Heuristic Solutions to the Problem of Routing School Buses with Multiple Objectives

ANGEL CORBERÁN

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

ELENA FERNÁNDEZ

Departament d'Estadística i Investigació Operativa, Universitat Politècnica de Catalunya, Barcelona, Spain, E.Fernandez@upc.es

MANUEL LAGUNA[†]

Graduate School of Business Administration, University of Colorado, Boulder, CO 80309-0419, USA
manuel.laguna@Colorado.edu

RAFAEL MARTÍ

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

Latest version: August 7, 2000

Abstract — In this paper, we address the problem of routing school buses in a rural area. We approach this problem with a node routing model with multiple objectives that arise from conflicting viewpoints. From the point of view of cost, it is desirable to minimize the number of buses used to transport students from their homes to school and back. And from the point of view of service, it is desirable to minimize the time that a given student spends in route. The current literature deals primarily with single-objective problems and the models with multiple objectives typically employ a weighted function to combine the objectives into a single one. We develop a solution procedure that considers each objective separately and search for a set of efficient solutions instead of a single optimum. Our solution procedure is based on constructing, improving and then combining solutions within the framework of the evolutionary approach known as scatter search. Experimental testing with real data is used to assess the merit of our proposed procedure.

[†] Partially supported by the visiting professor fellowship program of the University of Valencia (Grant Ref. No. 42743).

1. Introduction

The problem of scheduling and routing school buses deals with the important question of how to transport students to and from schools in the safest, most economical and most convenient manner. The scheduling and routing activities are often controversial because the problem must deal with multiple objectives. Although the vehicle routing literature in general has dealt with many objectives, the most relevant in the context of routing and scheduling of school buses are:

1. to minimize the transportation cost
2. to minimize the transportation time

Several alternatives exist when tackling these objectives. For example, the transportation cost can include capital and operational costs. The transportation time may be measured in terms of the total length of the route, the average time or the maximum time that a student spends in the bus. Since the motivation of our work stems from a specific transportation problem in large and sparse rural area, we consider the objectives of minimizing the total number of buses while simultaneously minimizing the maximum time that a student spends in the bus. Specifically, we have considered the bus routing problem in primary and secondary schools in the Province of Burgos (Spain). The current solution to this problem has been the focus of controversy because of severe service deficiencies. In this paper we use real data from these schools to show how the level of service can be increased with the same capital expenditures. Alternative solutions are also given where administrators and school officials can analyze the tradeoff between cost and quality of service.

In the scheduling and routing of school buses in a sparse rural area, the routing is the key element for finding good solutions. As mentioned in Bodin and Berman (1979), in a rural area, routes tend to be long and the buses do not reach their maximum physical capacity. That is, the buses reach an allowed route length (measured in terms of time) before they are completely full. In addition to the features that are common to transportation problems in rural areas, in the particular problem that we are tackling, we can ignore the time required to reach the first student. This is due to the definition of transportation time that we use, where we try to minimize the maximum time that a student spends in the bus. Since the bus is empty before picking up the first student, the transportation time before the first pickup occurs can be ignored. Although our problem occurs in a rural area, it is a single load problem. In the single load problem, routes either pick up or deliver students to a single school. Note that some rural areas consider the mixed problem, where a single bus picks up students from more than one school. In this situation, some students are delivered to one school and others remain on the bus and are transported to a different school along with students that are picked up between the two schools. Additional characteristics of a variety of routing and scheduling problems, including those related to school buses, are described in the comprehensive survey by Bodin, et al. (1983).

We approach this problem with a solution procedure based on the evolutionary approach called scatter search, which originated from strategies for creating composite decision rules and surrogate constraints. Recent studies have demonstrated the practical advantages of this approach for solving a diverse array of optimization problems from both classical and real world settings, including problems in transportation and routing. Scatter search, in contrast to other evolutionary procedures such as genetic algorithms, provides unifying principles for joining solutions based on generalized path constructions in neighborhood and Euclidean space and utilizes strategic designs where other approaches resort to randomization. Additional advantages are provided by intensification and diversification mechanisms that exploit adaptive memory, drawing on foundations that link scatter search to tabu search.

