# Tabu Search for a Multi-Objective Routing Problem

JOAQUÍN PACHECO

Fac. C Económicas y Empresariales. Universidad de Burgos.
Plaza Infanta Elena s/n. Burgos 09001, Spain.

RAFAEL MARTÍ[*]

Departament d'Estadística i Investigació Operativa, Universitat de València.
Burjassot 46100, Spain. rafael.marti@uv.es

## Abstract

Multi-objective optimization problems deal with the presence of different conflicting objectives. Given that it is not possible to obtain a single solution by optimizing all the objectives simultaneously, a common way to face these problems is to obtain a set of efficient solutions called the non-dominated frontier. In this paper we address the problem of routing school buses with two objectives: minimize the number of buses, and minimize the longest time a student would have to stay in the bus. The trade-off in this problem is between service level, which is represented by the maximum route length, and operational cost, which is represented by the number of buses in the solution. We present different constructive solution methods and a tabu search procedure to obtain non-dominated solutions. The procedure is coupled with an intensification phase based on the path relinking methodology; a strategy proposed several years ago, which has been rarely used in actual implementations. Computational experiments with real data in the context of routing school buses in a rural area, establish the effectiveness of our procedure in relation to the approach previously identified to be best.

**Key Words:** Meta-Heuristics, Bus Routing, optimization.

[*] Corresponding author. Research partially supported by the Ministerio de Ciencia y Tecnología of Spain: TIC2000-1750-C06-01.

## 1. Introduction

The multi-objective optimization problem consists in finding the values for a set of decision variables that satisfy a set of constraints and that optimize a vector function whose elements represent the objectives. These functions are a mathematical representation of performance criteria that are usually in conflict. Hence, an optimum solution is that which leads to objective function values that are acceptable to a decision maker.

In this paper we address a common variant of the bus routing problem that appears in sparse rural areas of Spain. It consists of transporting a group of students from their homes to a school. The students live in locations that are geographically dispersed around the school and the set of available buses have the same capacity. We consider the objectives of minimizing the total number of buses (operational cost) while simultaneously minimizing the maximum time that a student spends in the bus (service level). This specific problem was introduced in Corberan et al. (2002) where a scatter search algorithm was proposed. The authors mentioned several characteristics of this problem that can be summarized as:

- in a rural area, routes tend to be long and the buses do not reach their maximum physical capacity.
- due to the definition of transportation time we can ignore the time required to reach the first student.
- it is a single load problem (routes either pick up or deliver students to a single school).

Additional characteristics for a variety of routing and scheduling problems, including those related to school buses, are described in other previous works (Bodin and Berman, 1979, or Bodin, et al., 1983). We view our work as an extension of the paper by Corberan et al. (2002) since we consider the same problem and propose a new solution procedure, based on the tabu search methodology, that outperforms their scatter search implementation. We will follow their notation to give a formal description of the problem objectives:

$N = \{ 0, 1, \dots, n \}$ : a set of locations where 0 indicates the school and $j$ (for $j = 1, \dots, n$) is the index of a location where one or more students live.

$M = \{ 1, \dots, m \}$ : a set of buses

$R_i = \{ r_i(1), \dots, r_i(n_i) \}$ : the route for bus $i$, where $r_i(j)$ is the index of the $j$th location visited and $n_i$ is the number of locations in the route. We assume that every route finishes at the school, i.e., $r_i(n_i+1) = 0$.

$t_{jk}$ : the direct traveling time from location $j$ to location $k$, for $j = 0, \dots, n$ and $k = 0, \dots, n$ and $t_{jk} = 0$ for $j = k$.

$c$ : the capacity of a bus

$q_j$ : the number of students to be picked up at location $j$, for $j = 1, \dots, n$

Note that according to these definitions, the length of route $i$, $length(i)$, given by the maximum traveling time corresponding to the students picked up at the first location, is computed as:

$$length(i) = \sum_{j=1}^{n_i} t_{r_i(j), r_i(j+1)} \ ,$$

and the maximum time in the bus of all the routes (of an entire solution) is given by:

$$t_{\max} = \max_{i \in M} \{ length(i) \}$$

The bus routing problem is to find a set of routes in order to minimize the number of buses ($m$) and to minimize also the maximum time in the bus ($t_{max}$). Then, the objectives of the problem can be formally stated as (1) Minimize $m$ , and (2) Minimize $t_{max}$. Both objectives clearly are in conflict (i.e., a solution that minimizes the number of buses tends to increase the maximum traveling time, and vice versa). As it is usual in routing problems, routes must satisfy the capacity constraints; i.e. the total number of students picked up in a route cannot exceed the capacity of the bus. Moreover, routes in this problem must satisfy a "social" requirement that specifies that their length cannot exceed a maximum pre-established value (*TMAX*).

In this paper we follow the common approach when dealing with multi-objective problems; to solve the problem considering both objective functions separately and giving the decision maker (administrators and school officers in this case), a set of solutions that represent the "the best possible service" at a given cost. Since the value of the first objective, number of routes (buses), is a discrete number (bounded by the number of locations), we follow the simple an efficient method that consists of minimize the second objective, the maximum length of a route, for each possible value of the first one.

