Journal of Universal Computer Science, vol. 14, no. 15 (2008), 2512-2531
submitted: 18/2/08, accepted: 30/6/08, appeared: 1/8/08 © J.UCS

Bus Network Optimization with a Time-Dependent Hybrid Algorithm

Ana C. Olivera
(LIDeCC, Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Av. Alem 1253, B8000CPB Bahía Blanca, Argentina
aco@cs.uns.edu.ar)

Mariano Frutos
(Departamento de Ingeniería
Universidad Nacional del Sur
Av. Alem 1253, B8000CPB Bahía Blanca, Argentina
mfrutos@uns.edu.ar)

Jessica A. Carballido
(LIDeCC, Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Av. Alem 1253, B8000CPB Bahía Blanca, Argentina
jac@cs.uns.edu.ar)

Nélida B. Brignole
(LIDeCC, Planta Piloto Ingeniería Química (PLAPIQUI)
Universidad Nacional del Sur, Complejo CCT-UAT,CONICET
Camino la Carrindanga Km. 7, 8000 Bahía Blanca, Argentina
dybrigno@criba.edu.ar)

Abstract: This paper describes a new hybrid technique that combines a Greedy Randomized Adaptive Search Procedure (GRASP) and a genetic algorithm with simulation features in order to solve the Bus-Network Scheduling Problem (BNSP). The GRASP is used as an initialization method to find the routes between bus stops. The Genetic Algorithm is used to find the whole configuration of the bus network, together with a simulation tool that finds the values of the environmentally dependent dynamic variables. The new method was tested with an academic case of study, and the results clearly satisfy the requirements of both the transport user and the transport operator.

Keywords: Hybrid Genetic Algorithms, Bus Network Scheduling Problem, Optimization
Categories: I.6.3, I.2.m

1 Introduction to the Bus-Network Scheduling Problem

The Bus Network Scheduling Problem (BNSP) deals with the task of finding a bus network that fulfills several objectives. In a precise definition for a bus network it is necessary to consider several **entities**: the **user**, who is a passenger of the buses, the authorities that impose the system regulations and the **operator** of the lines. These

entities usually have different expectations for the bus network, which are generally confronted.

The scheduling of a bus network involves the determination of many aspects: routes, frequencies, time schedules, fleet size and number of employees [Bielli, 02], [Zhao, 05], [Zhao, 06]. The whole process can be decomposed in several **activities** [Ceder, 86] as follows:

1) Design of a route for the determination of the number of lines and paths,
2) Determination of line frequencies,
3) Determination of transfer times and synchronicity,
4) Assignment of available fleet,
5) Assignment of employees and other resources for each line.

The first two stages comprise the most important ones since the results of them are directly used to achieve the other three activities. The initial crucial activity is to find the route's design as an arrangement of lines and routes. The second one is to define the frequencies that vary according to the period of the day (midday, midnight, rush hour, etc) and the synchronicity of transfers, fleet size, resources and employees, which should be assigned for each line. Together with these requirements, the BNSP also involves many objectives, such as the maximization of quality of the service (i.e., minimization of traveling and waiting times) and the maximization of the benefit for the transport operator.

It is important to notice that BNSP solving, based on decision-support tools, is growing constantly everywhere, not only being important in developing countries. The main challenging edges of this problem lie on its NP-complexity [Ceder, 98], [Baaj, 91], economic and social interests, and technical difficulties. Furthermore, an extra hindrance is constituted by the need to consider temporal features in the model.

In this work, we have focused on the activities of line-scheduling and route-design (activities 1 and 2), and the entities user and operator are considered with the same level of importance. In essence, several stages were established for the Time-Dependent Hybrid Algorithm, by means of the combination of a Genetic Algorithm (GA), a Greedy Randomized Adaptive Search Procedures (GRASP) and a Simulation tool. The last component was included in order to proportion representative time-related elements for the calculation of fitness values, which are necessary to carry off the dynamics of the real scenarios with precision.

1.1 State of the Art

A realistic design of a bus network generally requires the minimization of several conflicting objectives under complex constraints. This feature characterizes our problem instance as a Multi-Objective Problem (MOP). Different methods have been proposed to solve the BNSP: mathematical optimization, heuristics [Lampkin, 67], [Byrne, 72], [Rapp, 76], [Chang, 89], [Ceder, 97], and several meta-heuristics like genetic algorithms, ant-colonies, simulated annealing, and combinations of them [Aarts, 97], [Yongshuang, 06], [Zhao, 06], [Cheng-Fa, 04], [Bielli, 02], [Bookbinder, 92].

The approaches based on mathematical optimization usually have rigorous problem statements, and a complete solution search space. However, these strategies may be too sensitive to the different settings of certain design parameters [Bookbinder, 92], [Bielli, 02]. Moreover, they also have drawbacks such as rapid convergence to a unique solution, which is clearly not desired in the case of MOPs.

The same happens with traditional heuristic approaches [Pearl, 84] like greedy algorithms, rollout algorithms, taboo search and simulated annealing [Bertsekas, 98], [Wolsey, 98]. Even more, these techniques usually obtain a local optimal, a sub-optimal or a unique solution that satisfies a number of requirements, thus their results are not global in most cases or - what is worse - they are not even locally optimal [Mandl, 80], [Baaj, 91], [Shin, 94].

Finally, the biologically inspired meta-heuristics like genetic algorithms are also used for solving the BNSP [Pattnaik, 98]. The main advantage of GA-based methods is that they can be naturally used to tackle MOPs, since they work with a population of individuals. Also, it is easy to incorporate constraints into a solution-searching GA process. However, GAs have their shortcomings too. They cannot guarantee that the optimal solution will be found, and in large-scale problems, they need to search huge solution spaces with an associated high cost. Though, they are especially suitable for MOPs since they naturally yield several solutions, instead of one.

The employment of all the aforementioned techniques to tackle the BNSP presents the following weaknesses. The stochastic process of arrival of the passengers is often ignored, or it is considered in a very simplified manner. Then, the formulation of the problem is extremely elementary. In order to attain better results, the tendency in the last decade is to combine two or more meta-heuristics with the objective of dealing with the limitations of the existing studies [Liu and Abraham, 07], [Zhao, 06], [Yongshuang, 06]. Therefore, in this work we have integrated several techniques, having a multi-objective GA as the central part of the whole procedure. As it was aforementioned, the major novelty consists in the inclusion of simulation features for the calculus of the fitness function. It is important to note that there are no earlier procedures that model the handling of dynamic variables. Hence, a comparison with other existing methodologies used for the BNSP cannot be performed.

This paper is structured in seven sections. In section two, the equations that are used to model the entities that participate in the bus-network optimization problem are introduced. Then, in the main section of the article, each stage of the hybrid procedure is presented: the GRASP algorithm and the GA are described; ending with the presentation of the Time-Dependent Hybrid Algorithm as an integration of them. The fourth section contains the experimental studies based on an academic case of study, and an analysis of the results. Finally, the last section discusses the conclusions, and outlines some speculations about future work on this issue.