In our application, we use the referent set of solutions in scatter search as a repository of efficient and non-efficient solutions that can be used to generate new ones. Efficient solutions are members of the reference set due to their quality, as measured by the values associated with the multiple objective functions. Dominated solutions are also part of the reference set to provide the diversity necessary to explore the solution space. Our implementation, as described subsequently, proposes innovative mechanisms to update and maintain the reference set of solutions as well as strategies for combining solutions. Experimental testing with real data is used to assess the merit of our proposed procedure.

2. Problem Description

Consider the problem 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 different capacities. We use the following notation to give a formal description of the problem:

$N = \{ 0, 1, \dots, n \}$	:	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 jth 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_i	:	the capacity of a bus i
q_j	:	the number of students to be picked up at location j, for $j = 1, \dots, n$
$length(i)$	:	the length of route i (which is also the maximum traveling time corresponding to the students picked up at the first location)

Note that according to our definitions, the length of route i is calculated as:

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

The bus routing problem is to find a set of routes in order to:

$$\text{Minimize} \quad m \tag{1}$$

$$\text{Minimize} \quad t_{\max} = \max_{i \in M} \{ length(i) \} \tag{2}$$

$$\text{subject to} \quad \sum_{j=1}^{n_i} q_{r_i(j)} \le c_i \qquad \text{for } i = 1, \dots, m \tag{3}$$

$$\sum_{i=1}^{m} n_i = n \tag{4}$$

$$r_i(j) \ne r_k(j) \qquad \forall\ i,k \in M \text{ and } \forall\ j \in N \tag{5}$$

Objective function (1) attempts to minimize the number of buses while objective function (2) minimizes the maximum time in the bus. Note that the maximum time in the bus is also the route length measured from the point that the first students are picked in location $r_i(1)$. Constraints (3) enforce the physical capacity of each bus. In rural settings, as we discussed in the introduction, these constraints are typically not binding. Equations (4) and (5) indicate that all the students must be picked up and that a given location cannot be assigned to more than one route.

The objectives in our problem are in conflict. That is, a solution that minimizes the number of buses tends to increase the maximum traveling time, and vice versa. In fact, if a single bus could have enough physical capacity to serve all the locations, this will be the trivial solution to the problem that considers only the first objective function. Similarly, the trivial solution to the problem that considers only the second objective function is to utilize one bus for each location, i.e., to make $m = n$. A common approach when dealing with multi-objective problems is to create a composite objective function that is a weighted combination of the individual objectives. For instance, the following objective function could be used as a proxy of our bi-objective function:

$$\text{Minimize } m + \lambda\, t_{\max}$$

The advantage of using a composite function is that all the known methodology to search for optimal solutions to single-objective problems can be applied. This is not a trivial item considering the amount of research that has been direct to develop exact and heuristic procedures for single-objective optimization problems. The disadvantage, however, is that in practical settings it is unrealistic to expect that the decision makers can agree on a λ value that represents their desire for trading off cost and service. This disadvantage can be overcome by solving the problem with several λ values. Nevertheless, a more direct approach is to solve the problem considering both objective functions separately and giving the decision maker a set of solutions that represent the "the best possible service" at a given cost. Since the number of buses in a solution is bounded by n, because in our setting it is possible to assume that $q_j \le c_i$ for $j \in$ N and $i \in$ M, a practical approach is to minimize $t_{\max}$ for each possible value of m. The solution approach that we describe in the following section follows this general philosophy without taking it to the extreme of treating objective (1) as a constraint in the model.

3. Solution Approach

The solution approach that we have developed for our bus routing problem consists of an adaptation of scatter search. Scatter search (SS) is a novel instance of evolutionary methods, because it violates the premise that evolutionary approaches must be based solely on randomization — though they likewise are compatible with randomized implementations such as the one we describe here. SS is also novel, in comparison to the well known genetic algorithm (GA) class of evolutionary methods, by being founded on strategies that only piecemeal came to be proposed as augmentations to GAs more than a decade after their debut in scatter search. Scatter search embodies principles and strategies that are still not emulated by other evolutionary methods, and that prove advantageous for solving a variety of complex optimization problems. More about the origin and multiple applications of scatter search can be found in Glover (1998), Glover, Laguna and Martí (1999) and Laguna (2000).

Our adaptation consists of the following elements:

H1 and H2:	Two constructive heuristics to generate routes
SWAP:	An exchange procedure to find a local optimal value for the length of each route
INSERT:	An exchange procedure to improve upon the value of $t_{\max}$
COMBINE:	A mechanism to combine solutions in a reference set of solutions in order to generate new ones.