In this paper we present different constructive solution procedures and a tabu search method. Section 2 revises two previous constructive procedures and introduces two new heuristics to obtain good initial solutions for the search method. Section 3 describes our tabu search implementation. The intensification component of the tabu search method is strengthening with a path relinking strategy. Section 4 is devoted to this strategy, which operates on a set of elite solutions recorded during the application of the basic tabu search method. Section 5 summarizes the computational experiments, and the paper finishes with the associated conclusions.

## 2. Constructive Methods

Corberán et al. (2002) propose two constructive heuristics for this problem (H1 and H2), the former is based on a clustering mechanism, while the latter performs a sweep to determine the routes. In this section we briefly describe both approaches and introduce two new methods (H3 and H4). H3 is based on the well known heuristic for the VRP due to Fisher and Jaikumar (1981), and H4 implements an insertion mechanism. H3 and H4 construct a solution for a given value of the number of buses (routes) *m*. The number of buses in a solution of H1 and H2 is an output of the method and cannot be pre-specified (although in the case of H2 could be indirectly done). The objective of the four methods is to create a solution with a relative low value of the maximum length of the routes.

### *Constructive Heuristic H1*

The procedure starts with *n* routes (one for each location, $R_i = \{ i \}$ for $i = 1, \ldots, n$) and, at each iteration, two routes are merged. Instead of considering all pairs of routes, an ordered candidate list with the best pairs is constructed. The best pair of routes is the one with the minimum traveling time between the two routes, and the candidate list contains the pairs whose traveling time is within a percentage of the best pair. At each iteration, the algorithm randomly selects one pair from the candidate list and attempts to merge the routes. The procedure attempts to merge up to a prefixed number of pairs using the same candidate list. After these merging attempts, the candidate list is rebuilt. The process stops when no more routes can feasibly merge.

The method only considers two ways to combine routes $\mathrm{R}_i$ and $\mathrm{R}_k$:

$$\mathrm{R}' = \{ r_i(1), \ldots, r_i(n_i), r_k(1), \ldots, r_k(n_k) \} \text{ and } \mathrm{R}'' = \{ r_k(1), \ldots, r_k(n_k), r_i(1), \ldots, r_i(n_i) \}$$

Note that there are two feasibility conditions for each route; route length ≤ TMAX, and number of students in the bus ≤ c. Then, a feasible merging of routes $\mathrm{R}_i$ and $\mathrm{R}_k$ is such that the resulting route satisfies both inequalities.

### *Constructive Heuristic H2*

This constructive heuristic is based on creating sectors around locations that are sequentially chosen. The size of a sector is determined by an input parameter. When a location is selected and it does not belong to an already defined sector, a new sector is defined around the chosen location. This is the case, for instance, when the procedure starts and the first location is chosen.

The locations are ordered according to the decreasing value of $t_{j0}$ (the traveling time from location *j* to the school) for $j = 1, \ldots, n$. Let *tm* be the traveling time of the first location in the list (which is the unassigned location that is farthest away from the school). The next location to be assigned is randomly chosen from all those unassigned locations with traveling time to school larger than or equal to $tm*\alpha$, where $\alpha$ is a parameter for this heuristic. A chosen location is either assigned to an existing sector if the assignment does not violate the feasibility constraints or a new sector is defined. In the former case, the chosen location is inserted in the position of the route that causes the least increase in length. When a new route is created, we allow for either the first location or the last location of existing routes in the same sector to move to the new route if this move decreases $t_{\max}$.

### *Constructive Heuristic H3*

We adapt the classical method for the VRP by Fisher and Jaikumar (1981) to this problem. The method reformulates the routing in terms of a Generalized Assignment Problem (GAP) to determine a feasible assignment of locations to vehicles. Then, a traveling salesman problem (TSP) is solved to establish a pick up sequence for the locations assigned to each vehicle.

The assignment step starts by determining $m$ seeds points $w_1, w_2, ..., w_m$ in the plane. To this end, the plane is partitioned into $m$ cones corresponding to the $m$ buses or routes. Then $w_i$ is located on the ray bisecting cone $i$. The cost $c_{ij}$ of assign location $i$ to cluster (bus) $j$, for the symmetric case, is given by the following expression (we have replaced the original Euclidean distances with the traveling times):

$$c_{ij} = t_{0i}+t_{iwj}-t_{0wj}$$

The procedure then solves a GAP with these costs, location weights $q_i$, and bus capacity $c$. In the second step, a TSP is solved for each cluster corresponding to the GAP solution.

In our implementation (H3), we consider the heuristic by Martello and Toth (1981) to solve the GAP. We modified this heuristic to keep ordered the locations assigned to a cluster (route). When a location is assigned to a cluster, we check all the possible positions for insertion in this cluster, and then assign the location to the position that minimizes the length. Therefore, after the application of this assignment step, we already have a solution of the second step (we have a pick up sequence in each route) and we don't need to solve a TSP. Of course, this solution could be easily improved by applying a local search procedure.

At each step of H3, a location is selected and inserted into a route. We compute the candidate list of feasible routes *CL(i)* for each location *i* with those routes satisfying that if *i* is inserted, the length of the new route will be lower than the current value of $t_{max}$, and the number of students in the bus will not exceed the capacity c. If there is no route satisfying the first condition, then it is ignored and *CL(i)* is computed with all the feasible routes (with respect to the capacity and the TMAX value). Then, as proposed in Martello and Toth ( ), we compute $b_i=c_{i\,j''(i)} - c_{i\,j'(i)}$ where:

$$c_{i\,j'(i)} = \min_{j\in CL(i)} c_{ij} \quad \text{and} \quad c_{i\,j''(i)} = \min_{j\in CL(i)-\{j'(i)\}} c_{ij} \,.$$

The location $i*$ with maximum $b$-value is selected and inserted in route $R_{j'(i*)}$ in the position that minimizes its length. The procedure finishes when all the locations have been inserted into one of the $m$ routes.