2 Modeling the entities

Due to its structural complexity, methods that have previously tackled this problem do not contemplate the randomness of real systems. The strategy presented in this work is based on the model proposed by Gruttner [Gruttner, 02]. The objectives of the problem, related to each of those entities, are modeled through equations, which will be used by the GA to guide the search. The equations associated to the operator can be expressed as follows:

$$\sum_{L=1}^{M} FO_L = \sum_{L=1}^{M} \left(IO_L - CO_L \right) \qquad (1)$$

$$IO_L = AF_L \; T_L \tag{2}$$

$$CO_L = D_L \; K_L \tag{3}$$

where

AF_L: Total client influx for line L,

T_L: Price for a journey in line L,

D_L: Travel distance for each line L,

K_L: Unitary operative cost of operation per kilometer for line L.

In Equation 1, FO_L corresponds to the economic benefit to the operator. Equation 2 represents the total income for line L (IO_L), taking into account the total influx for L (AF_L) and the price per trip in line L (T_L). In contrast, Equation 3 (CO_L) is the cost the operator has to assume, considering traveling distance (D_L) and the unitary operative cost per kilometer (K_L).

The equations associated to the client are defined as follows:

$$\sum_{L=1}^{M} FU_L = \sum_{L=1}^{M} CU_L = \\ = \sum_{L=1}^{M} \left[\sum_{i=1}^{N} \sum_{j=1}^{N} \left(\delta t^{A}_{ijL} + t^{J}_{ijL} + \eta t^{W}_{ijL} \right) \; VST_{ijL} \; V_{ijL} \right] \tag{4}$$

Where,

VST_{ijL}: Subjective value for the time corresponding to each pair origin-destiny (i,j) that employs line L,

V_{ijL}: Number of journeys for each pair (i, j) that employs line L,

t^{A}_{ijL}: Access time for line L,

t^{J}_{ijL}: Journey time for line L,

t^{W}_{ijL}: Waiting time for line L,

δ y η: Relative weights between access time and waiting time with respect to the journey time.

In Equation 4, FU_L represents the cost related to transporting a client in line L. The customer's cost CU_L considers the access time and the queue for line L.

From the formulation of these equations arises the need to model the attribute of time. For this reason, in a later subsection this feature will be explained in the context of the calculation of the fitness function in the GA that constitutes de core of the strategy presented in this paper.

3 The Time-Dependent Hybrid Algorithm

The main stages of the algorithm presented in this article comprise: the initialization, which constitutes the estimation of the paths between each pair of bus stops and the corresponding distances; and the core, which yields the entire bus network. The Hybrid Algorithm defines the routes and distances from any pair of bus stops by means of the GRASP method. Then, it's the turn of the multi-objective GA, which is based on the Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) proposed

by Deb [Deb, 00] [Deb, 02] and performs the most important task, assisted by the simulation tool. The general layout of the whole procedure is depicted in Figure 1.

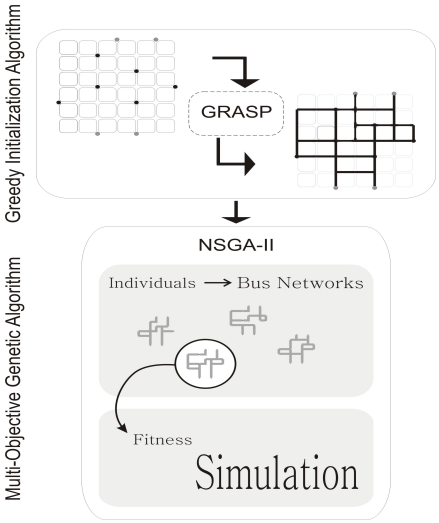

Figure 1: Time-Dependent Hybrid Algorithm general layout

3.1 GRASP Stage

An important meta-heuristic that has recently arisen is the Greedy Randomized Adaptive Search Procedure (GRASP). It is an exploratory local search method used to solve combinatorial optimization problems [Pitsoulis, 01]. In its simplest version, it is a procedure whose iterations consist of two phases: a constructive phase, and a local search procedure, where solution neighborhoods are examined until a local optimum is found. The solution is constructed step by step, by adding one new element at a time. The next element is randomly chosen from a candidate list.

For the BNSP treatment, a classification of a node map was introduced. A node is any point in the map. In particular, **bus stops** are special nodes since it is important to know the distance between one bus stop and the others. When a line has the same bus stop in its route as another line, then this shared bus stop is called a **transfer point**. This difference between bus stops and transfer points is crucial for the understanding of the whole procedure.

3.1.1 Randomized Step

The GRASP constitutes the initialization stage and it works on two bus stops for a given graph that contains all the nodes. It generates a sub-optimal route of nodes between both bus stops, and it also yields the distance between them. Given S, a partial feasible route between two bus stops, and k, the last node visited in the partial route, the algorithm selects the possible edges that contain k as their origin. Then, the probabilities of the edges to be selected in order to integrate S are set. For each node in the set of final nodes associated to the arcs e which depart from node k, the

selection probability contemplates the distance to the destination bus stop *j* (see Algorithm 1).

```
Algorithm GRASP
Input: i,j:  BusStop, iterMax:integer, beta: real;
OutPut:   bestRoute Route, bestDist: real;
Var:      alpha,distance: real;
          e: set of edges;
          k: node;
          n: integer;
          r: Route;
1.        bestDist:= infinity;
2.        for count := 1 to iterMax
3.                  alpha := (beta / count)
4.                  e := outEdges(i)
5.                  CreateProbabilities(e,alpha)
6.                  r(i,j).Actual := SelectAleatory(e)
7.                  Repeat
8.                      r(i,j).Distance := r(i,j).Distance
                            + r(i,j).Actual.EdgeDistance
9.                      r(i,j).ListAdd(Route(i,j).Actual)
10.                     k := r(i,j).Actual.finalNode
11.                     e := outEdges(k)
12.                     CreateProbabilities(e, alpha)
13.                     r(i,j).Actual := SelectAleatory(e)
14.                 until k = j
15.                 r(i,j) := localSearch(r(i,j))
16.                 if r(i,j).Distance < bestDist then
17.                         bestRoute := r(i,j)
18.                         bestDist := r(i,j).Distance
19.                 end if
20.       end for
21.       return bestRoute, bestDist
22.       end Algorithm
```

Algorithm 1: Layout of the GRASP Stage

In order to converge to a feasible solution, the variable α is introduced. α is equal to the function *f,* which depends on parameter β and the present iteration (equations 5 and 6). In addition, *f* converges to zero as the iteration number grows.

$$\alpha = f(\beta, \text{iter});\ \text{iter} \in [1, \text{iterMax}], \beta \in \Re^{[0,1]}. \quad (5)$$

$$f(\beta, \text{iter}) = \frac{\beta}{\text{iter}}, \beta \in \Re^{[0,1]} \quad (6)$$