The elements in our procedure are designed in such a way that the routes continue to approach the school as the bus moves. We first describe each element and then summarize the entire procedure, which provides an overall view of how these elements interact.

Constructive Heuristic H1

This heuristic is based on a clustering mechanism. However, the procedure keeps the locations assigned to each cluster ordered, and therefore the clusters can be referred to as routes. The procedure starts with the following n routes:

$$R_i = \{\ i\ \} \text{ for } i = 1, \ldots, n$$

Candidate List — We build an ordered candidate list with the *BestPairs* best route pairs (R_i, R_k). The best pair of routes is the one with the minimum traveling time between the two routes. When the routes in a given pair contain a single location, the time between the pair of routes is simply the traveling time between their corresponding locations. When a route has more than one location, we consider the endpoints of the routes to find the minimum distance. That is, the minimum distance between routes R$_i$ and R$_k$ is given by:

$$\min\left\{t_{r_i(1)r_k(n_k)}, t_{r_k(1)r_i(n_i)}\right\}$$

Note that this assumes that there are only two ways in which routes R$_i$ and R$_k$ can be joined:

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

Merging — We randomly choose one pair of routes from the candidate list. Let routes *i* and *k* be such that $t_{r_k(n_k),0} \le t_{r_i(n_i),0}$. We attempt to merge the routes, considering R′ first and then R″. If the merging R′ is feasible, we stop. A feasible merging of routes R$_i$ and R$_k$ is such that the resulting route does not violate either of the following inequalities:

route length ≤ TMAX (6)
number of students in the bus ≤ capacity (7)

Note that TMAX is a target value for the second objective function value. This value is necessary to control the length of the routes generated with H1, so the routes do not become impracticably long. The procedure attempts to merge up to *TrialPairs* pairs using the same candidate list, where *TrialPairs* < *BestPairs*. After *TrialPairs* merging attempts have been made, the candidate list is rebuilt. The process stops when no more routes can feasibly merge. The output of the heuristic is a set of solutions with number of routes ranging from m_l to m_h, where $m_l \ge 1$ and $m_h \le n$.

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 (*Angle*). When a location is selected and it does not belong to an already defined sector, a new sector is defined around the chosen location (see Figure 1). This is the case, for instance, when the procedure starts and the first location is chosen. In subsequent selections, we check whether the chosen location belongs to a sector or not.

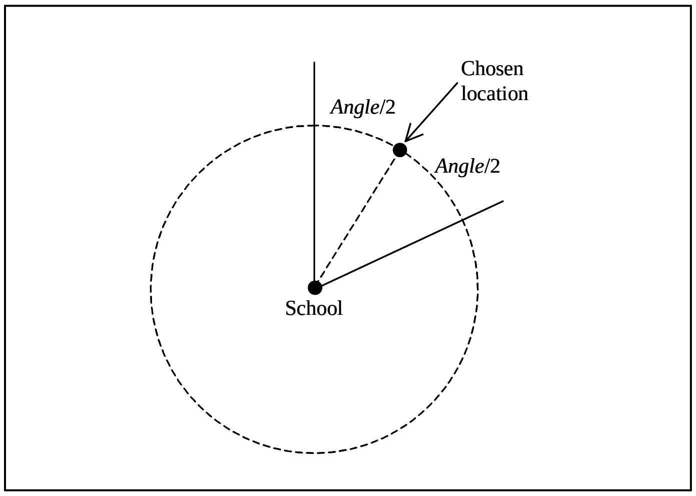

Figure 1. Definition of a sector around a chosen location.

The locations are ordered according to the decreasing value of t_{j0} (the traveling time from location *j* to the school) for *j* = 1, …, *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 α 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 (6) and (7) or a new sector is defined. To test the feasibility constraint (6), 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 $t_{\max}$.

Exchange Procedure SWAP

This exchange procedure has the goal of reducing the length of a route. The procedure is applied to route k, for $k = 1, \ldots, m$, and the outcome is a locally optimized route length. From $i = 1, \ldots, n_k-1$ and $j = i+1, \ldots, n_k$, the procedure tests the reduction of $length(k)$ produced by exchanging the positions of $r_k(i)$ and $r_k(j)$. If $length(k)$ is reduced, the locations are exchanged. The procedure stops when no more exchanges are possible that result in a shorter route length.