### *Constructive Heuristic H4*

In this method, we first compute an order of the locations. Then, at each step, the next location *i* in the list is selected for insertion in a route. As in the previous method, we construct a candidate list *CL(i)* of feasible routes where location *i* can be inserted. Then, for each route *j* in *CL(i)*, we compute *value(i,j)* as the value of the length when *i* is inserted in the best position*, p(i,j)*, in the route. Inspired in the construction step of the GRASP methodology (Resende and Ribeiro, 2003), we consider now a restricted candidate list *RCL(i)* with those elements in *CL(i)* with an attractive value:

$$RCL(i) = \{j \in CL(i) \,/\, value(i,j) \le \alpha MinVal(i) + (1\text{-}\alpha)MaxVal(i)\ \},\ where$$

$$MinVal(i) = \min_{j\in CL(i)} value(i,j) \quad \text{and} \quad MaxVal(i) = \max_{j\in CL(i)} value(i,j)\,.$$

Then, a route $j*$ is randomly selected in *RCL(i)* and the location *i* is inserted in the best position of $R_{j*}$ to minimize its length. The constructive method H4 repeats this procedure *niter* times and keeps the best solution found across the different constructions. We have considered five strategies to construct the initial order of locations:

- Far: distance to school, where the farthest location is the first in the list
- Close: distance to school, where the closest location is the first in the list
- Angle: the angle with a fixed line with origin in the school
- Dif: according to the differences bi computed in H3.
- TSP: according to a heuristic TSP solution considering all the locations together.

In the computational experiments reported in Section 4, we will compare these five strategies to select the best one.

## 3. Tabu Search Procedure

Tabu search is by now one of the most successful metaheuristics. It is based on the general telnets of intelligent problem solving, implementing the "memory" in a very strategic and direct way. This methodology is rapidly becoming the method of choice for designing solution procedures for hard combinatorial optimization problems. A comprehensive examination can be found in the book by Glover and Laguna (1997).

In this section we propose a tabu search algorithm for the routing problem (TS_RP) to improve the solutions generated with the constructive methods described above. Therefore, for each possible value of the number of routes, a solution is first constructed and then improved with TS_RP. This improvement method keeps the number of routes fixed while tries to reduce the $t_{max}$ value.

Tabu search begins in the same way as ordinary local search, proceeding iteratively from one solution to another until a chosen termination criterion is satisfied. Exchanges, as introduced in Taillard et al. (1997), are used as the primary move mechanism in TS_RP. We define EXCHANGE_MOVE($R_i,s,R_j,t,l$) as the exchange of the chain of locations $r_i(s)$, $r_i(s+1)$, …, $r_i(s+l)$ from route $R_i$ with the chain of locations $r_j(t)$, $r_j(t+1)$, …, $r_j(t+l)$ from route $R_j$. This operation results in the new routes $R_i'$ *and* $R_j'$, as follows:

$$R_i' = \{\ r_i(1), \ldots, r_i(s\text{-}1), r_j(t), r_j(t+1), \ldots, r_j(t+l), r_i(s+l+1), \ldots, r_i(n_i)\ \}$$
$$R_j' = \{\ r_j(1), \ldots, r_j(t\text{-}1), r_i(s), r_i(s+1), \ldots, r_i(s+l), r_j(t+l+1), \ldots, r_j(n_j)\ \}$$

Figures 1 and 2 illustrates this move. Figure 1 shows an initial configuration of two contiguous routes, and Figure 2 shows the new routes after applying EXCHANGE_MOVE($R_i,s,R_j,t,l$).