The probability function to choose a node k_j is defined through α and the inverse of the distance (d_{kj}^{-1}). A distortion in the probability function is generated by ordering the nodes on the basis of the function. They are chosen according to the minor distance to the final bus stop. Let *n* be the amount of final nodes associated to the arcs *e* which depart from node *k*. The distortion considers *m* nodes, whose distance to the final bus stop is less than or equal to the previous one. The probabilities of these nodes are calculated by applying Equation 8. Then, the new probabilities for nodes with a greater distance are calculated with Equation 7.

$$P(k_j) = \frac{\frac{(1-\alpha)}{m} d_{k_j}^{-1}}{\frac{1}{m}\sum_{j=(n-m)}^{n} d_{k_j}^{-1} + \alpha \left[\left(\frac{1}{(n-m)} \sum_{i=1}^{n-m} d_{k_i}^{-1} \right) - \left(\frac{1}{m} \sum_{j=(n-m)}^{n} d_{k_j}^{-1} \right) \right]} \quad (7)$$

$$P(k_i) = \frac{\frac{\alpha}{(n-m)} d_{k_i}^{-1}}{\frac{1}{m}\sum_{j=(n-m)}^{n} d_{k_j}^{-1} + \alpha \left[\left(\frac{1}{(n-m)} \sum_{i=1}^{n-m} d_{k_i}^{-1} \right) - \left(\frac{1}{m} \sum_{j=(n-m)}^{n} d_{k_j}^{-1} \right) \right]} \quad (8)$$

$$i = 1,...,(n-m) \qquad j = (n-m),...,n \qquad k = 1,...,n$$

$$d_{k1} \geq ... \geq d_{ki} \geq ... \geq d_{k(n-m)} \qquad d_{k(n-m)} = ... = d_{kj} = ... = d_{kn}$$

3.1.2 Local Search Step

The local search stage is necessary in order to return sub-optimal solutions. Then, the neighborhoods of each node in the feasible solution are explored to find a better solution. The local search algorithm (see Algorithm 2) takes the best route generated in the randomized GRASP step and examines the neighborhood of each node.

Let i and j be the bus stops for which the route will be enhanced. We use an auxiliary variable k that goes from i to j, and for every node of this path we build a set of arcs, namely e, that emerge from node k. The **set e is what we call the neighbourhood of a node**. Then we search for all the nodes that connect k with j; formally $r_{k,j} = (k, \ldots, j)$. If there is a node l in a sub-route $r_{k,l} = (k, \ldots, h, \ldots, l)$ that belongs to e, and $h \neq \varnothing$, then we replace sub-route $r_{k,l} = (k, \ldots, h, \ldots, l)$ by (k, l) and the new path from k to j becomes $r_{k,j} = (k, l, \ldots, j)$. This procedure is repeated for all the subsequent nodes until $k = j$. Figure 2 shows the improvement achieved with the local search stage (Fig. 2.b) starting from the route between two bus nodes yielded by the randomized stage (Fig. 2.a).

Algorithm *LocalSearch*
Input: $r(i, j)$: Route;
OutPut: $r_{i,j} = (i,...,j)$: Route;
Var: k,l: node;
1. $r_{i,j} := r(i, j)$
2. $k := i$
3. **while** not ($k = j$) **do**
4. e:=outEdges(k)
5. **if** there is a bus stop l such that $(k,l) \in e$ and $r_{k,j} = (k,...,j)$ contains a sub-route $r_{k,l} = (k,...,h,...,l)$ **then**
6. replace $r_{k,l} = (k,...,h,...,l)$ with (k, l)
7. **end if**
8. $k := \text{next}(k, r_{k,j})$
9. **end while**
10. **return** $r_{i,j} = (i,...,j)$
11. **end Algorithm**

Algorithm 2: Layout of the Local Search Procedure.

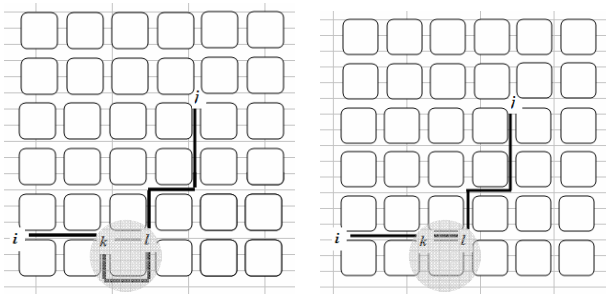

Figure 2(a): Randomized Step *Figure 2(b): Local Search Step*

3.2 Genetic-Algorithm Stage

Several authors have shown that GAs succeed in obtaining sub-optimal solutions for NP-Hard problems [Baaj, 91]. The multi-objective GA proposed in this work is based on the NSGA-II proposed by [Deb, 00]. It starts from a set of parameters, the distances between bus stops obtained through GRASP, and an initial population randomly generated. Each individual of the population represents a bus network. The GA evolves the population until several satisfactory bus networks are achieved in the last generation. The parameters are the following: number of bus stops with concentrated demand, information of each bus stop (position, time between arrivals), number of lines (M), fee for traveling with each line L (T_L), unitary cost of operation per kilometer for each line L (K_L), starting nodes B_S, return nodes B_F, and maximum capacity of buses C_{MAX}.

3.2.1 The individuals: Set of lines

An individual represents a bus network by means of a set of lines. Each route of a line is modeled by an ordered list of integers, starting from an initial bus stop B_S and ending with a final bus stop B_F. It is important to note that not all the integer strings constitute feasible solutions. Restrictions related to the feasibility of individuals are expressed according to the Set Theory. Consider that L_1, L_2,…, L_M are the lines represented by the individual *x*. A feasible solution for the BNSP should fulfill these constraints:

1. The solution should contain the same number of routes as the amount of lines (M) that were defined as input parameter of the algorithm (Equation 9).

$$x = \{L_1, \ldots, L_M\} \quad (9)$$

2. The number of transfer points should be consistent with the input parameter for the GA (Equation 10).

$$\forall L_i \exists L_j, i \neq j \Rightarrow L_i \cap L_j \neq \varnothing \quad (10)$$

3. All the bus stops should be present at least in one of the lines (Equation 11).

$$\{B_S\} \cup \{B_I\} \cup \{B_F\} = \{L_1\} \cup \{L_2\} \cup \ldots \cup \{L_M\} \quad (11)$$

3.2.1.1 Initial Population

The procedure we implemented in order to generate the initial population is performed as follows. The bus stops in each line are determined from those that constitute the sets of initial (B_s), intermediate (B_I) and final (B_F) bus stops. Each bus stop in each set has a probability of being chosen equal to 1 divided by the amount of elements in the corresponding set. Then, for every line in the individual, a random number between 1 and the amount of possible bus stops in the set is generated. This first number, namely #BusStops, corresponds to the quantity of bus stops that will to be selected for that line. Then, #BusStops random numbers are generated in order to select the corresponding bus stops for the given line. This task is repeated for each line. When this job is finished, equations 9, 10 and 11 are evaluated. If some of them are not satisfied, the individual is modified according to the some feasibility rules. For example, in the case where a line is not connected with any other line (Equation 10), an intermediate bus stop is randomly selected from another line, and is inserted in the disconnected one.

