figure 2 it is possible to see the quality difference between the two reference sets obtained applying both algorithms in a test instance. The number of the non-dominated solutions obtained in each test instance shows that, considering the set cardinality factor, the proposed algorithm also reached better results than the literature algorithm.

The improvement obtained by our approach is due to the new operators here proposed. They generate only feasible solutions. Given that the problem has many complex constraints, the crossover and mutation operator may have a small chance to remove infeasibility of solutions based solely on penalties using simple blind exchanges. This results in a slow convergence and a worse set of non-dominated solutions.

6. Conclusion

We can conclude that the multi-objective genetic algorithm is a good approach to solve this complex problem. For all exact solutions that the model reached, both GA approaches converge to solutions with at least the same objective function value (number of vehicles). Our approach proposes new operators that keep the feasibility of the solutions. According to the two quality indicators widely used in the literature, the proposed approach performed better than the literature approach that works with infeasible solutions. For some test instances, the reference set found by our approach covers the reference set found by the literature approach. Therefore, the focus on the feasible search space has brought improving in the quality of the solutions. Future works include the use of other metaheuristics, mainly those with selection based on quality indicators in order to find a better reference set for larger instances. Other direction is the use of integer programming techniques to improve the model and explore the dynamic variant of the problem.

Acknowledgement

The authors thank to CAPES, CNPq and FAPEMIG for the partial financing of the project. The first author worked on this project funded by CAPES.

References

Ambrosino, G., Nelson, J. D. and Romanazzo, M. (2004), Demand responsive transport services: Towards the flexible mobility agency, ENEA, Italian National Agency for New Technologies, Energy and the Environment.

Attanasio, A., Cordeau, J. F., Ghiani, G., and Laporte, G. (2004), Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem, Parallel Computing, 30(3), 377-387.
Berbeglia, G., Cordeau, J. F. and Laporte, G. (2012), A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem, INFORMS Journal on Computing, 24(3), 343-355.
Chevrier, R., Liefooghe, A., Jourdan, L. and Dhaenens, C. (2010), On optimizing a demand responsive transport with an evolutionary multi-objective approach, Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on, 575-580.
Chevrier, R., Liefooghe, A., Jourdan, L. and Dhaenens, C. (2012), Solving a dial-a-ride problem with a hybrid evolutionary multi-objective approach: Application to demand responsive transport, Applied Soft Computing, 12(4), 1247-1258.
Chevrier, R., Optimisation de Transport à la Demande dans des territoires polariss, PhD thesis, Universit dAvignon et des Pays de Vaucluse, Lille, 2008.
Coello, C. C., Lamont, G. B. and Van Veldhuizen, D. A., Evolutionary algorithms for solving multi-objective problems, Springer Science & Business Media, 2007.
Cordeau, J. F. and Laporte, G. (2003), A tabu search heuristic for the static multi-vehicle dial-a-ride problem, Transportation Research Part B: Methodological, 37(6), 579-594.
Cordeau, J. F. and Laporte, G. (2007), The dial-a-ride problem: models and algorithms, Annals of Operations Research, 153(1), 29-46.
Cordeau, J. F., Laporte, G., Potvin, J. Y., and Savelsbergh, M. W. (2007), Transportation on demand, Handbooks in operations research and management science, 14, 429-466.
Cordeau, J. F. (2006). A branch-and-cut algorithm for the dial-a-ride problem, Operations Research, 54(3), 573-586.
Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002), A fast and elitist multiobjective genetic algorithm: NSGA-II, Evolutionary Computation, IEEE Transactions on, 6(2), 182-197.
Deb, K., Multi-objective optimization using evolutionary algorithms, John Wiley & Sons, 16, 2001.
Dhaenens, C., Lemesre, J. and Talbi, E. G. (2010), K-PPM: A new exact method to solve multi-objective combinatorial optimization problems, European Journal of Operational Research, 200(1), 45-53.
Mageean, J. and Nelson, J. D. (2003), The evaluation of demand responsive transport services in Europe, Journal of Transport Geography, 11(4), 255-270.
Parragh, S. N., Doerner, K. F. and Hartl, R. F. (2008), A survey on pickup and delivery problems Part II: transportation between pickup and delivery locations, Journal für Betriebswirtschaft, 58(1), 81-117.
Parragh, S. N., Doerner, K. F. and Hartl, R. F. (2010), Demand responsive transportation, Wiley Encyclopedia of Operations Research and Management Science.
Talbi, E. G., Metaheuristics: from design to implementation, John Wiley & Sons, 74, 2009.
Zitzler, E. and Künzli, S. (2004), Indicator-based selection in multiobjective search, Parallel Problem Solving from Nature-PPSN VIII, Springer Berlin Heidelberg, 832-842.
Zitzler, E. and Thiele, L. (1998), Multiobjective optimization using evolutionary algorithms - a comparative case study, Parallel problem solving from nature - PPSN V, Springer Berlin Heidelberg, 292-301.
Zitzler, E., Laumanns, M. and Thiele, L. (2001), SPEA2: Improving the strength Pareto evolutionary algorithm, Tech. Rep. 103, Computer Engineering and Networks Lab (TIk), Swiss Federal Institute of Technology (ETH), Zurich, Switerland.
Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M. and Da Fonseca, V. G. (2003), Performance assessment of multiobjective optimizers: an analysis and review, Evolutionary Computation, IEEE Transactions on, 7(2), 117-132.
Zitzler, E., Evolutionary algorithms for multiobjective optimization: Methods and applications, Ithaca: Shaker, 63, 1999.