Exchange Procedure INSERT

This exchange mechanism is based on removing a location from one route and inserting it into another route. We select i, the index of the longest route, i.e., $length(i) = t_{\max}$. If more than one route has the maximum length one is arbitrarily chosen. A location v is chosen such that the removal of this location from route i causes the maximum decrease in $length(i)$. We then search for a route k where to insert location v. We build a list of *NearLoc* locations w that consists of the nearest locations to v. We attempt to insert v before and after w (which is currently in route k) provided:

1. $k \neq i$
2. $length(k) + slack \leq length(i)$
3. number of students in route $k + q_v \leq c$

We select the best insertion of the two. The procedure is performed until no insertion is found that results in a decrease in the value of $t_{\max}$. Our insertion procedure is similar to the one suggested in Chapleau, et al. (1984) in the context of the capacitated arc routing problem.

Combination Method COMBINE

This method consists of generating new solutions from the combination of two existing solutions. The procedure starts by matching the routes from one solution to the routes of the other solution. The rationale behind route matching is to find the common structure of the two solutions being combined to transfer this structure to the newly generated solution. After the solutions are matched, the procedure generates a newly combined solution by a voting mechanism. The combination method can be better explained with an illustrative example.

Suppose that we would like to combine the following two solutions (A and B) to a problem with $n = 10$. Both solutions have two routes and we assume that A_i is the ith route for solution A and B_i is the ith route for solution B:

$A_1 = \{ 4, 2, 7, 1 \}$
$A_2 = \{ 5, 10, 6, 9, 3, 8 \}$

$B_1 = \{ 2, 6, 8, 10, 9 \}$
$B_2 = \{ 3, 4, 7, 1, 5 \}$

The matching procedure builds a square matrix *match*, where the (i,k) element contains the number of common elements between route i and k (i.e., the cardinality of the intersection of the elements in the matched routes). The matching matrix for our example looks as follows:

	B_1	B_2
A_1	1	3
A_2	4	2

Note that A_2 and B_1 have four elements in common (6, 8, 9 and 10) while A_1 and B_2 have 3 elements in common (4, 7 and 1), resulting in *match*(1,2) = 3 and *match*(2,1) = 4. Also, *match*(1,1) = 1 because $A_1 \cap B_1 = \{ 1 \}$ and *match*(2,2) = 2 because $A_2 \cap B_2 = \{ 3, 5 \}$. According to the maximum matching values, we match A_2 with B_1 and consequently A_1 and B_2 to generate, respectively, routes N_1 and N_2 in a new solution. The new solution is constructed in *n* steps, in which one location is assigned to a route in each step. The matched pairs are alternatively used for each assignment. Table 1 shows the sequence of assignments associated with our illustrative example.

Table 1. Combination steps.

Step	Pair	Vote 1	Vote 2	Assignment	Selection rule
1	(A_1,B_2)	4	3	$N_1 = \{ 4 \}$	random
2	(A_2,B_1)	5	2	$N_2 = \{ 2 \}$	random
3	(A_1,B_2)	7	3	$N_1 = \{ 4, 3 \}$	3 before 7
4	(A_2,B_1)	5	6	$N_2 = \{ 2, 5 \}$	5 before 6
5	(A_1,B_2)	7	7	$N_1 = \{ 4, 3, 7 \}$	same location
6	(A_2,B_1)	10	6	$N_2 = \{ 2, 5, 6 \}$	random
7	(A_1,B_2)	1	1	$N_1 = \{ 4, 3, 7, 1 \}$	same location
8	(A_2,B_1)	10	8	$N_2 = \{ 2, 5, 6, 10 \}$	10 before 8
9		9	8	$N_2 = \{ 2, 5, 6, 10, 8 \}$	8 before 9
10		9	9	$N_2 = \{ 2, 5, 6, 10, 8, 9 \}$	same location

In step 1, we arbitrarily start building route N_1 with the pair (A_1,B_2). The first element in route A_1 is location 4, and therefore this route votes for location 4 to be in the first position of N_1. The vote is shown in the column labeled "Vote 1". The vote associated with B_2 goes to location 3 and is shown in the column labeled "Vote 2". Since locations 4 and 3 occupy the first position in their respective routes, we break the tie by randomly choosing one location. The chosen location is number 4 and is placed in the first position of route N_1. The assignment in step two is similar to the assignment in step 1, with the first location for N_2 chosen randomly between the first locations of A_2 and B_1. In step 3, A_1 votes for location 7 and B_2 votes for location 3. Note that A_1 cannot vote for location 2, because this location was already assigned to N_2 in the previous step. Since location 7 is in the third position of route A_1 and location 3 is in the first position of route B_2, the rule is to give preference to the location in the earlier position and therefore location 3 is selected. Note that this rule is also applied in steps 4, 8 and 9. Also note that in steps 5, 7 and 10, the same location receives both votes.