3.2.2 The fitness function: Optimization of FO_L and FU_L

Taking into account that FO_L and FU_L are two clear objectives of this problem, there is not only one solution for the problem. Moreover, both objectives (FO_L and FU_L) are equally important. For this reason, in this paper we are proposing a multi-objective GA with the *Pareto frontier* concept to tackle this MOP. In this context, the FO_L objective in Equation 1 is redefined, and Equation 12 shows the new formulation of this objective.

$$1 \Big/ \left(1 + \sum_{L=1}^{M} FO_L \right) \tag{12}$$

Then, the multi-objective optimization for the BNSP can be formalized as follows:

Find a design-variable set:

$$\boldsymbol{x} = (x_1, x_2) \tag{13}$$

that minimizes the following objective function:

$$f_i(\boldsymbol{x}) = f_i(\boldsymbol{x}^*) \forall i \in \{1,2\} \tag{14}$$

$$f_i(\boldsymbol{x}) = f_i(\boldsymbol{x}^*) \text{ where}$$

$$f_i \in \left\{ 1 \Big/ \left(1 + \sum_{L=1}^{M} FO_L \right), \sum_{L=1}^{M} FU_L \right\} \tag{15}$$

In that case, the solution set for the BNSP consists of all decision vectors, whose corresponding objective vectors cannot be improved in any dimension without degradation in another one. These vectors are known as *Pareto optimal*. For the BNSP, we consider two decision vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Then, for minimization problems, $\boldsymbol{x}_2$ is said to dominate $\boldsymbol{x}_1$ if:

$$\forall i \in \{1,2\}\ f_i(\boldsymbol{x}_2) \leq f_i(\boldsymbol{x}_1)\ \textit{and}$$
$$\exists i \in \{1,2\}\ f_i(\boldsymbol{x}_2) < f_i(\boldsymbol{x}_1) \quad (\mathbf{16})$$

Additionally, all the decision vectors that are not dominated by any other decision vector are called *non-dominated,* vis-à-vis this set. Non-dominated decision vectors constitute the so called *Pareto-optimal* front. As to the fitness assignment, the Hybrid Algorithm uses the *Pareto-based* approach proposed by Fonseca and Fleming [Fonseca, 93].

3.2.2.1 The Simulation

The manner in which the simulation is faced depends to a large extent on the optimization model built for the problem at hand. For the formal BNSP treatment, it was necessary to apply simulation techniques related to queue theory and access to resources. The static structure of a bus network is basically composed of different routes, the operator's fleets, the network users, and the transfer points. A transfer point is in more than one route. This case may be associated to a user-change-of-unit centre that connects the transport network and enables the user to access every point. It should be taken into account that a bus moves from the initial line stop to the final one, and then it travels back directed towards the initial point.

During the simulation, each entity is associated to some information that should be obtained, so that this information is later used to calculate the fitness function of the GA. For each user, we are interested in its own waiting time (t^W_{ijL}), trip time (t^J_{ijL}), access time (t^A_{ijL}), destination node, and - in the case he is traveling - the mobile the user is in. Each node keeps two lists: a list of clients who are queuing, i.e. waiting to get into a mobile (there is a list for each line crossing that node), and a list of the users or elements that will arrive in that stop. It also stores information on the number of users that arrived to this centre (V_{ijL}). The line's attributes include a list of the nodes the line goes through, its fleet and the total influx of trips (AF_L). Finally, each mobile has information about the point it is on, the next node, the current capacity (CA), the maximum capacity (CM), and the present clock (t_{Now}).

Event-Planner Implementation: Building the computational model for a simulation is a challenging enterprise, due to the complexity of posing the problem. The simulation time progresses in a discrete and synchronic way. For our problem instance, the planned event that advances the simulation clock is the arrival of a mobile to a node. Before beginning the simulation effectively for a mobile's working-day, the arrival of all the clients to the respective transfer centers is generated, and they are put on the customers' list that will potentially arrive to the stop.

The simulation begins by generating each mobile's arrival to its route's initial point. The simulation clock advances to the first arrival of the first transport of the fleet. Two lists are updated: the list of clients that arrived at the node, and the list of clients that will arrive later. Besides, the transport arrival to the next node is planned taking into account the present clock and the distance that has been traveled, which was obtained through GRASP. The following attributes are updated: those that belong to the lines (A_{FL}, V_{ijL}), to the user (t^W_{ijL}, t^J_{ijL}, t^A_{ijL}), and to the transport (CA, t_{Now}). In this way, the clock moves forward until the simulation is over.

3.2.3 Genetic operators

3.2.3.1 Crossover

A variation of the two-point crossover operator (OX) was adopted for this problem. It is interesting to point out that the one-point approach yielded unsatisfactory results since the children's fitness value became lower than their parents. The crossover method works in this way: The algorithm selects two edges for each route of each parent (two-point crossover). For the first child (see Figure 4.a), the algorithm picks out the first parent's first sub-routes (Line 1: 1-5-14-; Line 2: 2-6-13-16-23; Line 3: 2-7-12-11; Line 4: 3-8-11-18-22; Line 5: 4-9-10, see Figure 3.a), and they are connected with the second parent's middle sub-routes (Line 1: 15-24; Line 2: 16-25-33; Line 3: 12-17-23; Line 4: 22-28; Line 5: 20-19-21-29, see Figure 3.b). For the last connection, it is done with the last sub-routes that belong to the first parent (Line 1: 25-26-34-35-36; Line 2: 37; Line 3: 27-32-38; Line 4: 30-40; Line 5: 31-39, see Figure 3.a).