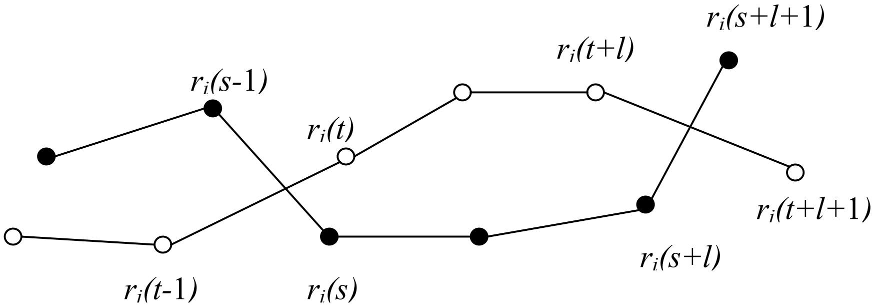

**Figure 1**. Initial configuration of two contiguous routes

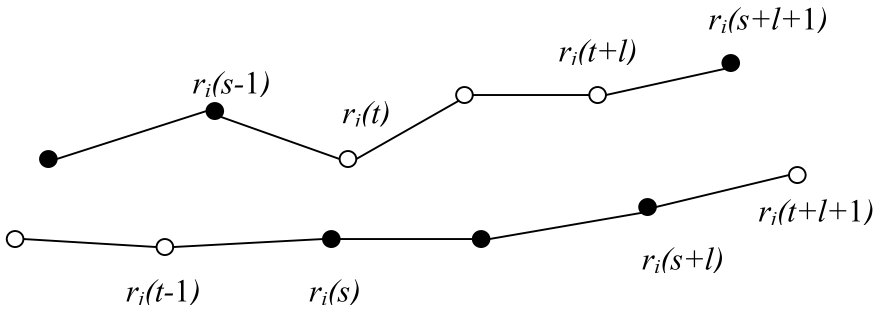

**Figure 2**. Final configuration after applying EXCHANGE_MOVE($R_i,s,R_j,t,l$).

Since our aim is to modify the routes in order to reduce the current value of $t_{max}$, we construct a *candidate list* of moves based on a set of critical and near-critical routes. A critical route $R_i$ is one for which $length(i)=t_{max}$ , while a near-critical route $R_j$ is one for which $length(j)\geq \alpha * t_{max}$ and $1 > \alpha > 0$. Near-critical routes do not determine the value of the objective function $t_{max}$ in the current solution, but they are considered likely to do so in subsequent iterations. In other words, if a move is able to eliminate a critical route, it is likely that a near-critical route will become critical in subsequent iterations. The set that consists of both critical and near-critical routes can be defined as:

$$CR(\alpha) = \{ R_i / length(i)\geq \alpha * t_{max} \} ,$$

where $1 \geq \alpha \geq 0$. In any given iteration, a move is generated by randomly choosing a route $R_i$ in *CR*. We consider all the moves that can be obtained from EXCHANGE_MOVE($R_i$, $s$, $R_j$, $t$, $l$) where $R_j$, is any of the other $m$-1 routes, $s$ and $t$ are locations in $R_i$ and $R_j$ respectively, and $l$ is a search parameter. As it is usual in neighborhood exploration, two strategies can be considered. The *best* strategy selects the move with the largest move value among all the moves in the neighborhood. The *first* strategy, on the other hand, scans the list of routes (in order of proximity) seeking for the first route with an associated movement that results in an strictly positive move value. Both strategies, as well as different $l$-values, will be tested in Section 5.

One of the key elements in heuristic search is the definition of the value of a move. Although the most common practice is to define the move value as the change in the objective function value, in the context of maximum traveling time, the change in the objective function value $t_{max}$ provides little information during the search. As a result, we have defined the value of a EXCHANGE_MOVE($R_i,s,R_j,t,l$) as

$$\text{MoveValue}(R_i, R_j) = max\{ length(i), length(j)\} - max \{ length(i'), length(j')\},$$

where $R_i'$ *and* $R_j'$, are the new routes after the move. The move is executed even when the move value is not positive, resulting in an increasing of a length's route, and eventually, in a deterioration of the current objective function value $t_{max}$. The moved locations become tabu-active for *TabuTenure* iterations, and therefore they cannot be moved back to route $R_i$ during this time. We apply the default aspiration criteria consisting of permitting tabu moves when $t_{max}$ is reduced.

After performing a move, we apply an improvement method to each of both modified routes. We adapt the 3-opt swapping heuristic proposed in Or(1976) in which a chain of consecutive elements is reallocated in another position within the route. We restrict our implementation to chains of 1, 2 or 3 consecutive locations. Specifically, for route $R_i$ and from $j$=1, 2, …, $n_i$-1, the procedure tests the reduction of *length(i)* produced by removing $\{r_i(j)\}$, $\{r_i(j), r_i(j+1)\}$ or $\{r_i(j), r_i(j+1), r_i(j+2)\}$ from its current position and, inserting it in a posterior position in $R_i$ (all posterior positions are tested). If *length(i)* is reduced, the chain of locations is moved. The procedure stops when no more moves are possible that result in a shorter route length.

## 4. Path Relinking Procedure

Path relinking (PR) was originally suggested as an approach to integrate intensification and diversification strategies in the context of tabu search (Glover and Laguna, 1997). This approach generates new solutions by exploring trajectories that connect high-quality solutions, by starting from one of these solutions, called an *initiating solution*, and generating a path in the neighbourhood space that leads toward the other solutions, called *guiding solutions*. This is accomplished by selecting moves that introduce attributes contained in the guiding solutions.

The approach may be viewed as an extreme (highly focused) instance of a strategy that seeks to incorporate attributes of high quality solutions, by creating inducements to favor these attributes in the moves selected. However, instead of using an inducement that merely encourages the inclusion of such attributes, the path relinking approach subordinates other considerations to the goal of choosing moves that introduce the attributes of the guiding solutions, in order to create a "good attribute composition" in the current solution. The composition at each step is determined by choosing the best move, using customary choice criteria, from a restricted set — the set of those moves currently available that incorporate a maximum number (or a maximum weighted value) of the attributes of the guiding solutions. (Exceptions are provided by aspiration criteria, as subsequently noted.) The approach is called path relinking either by virtue of generating a new path between solutions previously linked by a series of moves executed during a search, or by generating a path between solutions previously linked to other solutions but not to each other.

There are many variants of the path relinking methodology (Laguna and Marti, 2003). In this section we propose a "multiple guiding solutions" method for the vehicle routing problem (PR_RP) with the goal of intensify the search in those areas where good solutions have been found.