Note that although in rural areas buses are normally operating under capacity, the combination procedure must consider that the new routes cannot violate the physical capacity of the buses. For simplicity, we did not address this issue in the description above. However, in the cases when an assignment cannot be made because of a violation in the bus capacity, the location is placed in a special unassigned set. We attempt to assign the locations in the special unassigned set to the new routes using a fast bin-packing heuristic. If the heuristic fails, the combined solution is then discarded.

Scatter Search Outline

The overall procedure operates as follows and is outlined in Figure 2. The constructive heuristics H1 and H2 are applied with several values for TMAX and the resulting solutions are stored in separate pools, one for each value of *m*. As expected, the larger the value of TMAX the larger the frequency in which the heuristics construct solutions with a small number of routes. Conversely, solutions with large number of routes are obtained when the value of TMAX is decreased. The procedure then attempts to improve upon the solutions constructed by H1 and H2. The improvement consists of first applying SWAP to each route and then applying INSERT to the entire solution. If any route is changed during the application of INSERT then we apply SWAP one more time to all the changed routes. The procedure now iterates within a main loop, in which a search is launched for solutions with a common number of routes. The main loop terminates when all the *m*-values have been explored.

From all the solutions with *m* routes, choose the best *b* to initialize the reference set (*RefSet*). Note that the criterion for ranking the solutions at this step is $t_{\max}$, since all solutions have the same number of routes. The procedure performs iterations in an inner-loop that consists of searching for a solution with *m* routes with an improved $t_{\max}$ value. The combination procedure COMBINE is applied to all pairs of solutions in the current reference set *RefSet*. Since the reference set consists of *b* solutions, the number of solutions generated by COMBINE is $(b^2\text{-}b)/2$. The combined solutions are improved in the same way as described above, that is, by applying SWAP then INSERT and finally SWAP to the routes that changed during the application of INSERT. We refer to the resulting set of distinct solutions as *ImpSet*. The reference set is then updated by selecting the best *b* solutions from the union of *RefSet* and *ImpSet*. Steps 5, 6 and 7 in the outline of Figure 2 are performed as long as at least one new solution is admitted in the reference set.

```
1. Construct solutions — Apply constructions heuristics H1 and H2
   with several values of TMAX.
2. Improve solutions — Apply SWAP to each route in a solution and
   INSERT to the entire solution.  Finally, apply SWAP o any route
   changed during the application of INSERT.
3. Build solution pools — Put all solutions with the same number of
   routes in the same pool.
for ( each solution pool ) do
    4. Build the reference set — Choose the best b solutions in the
       pool to build the initial RefSet.
    while ( new solutions in RefSet ) do
        5. Combine solutions — Generate all the combined solutions
           from pairs of reference solutions where at least one
           solution in the pair is new.
        6. Improve solutions — Apply SWAP to each route in a
           solution and INSERT to the entire solution.  Finally,
           apply SWAP o any route changed during the application
           of INSERT.
        7. Update reference set — Choose the best b solutions from
           the union of the current reference set and the combined-
           improved solutions to update the RefSet.
    end while
end for
```

Figure 2. Scatter search outline.

Note that after the reference set is updated, the combination procedure may be applied to the same solution pairs more than once. However, the combination procedure includes some randomized elements and therefore the combination of two solutions may result in a different outcome every time COMBINE is applied. Also, the size of the reference set is increased if the updating procedure fails to add at least one new solution. The additional solutions come from the original pool of solutions generated with the construction heuristics. The reference set size is increased up to 2**b*, where *b* is the initial size.