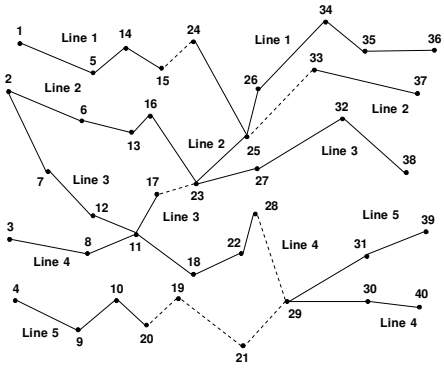

Figure 3(a): Scheme for parent 1

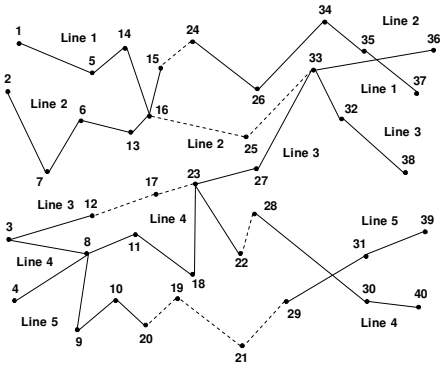

Figure 3(b): Scheme for parent 2

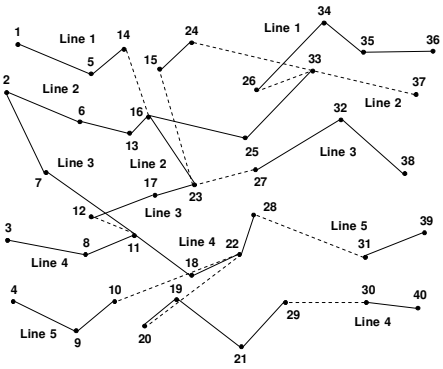

Figure 4(a): Scheme for child 1

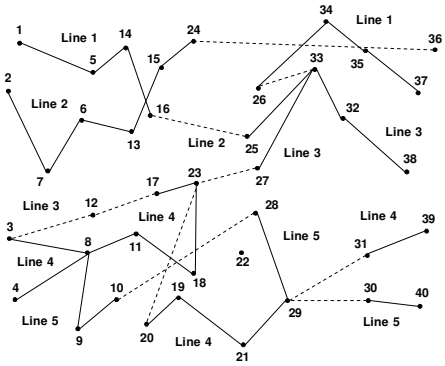

Figure 4(b): Scheme for child 2

In the same way, for the second child (see Figure 4.b) the first sub-routes belonging to the second parent are taken (Line 1: 1-5-14-16; Line 2: 2-7-6-13: Line 3: 3; Line 4: 3-8-11-18-23; Line 5: 4-8-9-10, see Figure 3.b). Then, they are connected with the first parent's middle sub-routes (Line 1: 15-24; Line 2: 25-33; Line 3: 17-23; Line 4: 28-29; Line 5: 20-19-21-29, see Figure 3.a). Finally, they are connected with the second parent's last sub-routes (Line 1: 26-34-35-37; Line 2: 36; Line 3: 27-33-32-38; Line 4: 30-40; Line 5: 31-39, see Figure 3.b).

By using the constraints defined in equations 9, 10 and 11, the algorithm detects unfeasible individuals in order to keep feasible individuals in the population. When the crossover produces unfeasible children, they are discarded and new cut-points for their parents are selected.

3.2.3.2 Mutation

As regards mutation, the children have two options: either edge or node mutation. Figure 5 shows a child before applying the mutation operator (Line 1: 1-5-14-16-15-24-26-34-35-37; Line 2: 2-7-6-13-16-25-33-36; Line 3: 3-7-12-17-23-27-32-38; Line 4: 3-8-11-18-22-28-30-40; Line 5: 4-8-9-10-20-19-21-29-31-39).

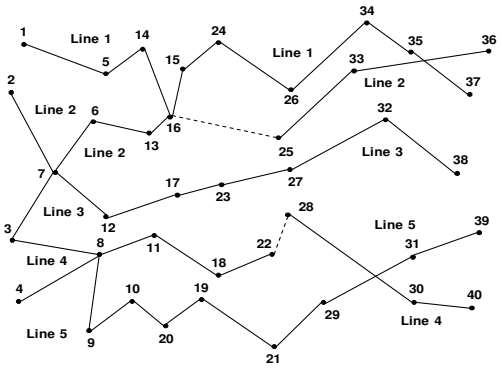

Figure 5: Child about to be mutated.

Edge mutation randomly chooses two edges from two different routes and inverts the connection (see Figure 6.a; Line 1: 1-5-14-16-15-24-26-34-35-37; Line 2: 2-7-6-13-16-28-30-40; Line 3: 3-7-12-17-23-27-32-38; Line 4: 3-8-11-18-22-25-33-36; Line 5: 4-8-9-10-20-19-21-29-31-39).

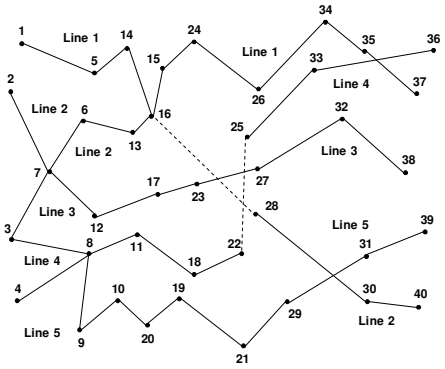

Figure 6(a). Child after Edge Mutation

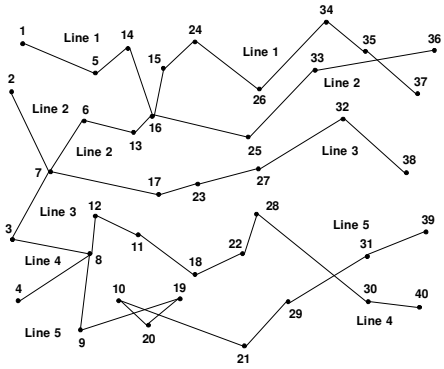

Figure 6(b). Child after Node Mutation

There are two alternatives for node mutation: to insert the randomly selected node into another route or to introduce it into the same route (see Figure 6.b; Line 1: 1-5-14-16-15-24-26-34-35-37; Line 2: 2-7-6-13-16-25-33-36; Line 3: 3-7-17-23-27-32-38; Line 4: 3-8-12-11-18-22-28-30-40; Line 5: 4-8-9-19-20-10-21-29-31-39). In both cases, if the resulting individual is unfeasible, it is discarded. Then, the original individual is mutated again until a feasible individual has been obtained.

3.2.4 Putting it all together: GRASP + NSGA-II * SIMULATION

In this section we will explain how all the aforementioned pieces are assembled. Firstly, the GRASP calculates the different paths and distances between every pair of bus stops. Then, based on the "map" yielded by the GRASP stage, a random population of feasible bus networks is built. Later, in order to evaluate the fitness of the individuals in the population, it is necessary to calculate the value of each one of the objectives, FO_L and FU_L to be short. For this aim, the algorithm simulates a day of work of the lines represented by each one of the individuals, and returns the values for the variables that are necessary for the calculation of the equations shown in section 3.2.2. Afterward, a binary tournament selection is performed, and a variant of the two-point crossover and a mutation operator adapted for this problem instance are used to create the population Q_0 of size N. In Algorithm 3 we show a simplified pseudo-code of the whole procedure, with a special focus on the NSGA-II.

The solutions are ordered in R_t according to the frontier concept, and also with respect to the relative distance with two other near solutions, the so called *crowding distance* [Deb, 00]. The population is arranged in descending form of magnitude bearing in mind these values. Thus, the solutions that are more separated are selected and the population P_{i+1} is created. Then, a Crowded Tournament Selection is realized who compares two solutions and returns the winners of the tournament. This is realized in every generation until a given generation number is reached.