The procedure stores a small set of high quality (elite) solutions to be used for guiding purposes. Specifically, after each *niter* iterations of the TS_RP, the current solution is compared to the best *nelite* solutions found during the search. If the new solution is better than any one in the elite set, the set is updated. Instead of using attributes of all the elite solutions for guiding purposes, we only use the best half, and the others are used as initiating solutions.

The relinking process implemented in our search may be summarized as follows. The set of elite solutions is constructed during the application of the tabu search method. Every solution of the worst *nelite* / 2 in the elite set (called the initiating solution) is subject to a relinking process by performing moves that incorporate attributes of the best *nelite* / 2 solutions in the elite set (called the guiding solutions) into it. We apply the 3-opt swapping heuristic described above to each initiating solution with modified traveling times $t'_{ij}$. For each pair of locations $(i,j)$ we define $freq(i,j)$ as the number of times both locations appear in consecutive positions in a route in any of the best *nelite* / 2 solutions in the elite set. Then, $t'_{ij}$ is defined as:

$$t'_{ij} = t_{ij} \ ((nelite / 2) - freq(i,j))$$

This modified traveling times, encourages the presence of "good" locations' chains in the re-linked solutions, thus, altering the way in which the solution space is explored. Note that the minimization of the objective function $t_{max}$ is now subordinated to the incorporation of "elite" attributes in the solutions.

During the path relinking phase, a number of intermediate solutions are generated. These intermediate solutions are good candidates for additional exploration by way of applying a local search procedure. We apply a simple descent procedure with the EXCHANGE_MOVE neighborhood and with the original traveling times $t_{ij}$ to intermediate solutions once every *nrelink* path-relinking iterations. The procedure only performs improving moves until no further improvement is possible. The rational for applying local search to every *nrelink*-th solution is based on the fact that consecutive intermediate solutions hardly differ. Therefore, a local search procedure applied to consecutive intermediate solutions would likely converge to the same local optimum. The path relinking process terminates when all *nelite* / 2 initiating solutions have been considered.

## 5. Computational Testing

We have considered, for the experiments, the data of 58 schools in the province of Burgos (Spain) used in the previous work by Corberan et al. (2002). Since the students are already assigned to each school, we have 58 different problems. 42 of these instances correspond to primary (elementary) schools and 16 to secondary (middle) schools. There are only 3 out of the 42 elementary school problems, with enough locations to make the problem difficult enough to differentiate the solution methods (the other are trivial problems). These problems have 46, 49 and 55 locations (i.e., pickup points), and we will refer to them as problems P7, P14 and P41, respectively. All the instances from secondary schools will be considered in our study. The actual transportation problem has a policy in which no route should be longer than 60 minutes, then we will set TMAX=60 in these experiments.

Before testing the effectiveness of our procedure, we perform 4 preliminary experiments. We use the three problems P7, P14 and P41 to perform an initial experiment consisting of comparing the solutions generated by the four constructive methods described in section 2. We first compare the five strategies to construct the initial order of locations in method H4. We have also added a new strategy labeled *Random*, used as a baseline in our comparison, in which the initial ordering of the locations is randomly chosen. Table 1 reports the results ($t_{max}$ for each $m$-value) obtained with the heuristic H4, with *niter*=200 and $\alpha$=0.8, with these five strategies in the three problems considered. It is shown in bold type the best solution found (minimum value of $t_{max}$) for each specific number of buses.

| | **m** | **Initial Order** | | | | | |
|---|---|---|---|---|---|---|---|
| | | **Far** | **Close** | **Angle** | **Dif** | **TSP** | **Random** |
| **P7** | 8 | 75 | 83 | 101 | 93 | 100 | **68** |
| | 9 | 67 | 72 | 92 | 59 | 106 | **64** |
| | 10 | 62 | 73 | 86 | 69 | 106 | **60** |
| | 11 | 57 | 71 | 79 | 67 | 89 | **55** |
| | 12 | **52** | 69 | 73 | 59 | 82 | 55 |
| | 13 | **45** | 62 | 68 | 56 | 74 | 54 |
| | 14 | **45** | 61 | 68 | 53 | 71 | 53 |
| | 15 | **45** | 54 | 67 | 49 | 69 | 49 |
| | 16 | **45** | 55 | 64 | 46 | 62 | 49 |
| | 17 | **45** | 54 | 63 | 46 | 63 | 49 |
| | 18 | **45** | 52 | 58 | **45** | 61 | 48 |
| | 19 | **45** | 49 | 58 | **45** | 59 | 47 |
| | 20 | **45** | 49 | 56 | **45** | 56 | 48 |
| | 21 | **45** | 50 | 52 | **45** | 51 | **45** |
| **P14** | 4 | 60 | **43** | 77 | 57 | 79 | 58 |
| | 5 | 52 | **31** | 65 | 58 | 64 | 45 |
| | 6 | 36 | **31** | 54 | 45 | 59 | 40 |
| | 7 | 30 | **29** | 50 | 34 | 56 | 32 |
| | 8 | **28** | **28** | 45 | 29 | 53 | **28** |
| | 9 | 26 | 26 | 42 | **25** | 49 | 26 |
| | 10 | **22** | 25 | 39 | 30 | 48 | 26 |
| | 11 | **21** | 24 | 39 | 23 | 45 | 24 |
| **P41** | 10 | **59** | 66 | 87 | 60 | 73 | 67 |
| | 11 | 56 | 60 | 82 | **54** | 67 | 61 |
| | 12 | 54 | 59 | 76 | **53** | 57 | 57 |
| | 13 | 50 | 58 | 77 | **48** | 49 | 54 |
| | 14 | **47** | 59 | 74 | 53 | 49 | 51 |
| | 15 | **46** | 56 | 67 | 47 | 48 | 52 |
| | 16 | **46** | 56 | 66 | 50 | **46** | 50 |
| | 17 | **44** | 55 | 62 | 46 | 47 | 49 |