4. Computational Testing

All the experiments reported in this section were performed in a Pentium III 700 MHz machine. In the real-world problem that motivated this study, there are 42 primary (elementary) schools and 16 (middle) secondary schools. Since the students are already assigned to each school, we have a set of 58 bus routing problems. From the set of 42 elementary school problems, there are only 3 that have enough locations to make the testing of our procedure interesting. 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. The rest of the problems in the set have significantly less locations, making their solution a trivial exercise. We use these three problems to perform an initial experiment consisting of comparing the solutions generated by H1 and H2. The parameter values used for experimentations are:

NearLoc = 4
BestPairs = 20
TrialPairs = 5
Angle = 36
$\alpha = 0.75$
TMAX = 30, …, 90
b = 20

Figure 3 shows 1400 solutions obtained with H1 for problem P7, where the *y*-axis represents the value of $t_{\max}$ and the *x*-axis represents the value of *m*.

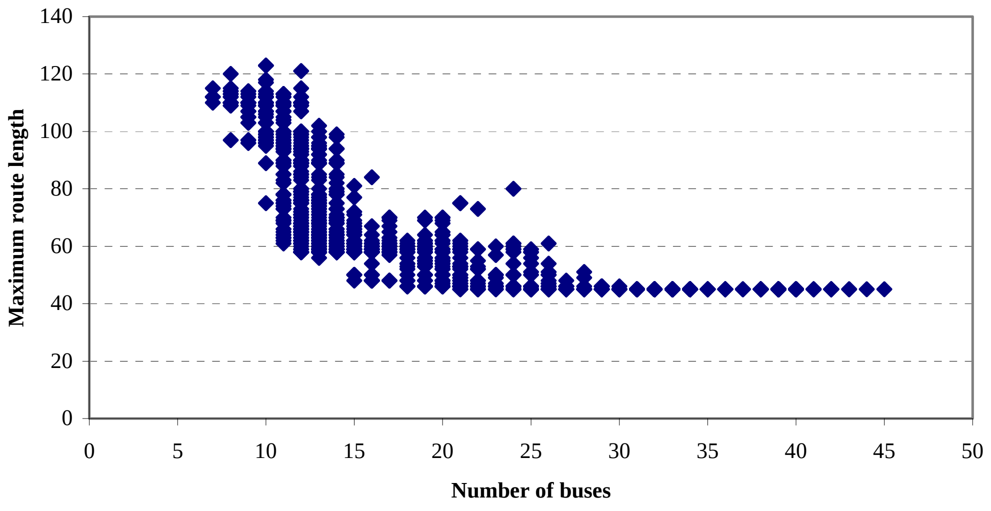

Figure 3. Solutions generated with H1 for problem P7.

Note that the maximum route length hits a barrier at 45 minutes and it does not decrease even if the number of buses is set to almost 50. Figure 4 shows 1650 solutions generated with H2 for the same problem P7. An obvious difference between the solutions generated by H1 and those generated by H2 is their dispersion with respect to the maximum route length. While H1 tends to generate more dispersed solutions, H2 tends to generate solutions of similar quality for each number of buses. Since our construction heuristics have complementary characteristics, we use both to generate the initial pool of solutions to be subjected to the scatter search.

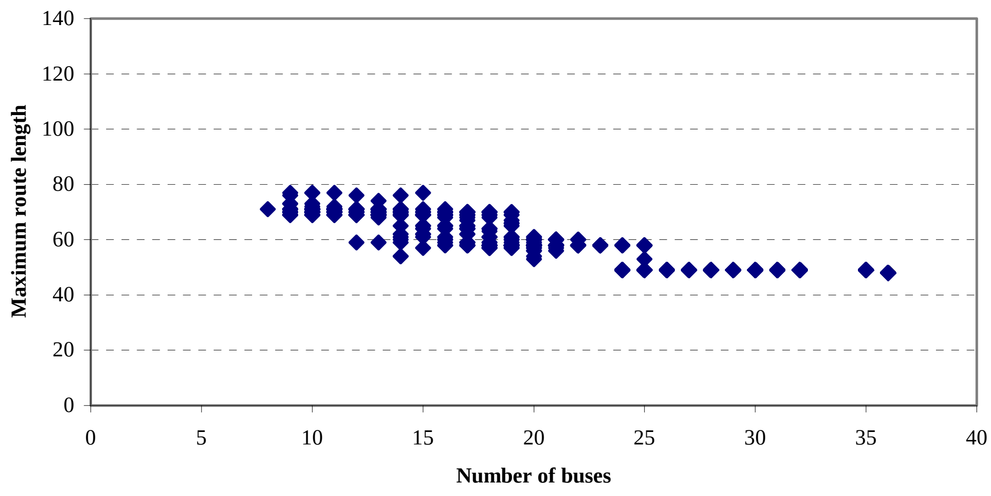

Figure 3. Solutions generated with H2 for problem P7.