Finally, a short complexity analysis of the algorithm will be presented. The initialization algorithm is executed only once at the beginning of the procedure, with an execution time in the $O(N^3)$ [Feo, 95], [Resende, 99]. It has been demonstrated that NSGA-II needs $O(gN^2)$ to order a population of size $2N$ with the domination criterion.

The calculation of the *crowding distance* [Deb, 02] needs no less than $O(gN*_{log}N)$ for what NSGA-II takes $O(gN^2)$. The simulation takes $O(NM)$ for a population of size N with each individual representing M lines. In g generations, the entire time of the genetic algorithm is $O(gN^2)$.

4 Practical Experience

In this section, an academic case study is presented, and the results achieved by the Hybrid Algorithm are reported and analyzed. The experiments were carried out on the map of a hypothetical city that is represented with 100 nodes, 2 initial stops (B_S), 6 intermediate stops (B_I), and 2 final stops (B_F). Figure 7 shows the node layout and the position of the stops in the map, and the colours help to distinguish each bus-stop category. As it can be observed in the figure, B_s = {61, 21}, B_F = {70, 40} and B_I = {16, 42, 48, 65, 83, 89}.

Algorithm: ***Time-Dependent Hybrid Algorithm***

Input: M: Map; B_S, B_F, B_I: set of busStops;
[AF_L, T_L, D_L, K_L]: set of real; [VST_{ijL}]: set of real;
δ,η : real; g: integer; (g is the number of generations)
N: integer; (N is the number of individuals in the Population)
OutPut: P_g
Var distanceBusStops: list of Distance; routesBusStops:list of Route;
0. Initialization of busStops' distances and routes between all bus stops (***GRASP***: B_S, B_F, B_I)
1. Generate random population P_0
2. **for each** $x \in P_i$ **do**
3. for each *line* L in *x* Simulation return [AF_L, T_L, D_L, K_L, t^A_{ijL}, t^J_{ijL}, t^W_{ijL}]
4. **end for**
5. Evaluate Objective FO and FU for each individual $x \in P_0$
6. Assign Rank (level) Based on Pareto dominance
7. Parents := Crowded Tournament Selection
8. Q_0 := Crossover(Parents)
9. Mutation(Q_0)
10. **for** i := 0 **to** g-1 **do**
11. **for each** $x \in Q_i$ **do**
12. for each *line* L in *x* Simulation return [AF_L, T_L, D_L, K_L, t^A_{ijL}, t^J_{ijL}, t^W_{ijL}]
13. **end for**
14. Evaluate Objective FO and FU for each individual $x \in Q_i$
15. $R_i := P_i \cup Q_i$
16. **for each** Parent and Child in Population R_i **do**
17. Assign Rank (level) based in Pareto
18. Generate sets of non-dominated vectors
19. Loop (inside) by adding solutions to next generation starting from the first front until N individuals found determine crowding distance between points of each front
20. **end for**
21. Select points (elitist) on the lower front (with lower rank) and are outside a crowding distance (crowded tournament selection)
22. Create Next generation P_{i+1}
23. Parents := Crowded Tournament Selection
24. Q_i := Crossover(Parents)
25. Mutation (Q_i)
26. **end for**
27. **end Algorithm**

Algorithm 3: Hybrid Algorithm Layout.

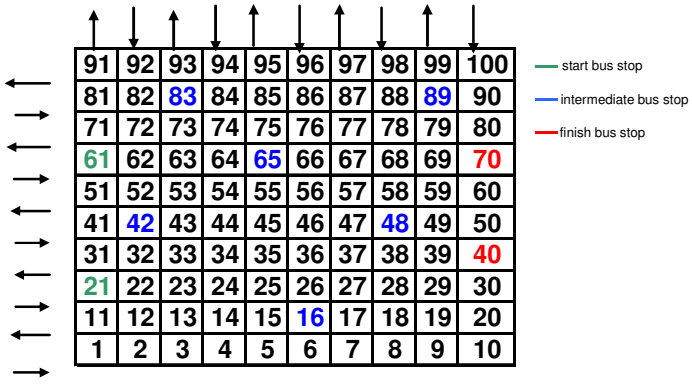

Figure 7: Nodes and bus stops of the fictitious city

The goal is to design the routes for two bus lines L_0 and L_1. There is room for 25 people in each bus, which is the maximum amount to be admitted. For this experiment, we modeled only one time frequency (the morning) that is set at the beginning of the run. The client arrivals and the corresponding destinies are arranged on a table that constitutes the entry to the simulation phase. The bus arrivals at each line's initial stop are also stored on a table that is used during the simulation. Figure 10 represents the nodes with numbers, the roads' directions with arrows, and the bus stops with numbers in a different colour.

The experimentation was conducted in two phases. The first phase was proposed to provide some expectation of convergence of the algorithm. All of the parameters were set fixed except for the number of generations, which varied among 50, 100, 150 and 200. The parameters that remained fixed were the following. For the GRASP: *iterMax* = 1 and β = 0.4; for the GA: *popSize* = 100, *CrossoverProb* = 0.9, *MutationProb* = 0.2; and for the simulation step the *BusFrequency* was of 10 minutes. For the second phase, the amount of generations was established from the analysis of the results of phase one. In this new stage, the aim was to test the Hybrid Algorithm in different scenarios, varying the bus frequency between 10, 20 and 40 minutes as shown in Table 1.

	Simulation Parameters
Scenarios	*Bus Frequency*
1	10
2	20
3	40

Table 1: Bus Frequency in minutes for each scenery

At this point, it is important to remark that for this problem instance, *iterMax* =1 because one single repetition was enough to achieve good results in short computing times. However, for real maps, several iterations might be needed and the algorithm presented in Section 3 should be implemented.

4.1 Experimentation: Phase 1

For all the cases in the first phase, the performance of the Hybrid Algorithm was tested by means of ten independent runs for each amount of generations. At the end of each run, the final population was saved. Then, all these populations were gathered and analyzed in order to build a front of non-dominated individuals. The fronts obtained from each of these four pools of individuals are shown in Figure 8, as regards the values obtained for equations 1 and 4.