**Table 1**. Constructive method H4 with different strategies

Table 1 shows that the best results are, in general, obtained with the "Far" strategy, since it is able to match the minimum value of $t_{max}$ in 10 out of 14 cases for P7, 3 out of 8 for P14 and, 5 out of 8 cases for P41. It is also interesting to observe that when the number of buses ($m$) is very limited (the firsts rows of each instance), this strategy fails in obtaining the best solution (with the exception of $m$=10 in P41). Note also that the strategy "close" provides the best results in problem P14 given that it obtains 5 out of 8 minimum values. We compare now the four constructive heuristics described in Section 2. Table 2 shows, in the same three instances considered above, the value of $t_{max}$ that each method obtains for each particular value of $m$.

| | m | H1 | H2 | H3 | H4(Far) |
|---|---|---|---|---|---|
| **P7** | 8 | 97 | 71 | **59** | 75 |
| | 9 | 96 | 69 | **55** | 67 |
| | 10 | 75 | 69 | **52** | 62 |
| | 11 | 61 | 69 | **52** | 57 |
| | 12 | 58 | 59 | **45** | 52 |
| | 13 | 56 | 59 | **45** | **45** |
| | 14 | 58 | 54 | **45** | **45** |
| | 15 | 48 | 57 | **45** | **45** |
| | 16 | 48 | 58 | **45** | **45** |
| | 17 | 48 | 58 | **45** | **45** |
| | 18 | 46 | 57 | **45** | **45** |
| | 19 | 46 | 57 | **45** | **45** |
| | 20 | 46 | 53 | **45** | **45** |
| | 21 | **45** | 56 | 51 | **45** |
| **P14** | 4 | 77 | 68 | **41** | 60 |
| | 5 | 34 | 41 | **31** | 52 |
| | 6 | 30 | 35 | **29** | 36 |
| | 7 | 28 | 35 | **26** | 30 |
| | 8 | 26 | 35 | **24** | 28 |
| | 9 | 23 | 32 | **22** | 26 |
| | 10 | 25 | 29 | **22** | **22** |
| | 11 | 21 | 24 | **21** | **21** |
| **P41** | 10 | 76 | 62 | **58** | 59 |
| | 11 | 56 | 62 | **47** | 56 |
| | 12 | 55 | 61 | **47** | 54 |
| | 13 | 48 | 60 | **44** | 50 |
| | 14 | 48 | 61 | **44** | 47 |
| | 15 | 48 | 53 | **44** | 46 |
| | 16 | 48 | 49 | **44** | 46 |
| | 17 | **44** | 49 | **44** | **44** |

**Table 2**. Comparison of constructive methods

Table 2 shows that the method H3 clearly outperforms the other three constructive procedures. H3 obtains the best solution in all the cases considered (with the exception of $m$=21 in problem P7). H4 also produces relative quality solutions since it obtains 12 best solutions out of the 30 cases considered. Regarding the average percent deviation from the best solution known (the minimum value of $t_{max}$ found for each instance and $m$-value after running all procedures during the experiment[*]), H1, H2, H3 and H4 obtain 20.6%, 27.5%, 0.5% and 11.1% respectively.

We have empirically found in these examples, that the $t_{max}$ value obtained with the constructive method H3 cannot be reduced by applying a relative simple improvement mechanism. Specifically, we have implemented a 3-opt exchange heuristic to reduce the length of each route after their construction. This method is not able to reduce the length of the longest route ($t_{max}$) in none of the 30 cases shown in table 2.

In our third preliminary experiment we compare the 'Best' and 'First' strategies described in Section 3. To this end, we design two local search descent procedures, LS(Best) with the best strategy, and LS(First) with the first strategy. These methods implement the neighborhood described in Section 3 but they do not incorporate any tabu search element, since the purpose of the experiment is to determine which neighborhood exploration to implement within TS_RP. They apply, from the solution constructed with H3, a simple descent procedure in which only improving moves are performed and they stop when no further improvement is possible. Table 3 shows, in the same three instances considered in the previous experiments, the value of $t_{max}$ that each method obtains for each particular

[*] We cannot assess how close these best values are from the optimal solutions (efficient frontier), so we are only using these values as a way of comparing the methods.

value of $m$. This table shows that both strategies provide almost the same results. Therefore, given that the 'Best' strategy performs a significantly higher number of operations than the 'First', we will implement the 'First' strategy in our solving method.