Table 2 shows the best $t_{\max}$ associated with a specific number of buses for each procedure and problem instance. Note that H1 not only produces solutions that are more dispersed, but also the best solutions tend to be better than those constructed by H2.

Table 2. $t_{\max}$ values of routes for three elementary school problems.

	P7		P14		P41	
Buses	H1	H2	H1	H2	H1	H2
4			77	68		
5			34	41		
6			30	35		
7	110		28	35		
8	97	71	26	35		63
9	96	69	23	32		62
10	75	69	25	29	76	62
11	61	69	21	24	56	62
12	58	59		23	55	61
13	56	59		27	48	60
14	58	54		23	48	61
15	48	57		29	48	53
16	48	58		31	48	49
17	48	58		31	44	49
18	46	57		31		49
19	46	57		47		49
20	46	53		23		49
21	45	56				48
22		58				47
23		58				47
24		49				46
25						45
26						45
27						44

An interesting research question regarding the performance of our algorithm relates to the contribution of scatter search to the final quality of the solutions for each number of buses. Figure 4 shows the approximation of the efficiency frontier for the solutions generated with H1 compared to the approximation that results after the application of scatter search. Although there are 5 instances (number of buses) in which the scatter search is not capable of improving upon the best solution generated by H1, in the remaining instances the average improvement is 15%. The average CPU time to run the scatter search procedure to generate each solution in the approximate efficient frontier is 0.6 seconds.

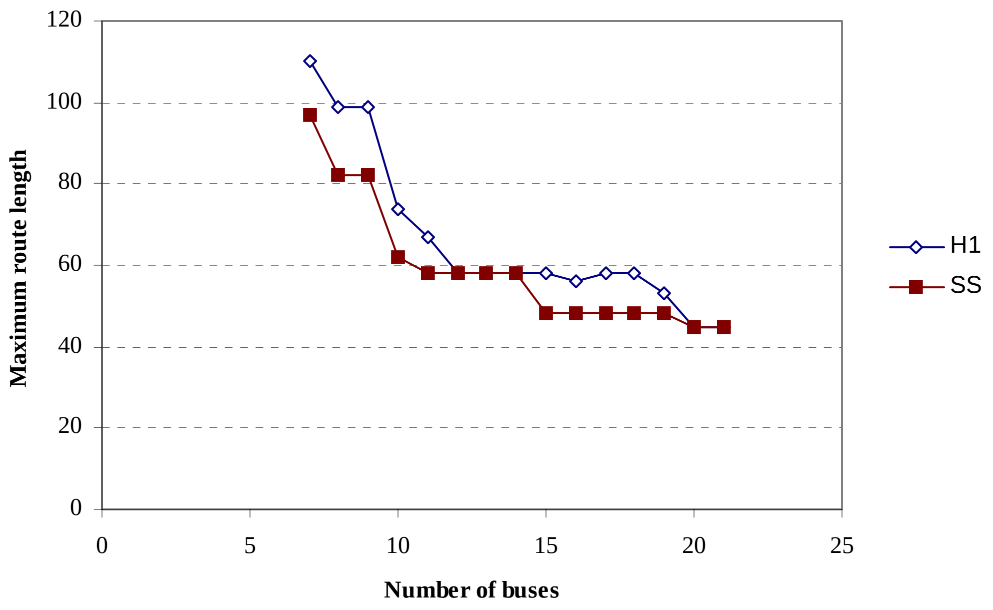

Figure 4. Approximation of the efficient frontier with H1 and after SS is applied.

In our next and final experiment, we compare the solutions generated by our procedure with the solutions currently implemented in practice in the 16 middle schools. (We refer to these problems as S1 to S16.) In addition, we compare our solutions with those generated by a tabu search implementation that seeks to minimize $t_{\max}$ while keeping the number of buses fixed (Delgado and Pacheco, 2000). The tabu search approach was designed to minimize $t_{\max}$ while keeping the total routing cost "close" to the cost of the solution currently used. The cost function considers the distance traveled, the number of students in the bus and the number of pickup points.