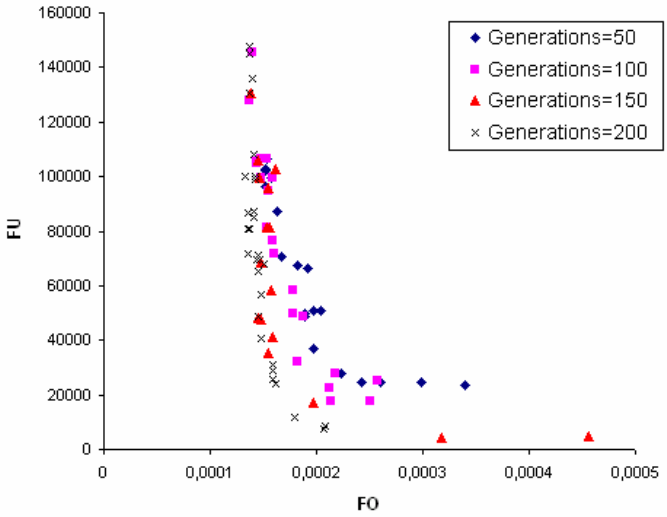

Figure 8: Non-dominated individuals obtained for different numbers of generations

As it can be seen in the graphic, the individuals for 200 generations are better than those of 50 generations, as it was to be expected. However, from 150 to 200 generations, the improvement is not so significant. Moreover, for the FO objective, some perspective of convergence can be perceived. Therefore, more than 200 generations do not seem to be necessary.

4.2 Experimentation: Phase 2

During the second phase, twenty runs with 200 generations were performed for each scenario. In order to give some illustrative examples before the analysis, four solutions were selected from the pool of non-dominated individuals generated during all the executions of the genetic algorithm, for each scenario. In terms of the policy on which solutions were best and why, we used the criteria of the entities. Two solutions were picked according to the client's wills, and the other two were selected regarding the operator's interests. At this point, it is important to note that an average of 20 non-dominated solutions was generated in each execution of the genetic algorithm. Table 2 illustrates the details of lines L_0 and L_1 represented by each one of these four non-dominated solutions found by independent runs of the Hybrid Algorithm for the three scenarios with runs of 200 generations. The numbers listed for each line correspond to the nodes illustrated in Figure 7.

Finally, Table 3 reports the statistics calculated from the solutions that were near the Pareto-Front. This table is used to show the contrast of the results in each scenario. The comparison is based on the variables that characterize the quality of the transit-route network configurations [Desaulniers, 07], such as percentages of trips with zero, one, or more than one transfers, total travel time in a non-dominated solution, total route length, etc. These variables are the same as those chosen for their examples by the referents in the field [Mandl, 79], [Shih, 94]. For each user that gets on the bus at stop *i* and wants to get off at stop *j*, the number of transfers he has to make to go from *i* to *j* must be reported. For this reason, all the simulated users were classified in different rows of the table. If a user needs no transfer between lines, the journey is called a 0-transfer trip. Similarly, when a user needs *q* transfers between lines, the journey is called a *q*-transfer trip. Then, the transfer percentage is calculated from the total amount of journeys that took place (see the first 3 rows in Table 3). The value associated to the unsatisfied demand percentage (row 4 in Table 3) corresponds to the amount of users that could not reach their destination as the simulation concluded. The route's length (row 5 in Table 3) is the addition of the distances between every pair of bus stops. The waiting time for a user (row 6 in Table 3) is the time that elapses since he arrives to the bus stop and the bus arrives. The operating cost (row 7 in Table 3) is an average of the FO objective for the non-dominated solutions.

		Scenarios		
		1	2	3
Route Network Layouts	*1*	Line 0: 21-42-83-48-40	Line 0: 21-83-16-42-65-48-40	Line 0: 21-83-16-48-65-42-89-40
		Line 1: 61-65-16-89-48-70	Line 1: 61-89-16-70	Line 1: 61-65-70
	2	Line 0: 21-83-65-42-16-48-89-40	Line 0: 21-42-48-16-40	Line 0: 21-83-42-16-48-65-89-40
		Line 1: 61-48-70	Line 1: 61-65-83-89-48-70	Line 1: 61-16-70
	3	Line 0: 21-65-83-42-16-48-40	Line 0: 21-65-83-42-16-48-40	Line 0: 21-83-16-48-42-65-89-40
		Line 1: 61-89-16-70	Line 1: 61-89-16-70	Line 1: 61-83-70
	4	Line 0: 21-83-42-83-48-40	Line 0: 21-16-48-40	Line 0: 21-83-65-42-16-48-89-40
		Line 1: 61-65-16-89-48-70	Line 1: 61-83-65-42-89-48-70	Line 1: 61-83-70

Table 2: Some examples of non-dominated solutions for the scenarios on Table 1

		Scenarios		
		1	**2**	**3**
Statistical Results	*% demand 0-transfer*	68,68	70,505	46,8
	% demand 1-transfer	31,3	29,245	13,2
	% demand +1-transfer	0	0	0
	% insatisfied demand	0	0	40
	Average Total Route Length	Line 0: 5533	Line 0: 3200	Line 0: 8900
		Line 1: 3566	Line 1: 3800	Line 1: 2500
	Average Wait Time for a user	2 min 30 sec	12 min 28 sec	25 min 54 sec
	Average Operation Cost	5932,03 m.u	3983,24 m.u	3936,67 m.u

Table 3: Statistical Results

As it can be observed, the best results were obtained for the scenario 2. It is interesting to note that scenario 3 was directly discarded from the analysis since the results show that the 40% of the users could not reach their destination. Therefore, one of the main objectives of the BNSP was not fulfilled in that scenario. On the other

hand, in scenario 2 the lines are balanced as regards their length. Also, the users needed fewer amounts of transfers between lines. Moreover, as it can be seen in row 1, there were more users that did not need any transfer at all. However, as it can be expected, since scenario 2 represents the situation with a frequency of 20 minutes, the wait time for the users grows from the one obtained for scenario 1. Nonetheless, the best trade-off between the user's and operator's interests was found in scenario 2.

5 Conclusions and Future Work

In this work we have presented the main features of a hybrid stochastic technique that efficiently combines several meta-heuristics to solve BNSP in a cooperative way. The proposed technique gathers a map initialization tool implemented by a GRASP with a route and scheduling technique based on a multi-objective GA. A simulation tool was included in the GA for the calculi of time-dependent variables associated to the fitness of the individuals.

The method was tested on the map of a hypothetical city, which was represented with 100 nodes. Three scenarios were generated for the tests, with different bus frequencies. The results showed that a 20-minute time frequency yields the best trade-off between the objectives of the user and the operator of the lines. These promising results encourage the testing of this novel technique on bigger maps. The main objective is to be able, in the future, to use the Hybrid Time-Dependent Algorithm as a decision support tool for the bus network scheduling of real congested urban areas.

Acknowledgements

We would like to thank for their economical support to: the *Consejo Nacional de Investigaciones Científicas y Técnicas* (CONICET), to the ANPCyT for BID OC/AR 1728 N°1652; and to UNS for Grant PGI 24/N019.

References

[Aarts, 97] Aarts E., Lenstra, J. K.: Local Search in Combinatorial Optimization, Wiley, New York, 1997.

[Baaj, 91] Baaj, M. H., Mahmassani H. S.: An AI-Based Approach for Transit Route System Planning and Design, Journal of Advanced Transportation, Vol. 25, No. 2, 1991, 187-210.