| | m | H3 | H3+LS(Best) | H3+LS(First) |
|---|---|---|---|---|
| **P7** | 8 | 59 | 58 | 58 |
| | 9 | 55 | 55 | 55 |
| | 10 | 52 | 51 | 51 |
| | 11 | 52 | 49 | 49 |
| | 12 | 45 | 45 | 45 |
| | 13 | 45 | 45 | 45 |
| | 14 | 45 | 45 | 45 |
| | 15 | 45 | 45 | 45 |
| | 16 | 45 | 45 | 45 |
| | 17 | 45 | 45 | 45 |
| | 18 | 45 | 45 | 45 |
| | 19 | 45 | 45 | 45 |
| | 20 | 45 | 45 | 45 |
| | 21 | 51 | 46 | 45 |
| **P14** | 4 | 41 | 41 | 41 |
| | 5 | 31 | 31 | 31 |
| | 6 | 29 | 29 | 29 |
| | 7 | 26 | 26 | 26 |
| | 8 | 24 | 23 | 23 |
| | 9 | 22 | 22 | 22 |
| | 10 | 22 | 22 | 22 |
| | 11 | 21 | 21 | 21 |
| **P41** | 10 | 58 | 58 | 58 |
| | 11 | 47 | 47 | 47 |
| | 12 | 47 | 47 | 47 |
| | 13 | 44 | 44 | 44 |
| | 14 | 44 | 44 | 44 |
| | 15 | 44 | 44 | 44 |
| | 16 | 44 | 44 | 44 |
| | 17 | 44 | 44 | 44 |

**Table 3**. Comparison of search strategies

We performed a final preliminary experiment to determine the best values for the parameters in the TS_RP and PR_RP methods. For each search parameter, we tested 3 values:

$\alpha = 0.3, 0.5, 0.8$
$l = 3, 5, 10$
$nelite = 4, 10, 20$
$nrelink = 3, 5, 7$

We do not reproduce now the tables for this experiment. The parameter combination that yielded the best results was $\alpha = 0.8$, $l = 3$ $nelite = 20$ and $nrelink = 3$. These values are used throughout the rest of our experimentation.

In the experiment to test the efficiency of the proposed method, we employ the 16 problem instances corresponding to secondary (middle) schools ( S1, S2,.. , S16). We compare the performance of tabu search with path relinking method (TS_RP) and scatter search, as implemented in Corberan et al. (2002) (SS_RP). We also compare against the solutions currently implemented in practice. Table 4 shows the results of this experiment. The columns in this table consist of: (1) identifier of instances, (2) objective values ($m/t_{max}$) of the solutions currently implemented, (3-7) a partial segment of the approximation of the efficient frontier generated with our TS_RP procedure (columns labeled $m$-1 to $m$+3 show the respective $t_{max}$ value), (8-12) a partial segment of the approximation of the efficient frontier generated with the scatter search method. The $m$ variable used in columns 3-12 corresponds to the current value given in column 2.

| Problem | Current | TS_RP | | | | | SS_RP | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | *m / tmax* | *m*-1 | *m* | *m+1* | *m+2* | *m*+3 | *m*-1 | *m* | *m*+1 | *m*+2 | *m*+3 |
| S1 | 12 / 70 | 52 | 51 | 48 | 47 | 48 | 57 | 56 | 53 | 51 | 48 |
| S2 | 5 / 45 | 46 | 36 | 32 | 29 | | 46 | 36 | 32 | 29 | |
| S3 | 6 / 60 | 52 | 45 | 43 | 39 | | 54 | 46 | 43 | 39 | |
| S4 | 3 / 70 | | 52 | 41 | 37 | 33 | | 52 | 42 | 37 | 33 |
| S5 | 4 / 60 | 55 | 45 | 39 | | | 55 | 44 | 39 | | |
| S6 | 4 / 80 | 79 | 62 | 52 | 45 | 43 | 81 | 62 | 53 | 47 | 44 |
| S7 | 6 / 60 | 51 | 45 | 37 | 36 | | 51 | 45 | 38 | 36 | |
| S8 | 9 / 75 | 59 | 53 | 48 | 45 | 42 | 61 | 59 | 50 | 47 | 44 |
| S9 | 5 / 90 | 90 | 65 | 53 | 49 | 46 | 82 | 65 | 53 | 50 | 48 |
| S10 | 6 / 60 | 43 | 41 | 40 | | | 44 | 41 | 40 | | |
| S11 | 4 / 60 | 42 | 40 | 40 | 40 | | 67 | 51 | 45 | 39 | |
| S12 | 2 / 25 | | 15 | 14 | 9 | | | 15 | 14 | 9 | |
| S13 | 6 / 45 | 40 | 34 | 29 | 29 | | 39 | 34 | 32 | 29 | |
| S14 | 5 / 60 | 53 | 45 | 37 | | | 53 | 46 | 37 | | |
| S15 | 7 / 50 | 49 | 42 | 41 | 40 | | 50 | 44 | 42 | 40 | |
| S16 | 2 / 60 | 84 | 51 | 38 | 35 | | 84 | 51 | 38 | 35 | |

**Table 4**. Comparison of the methods

The results in Table 4 disclose the advantage of using tabu search with path relinking (TS_RP) when compared to scatter search (SS_RP) for this bus routing problem. The TS_RP approach is able to obtain better $t_{max}$ values than SS in 10 problems for the *m*-values considered. SS_RP obtains better results in 1 out of the 16 problems considered, and in the remaining 5 examples, both methods provide the same results. For instance, in problem S8, the TS_RP solution has a $t_{\text{max}}$ of 53 minutes with 9 buses while SS_RP yields a $t_{\text{max}}$ of 59 minutes with the same number of buses (see entry in row S8 and column *m* for both methods). Moreover, the quality of the solutions found by our new method comes at a remarkable low computational effort. TS_RP presents an average running time of 0.29 seconds while SS_RP averages 1.41 seconds (the experiments were performed in a Pentium 4 2.4 GHz). Anyway, the solution times are not critical in this context, because the routes do not change in real time, so there is no problem on spending a few minutes to compute the routes that will be used for a whole academic year. Both procedures clearly outperform the solution currently implemented in practice.