Table 3 shows the results of this experiment. The first column shows the problem number. The second and third columns respectively show the number of buses and the maximum route length (in minutes) corresponding to each problem in the current solution (i.e., the solution currently implemented in practice). The fourth and fifth columns respectively show the number of buses and the maximum route length (in minutes) for each problem as generated with the tabu search procedure in Delgado and Pacheco (2000). The columns labeled *m*-1 to *m*+3 show a partial segment of the approximation of the efficient frontier generated with our scatter search procedure. The *m*-value corresponds to the number of buses in the current solution. This means that for problem S1, the approximate efficient frontier is given for a number of buses ranging from 11 to 15. To compare the maximum route length between the current solution and the scatter search solution, one must compare the values in the column labeled *m* in the scatter search section with the column labeled "Buses" in the current solution section. Note that the scatter search solutions consistently improve upon the current solution, with an average improvement of 23.4%, a maximum of 40% and a minimum of 12%.

The actual transportation problem has a policy in which no route should be longer than 60 minutes. However, the current solutions violate this policy in the middle schools S1, S4, S6, S8 and S9, with the maximum violation of 30 minutes in S9. The scatter search finds routes that comply with the policy in all but 2 schools (S6 and S9), for which the violations are 2 and 8 minutes, respectively. In some situation such as schools S1, S3, S5 and S7, scatter search finds routes that comply with the policy with one less bus than the current solution.

Table 3. Comparison of results for 16 middle school problems.

	Current Solution		Tabu Search		Scatter Search				
	Buses	Route		Route					
Problem	(*m*)	Length	Buses	Length	*m*-1	*m*	*m*+1	*m*+2	*m*+3
S1	12	70	14	48	57	56	53	51	48
S2	5	45	8	29	46	36	32	29	
S3	6	60	7	45	54	46	43	39	
S4	3	70	5	36		52	42	37	33
S5	4	60	4	43	55	44	39		
S6	4	80	6	45	81	62	53	47	44
S7	6	60	7	37	51	45	38	36	
S8	9	75	11	51	61	59	50	47	44
S9	5	90	7	52	82	65	53	50	48
S10	6	60	6	41	44	41	40		
S11	4	60	5	45	67	51	45	39	
S12	2	25	4	9		15	14	9	
S13	6	45	7	29	39	34	32	29	
S14	5	60	5	46	53	46	37		
S15	7	50	8	40	50	44	42	40	
S16	2	60	3	38	84	51	38	35	

We point out that in some problems, the tabu search procedure finds solutions that dominate those found with our scatter search. For example, in problem S4, that tabu search solution has a $t_{\max}$ of 36 minutes with 5 buses while scatter search yields a $t_{\max}$ of 37 minutes with the same number of buses (see entry in row S4 and column *m*+2). This situation occurs in 7 out of the 16 problems, however, the difference is never more than 3 minutes. Also note that the empty entries in the last five columns of the table mean that either scatter search was not able to find a solution with the specified number of buses or the solution it found was dominated. In terms of computational time, our scatter search procedure averages 4.8 seconds, which we believe is at least an order of magnitude less than the times required by the tabu search implementation, although we cannot confirm this given the information in Delgado and Pacheco (2000). The solution times are not critical in this context, because the routes do not change in real time.

5. Conclusions

The research described in this paper was motivated by a bus routing problem in a sparse rural area. The nature of the underlying optimization problem was bi-objective. We chose to develop a method capable of dealing with both objectives simultaneously, instead of creating a single objective function as a weighted combination of the two individual objectives. The procedure is based on the scatter search framework and uses two constructive heuristics to generate solutions for the initial reference set. The novel features of our procedure include the constructive heuristics as well as the combination method, which is based on a voting scheme. Our computational testing reveals the ability of our procedure to approximate the efficient frontier for each routing problem. Decision makers may use efficient solutions to estimate the best service level (given by the maximum route length) that can be obtained with each level of investment (given by the number of buses used). Our results show that several of the solutions implemented in practice are not efficient. The main contribution of our research is that our procedure gives the decision maker a set of solutions from which he/she can choose the best compromise between cost and quality of service.

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.

Delgado, C. and J. Pacheco (2000) “Problemas de Rutas Minmax: Aplicación al Transporte Escolar de Burgos,” Departamento de Economía Aplicada, Universidad de Burgos.

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., 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. (2000) “Scatter Search,” to appear in *Handbook of Applied Optimization,* P. M. Pardalos and M. G. C. Resende (Eds.), Oxford Academic Press.