[Bertsekas, 98] Bertsekas, D. P., Castañon, D. A.: Solving Stochastic Scheduling Problems Using Rollout Algorithms, Lab. for Information and Decision Systems Report P-12413, M.I.T., Cambridge, MA., 1998.

Bielli, 02] Bielli, M., Caramia, M., Carotenuto P.: Genetic Algorithms in Bus Network Optimization, Transportation Research Part C: Emerging Technologies, Elsevier, Vol. 10, Issue 1, 2002, 19-34.

[Bookbinder, 92] Bookbinder, J.H., Désilets A.: Transfer Optimization in a Transit Network, Transportation Science, Vol. 26, No. 2, 1992, 106-118.

[Byrne, 72] Byrne, B.F., Vuchic, V.R.; Public Transportation Line Position and Headway for Minimum Cost. Traffic Flow and Transportation, G.F. Newell, ed., American Elsevier, New York, 1972, 347-360.

[Ceder, 86] Ceder, A., Wilson, N. H. M.: Bus Network Design, Transportation Research, Vol. 20, No. 4, 1986, 331-344.

[Ceder, 97] Ceder A., Israeli Y.: Creation of Objective functions for transit network design. Proceedings of IFAC Transportation Systems Chania, Greece, 1997, 684–689.

[Ceder, 98] Ceder, A., Israeli, Y.: User and Operator Perspectives in Transit Network Design, Transportation Research Record, Vol. 1623, 1998, 3-7.

[Chang, 89] Chang, S.K., Schonfeld, P.M.: Analytical Optimization of Parallel Bus Routes with Time Dependent and Elastic Demand. Transportation Studies Center Working Paper 89-22, University of Maryland, College Park, 1989.

[Cheng-Fa, 04] Cheng-Fa, T., Chun-Wei, T., Ching-Chang, T.: A New Hybrid Heuristic Approach for Solving Large Travelling Salesman Problem, Information Sciences, Elsevier. Vol. 166, 2004, 67–81.

[Deb, 00] Deb, K., Agarwal, S., Pratap, A., Meyarivan T.: A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, Proceedings of the Parallel Problem Solving from Nature VI Conference, pages 849–858, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

[Deb, 02] Deb, K., Pratap, A., Agarwal, S., Meyarivan T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 2, 2002, 182-197.

[Desaulniers, 07] Desaulniers, G. and Hickman, M.: Public transit. Handbooks in Operations Research and Management Science, Transportation, Vol. 14, G. Laporte and C. Barnhart (eds), Elsevier, Amsterdam, 2007, 69-127.

[Feo, 95] Feo T. and Resende M.: Greedy randomized adaptive search procedures, Journal of Global Optimization, Vol. 6, 1995, 109-133.

[Fonseca, 93] Fonseca, C. M., Fleming, P. J.: Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization, Forrest, 1993, 416-423.

[Fonseca, 95] Fonseca, C. M., Fleming, P. J.: Multiobjective Genetic Algorithms Made Easy: Selection, Sharing and Mating Restriction, GALESIA, 1995, 45-52.

[Frutos, 07] Frutos, M., Casal, R., Olivera, A. C.: Algoritmo Híbrido Estocástico Aplicado al Diseño de Rutas y Determinación de Frecuencias en el Transporte Público Urbano, V Congreso Español sobre Meta-Heurísticas, Algoritmos Evolutivos y Bioinspirados, España, 2007.

[Goldberg, 89] Goldberg, D. E.: Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Massachusetts, 1989.

[Gruttner, 02] Gruttner, E., Pinninghoff, M. A., Tudela, A., Díaz, H.: Recorridos Óptimos de Líneas de Transporte Público Usando Algoritmos Genéticos, Jornadas Chilenas de Computación, Chile, 2002.

[Lampkin, 67] Lampkin, W. and Saalmans, P.D., The Design of Routes, Service Frequencies and Schedules for a municipal Bus Undertaking: Case Study, Operation Research Quarterly, Vol. 18, No. 4, 1967, 375-397.

[Liu and Abraham, 07] Liu, H. and Abraham, A.: An Hybrid Fuzzy Variable Neighborhood Particle Swarm Optimization Algorithm for Solving Quadratic Assignment Problems, Journal of Universal Computer Science, Vol. 13, No. 7, 2007, 1032-1054.

[Mandl, 79] Mandl, C.: Applied Network Optimization, Academic Press, New York, 1979.

[Mandl, 80] Mandl, C. E.: Evaluation and optimization of urban public transportation networks, European Journal of Operational Research, Elsevier, Vol. 5, No. 6, 1980, 396-404.

[Olivera, 07] Olivera, A. C., Frutos, M., Carballido, J. A., Brignole, N. B.: Bus Network Optimization Through Time-Dependent Hybrid Algorithm, International Conference on Intelligent Systems Design and Applications (ISDA'07), IEEE Computer Society, Los Alamitos, CA, USA, Vol. 0, 2007, 857-862.

[Pattnaik, 1998] Pattnaik, S. B., Mohan, S., Tom, V. M.: Urban Bus Transit Route Network Design Using Genetic Algorithm. Journal of Transportation Engineering, Vol. 124, No. 4, 1998, 368-375.

[Pearl, 84] Pearl, J.: Heuristics: Intelligent Search Strategies for Computer Problem Solving. Addison-Wesley, 1984.

[Pitsoulis, 01] Pitsoulis, L. S., Resende, M.G.C.: Greedy Randomized Adaptive Search Procedures. In P.M. Pardalos and M.G.C. Resende editors, Handbook of Applied Optimization. Oxford University Press, 2001.

[Rapp, 76] Rapp, M. H., Gehner, C. D.: Transfer optimization in an interactive graphic system for transit planning. Transportation Research Record, Vol. 619, 27-33. 1976.

[Resende, 99] Resende, M.: Greedy Randomized Adaptive Search Procedures (GRASP), To appear in Encyclopedia of Optimization, Kluwer Academic Press, 1999.

[Shih, 94] Shih, M., Mahmassani, H. S.: Design Methodology for Bus Transit Networks with Coordinated Operations, University of Texas at Austin, 1994, 197 pages.

[Wolsey, 98] Wolsey, L.A.: Integer Programming. John Wiley & Sons, New York, 1998.

[Yongshuang, 06] Yongshuang, Z., Baoding, L.: Fuzzy Vehicle Routing Model with Credibility Measure and its Hybrid Intelligent Algorithm, Applied Mathematics and Computation. Vol. 176, Issue 2, 2006, 673-683.

[Zhao, 05] Zhao, F., Ubaka, I., Gan, A.: Transit Network Optimization: Minimizing Transfers and Maximizing Service Coverage with an Integrated Simulated Annealing and Tabu Search Method, Transportation Research Record, Vol. 1923, Issue 1, 2005, 180-188.

[Zhao, 06] Zhao, F.: Simulated Annealing–Genetic Algorithm for Transit Network Optimization, ASCE Journal of Computing in Civil Engineering, Vol. 20, No.1, 2006, 57-68.