Figure 3 shows the average $t_{max}$ values, across the 16 problems considered, of the approximation of the efficient frontier generated with both methods. This figure clearly shows the improvement achieved with our proposed method compared with the previous approach in terms of approximating the efficient frontier with better non-dominated solutions ( a 3% of improvement on average).

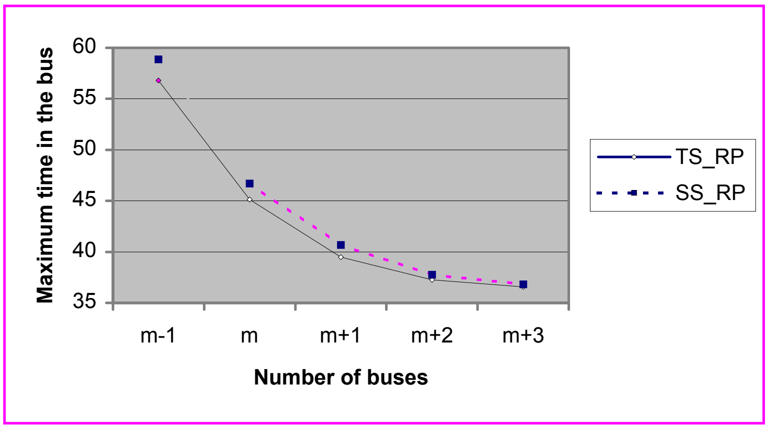

**Figure 3**. Approximation of the efficient frontier

## Conclusions

We have developed a heuristic procedure based on the tabu search methodology to provide high quality solutions to the problem of routing school buses in a rural area. Our method deals with the two objective functions of this problem simultaneously, cost and quality of service, thus providing an approximation of the efficient frontier. The implementation was shown competitive in a set of problem instances for which a recently scatter search method was applied. Tabu search with path relinking performed remarkably well, outperforming this scatter search implementation, which up to now was the best procedure reported in the literature for this problem. Our method, as well as the scatter search, outperform the solutions currently implemented in practice.

## References

Bodin, L. D. and L. Berman (1979) “Routing and Scheduling of School Buses by Computer,” *Transportation Science,* vol. 13, no. 2, pp. 113-129.

Bodin, L., B. Golden, A. Assad and M. Ball (1983) “Routing and Scheduling of Vehicles and Crews: The State of the Art,” *Computers and Operations Research,* vol. 10, no. 2, pp. 63-211.

Chapleau, L. J. A. Ferland, G. Lapalme and J-M. Rousseau (1984) “A Parallel Insert Method for the Capacitated Arc Routing Problem,” *Operations Research Letters,* vol. 3, no. 2, pp. 95-99.

Corberán, A., E. Fernández, M. Laguna and R. Martí (2002), “Heuristic Solutions to the Problem of Routing School Buses with Multiple Objectives” *Journal of the Operational Research Society,* vol. 53, no. 4, pp. 427 – 435.

Fisher, M. L. and R. Jaikumar (1981). "A Generalized Assignment Heuristic for Vehicle Routing". *Networks*, vol.11, nº 2, 109-124.

Glover, F. (1998) “A Template for Scatter Search and Path Relinking,” in *Artificial Evolution, Lecture Notes in Computer Science* 1363, J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer and D. Snyers (Eds.), Springer-Verlag, pp. 13-54.

Glover, F. and M. Laguna (1997) Tabu Search, Kluwer Academic Publishers, Boston.

Glover, F., M. Laguna and R. Martí (1999) “Scatter Search,” to appear in Theory and Applications of Evolutionary Computation: Recent Trends, A. Ghosh and S. Tsutsui (Eds.), Springer-Verlag.

Laguna, M. and R. Martí (2003) *Scatter Search – Methodology and Implementations in C,* Kluwer Academic Publishers, Boston.

Martello, S. and P. Toh (1981). “An algorithm for the generalized assignment probem”, in *Operational Research ’81*, North Holland, Amsterdam, 589-603.

Or, I. (1976). “Traveling Salesman Type Combinatorial Problems y their Relations to the Logistics of Blood Banking.” *Ph. Thesis.* Dpt. of Industrial Engineering y Management Sciences, Northwestern Univ.

Resende, M. G. C. and C. C. Ribeiro (2003) “Greedy Randomized Adaptive Search Procedures,” in *State-of-the-Art Handbook in Metaheuristics*, F. Glover and G. Kochenberger (Eds.), Kluwer Academic Publishers, Boston, 219-250.

Taillard, E., P. Badeau, M. Gendreau, F. Guertain and J.Y. Potvin (1997). "A Tabu Search heuristic for the Vehicle Routing Problem with Time Windows". *Transportation Science.* Vol. 31, pp 170-186.