Multiobjective metaheuristics for the bus-driver scheduling problem

Lourenço, H.R., Paixão, J.P. and Portugal, R. (2001), Multiobjective metaheuristics for the bus-driver scheduling problem. Transportation Science 35(3): 331-343.

ISSN: 0041-1655.

Link to publication

Multiobjective Metaheuristics

for

The Bus-Driver Scheduling Problem

Helena R. Lourenço
Grup de Recerca en Logistica Empresarial
DEE, Universitat Pompeu Fabra
R. Trias Fargas 25-27
08005 Barcelona, Spain
helena.ramalhinho@econ.upf.es

José P. Paixão
DEIO, Faculdade de Ciências
Universidade de Lisboa, Portugal
jpaixao@fc.ul.pt

Rita Portugal
ICAT - FCUL
Universidade de Lisboa, Portugal
ritamp@icat.fc.ul.pt

We present new multiobjective metaheuristics for solving real-world crew scheduling problems in public bus transport companies. Since the crews of these companies are drivers, we will designate the problem of the bus driver scheduling. Crew-scheduling problems are well known and several mathematical programming based techniques have been proposed to solve them, in particular, using the single objective set-covering formulation. However, in practice, there exists the need to consider multiple objectives, some of them in conflict between themselves; as for example, the cost and service quality, implying also that alternative solution methods have to be developed. We propose multiobjective metaheuristics based on the tabu search and genetic algorithms. These metaheuristics also present some innovation features related with the structure of the crew scheduling problem that guide the search efficiently and enable them to find good solutions. Some of these new features can also be applied into the development of heuristics to other combinatorial optimization problems. A summary of computational results with real-data problems is presented. The methods have been successfully incorporated in the GIST Planning Transportation Systems and are actually used by several companies.

Public transportation companies are faced with important challenges in the area of transportation planning due mainly to population growth, environmental policies, requirements for service with quality, and the pressure from governments for better use of their resources. Therefore, transportation planning systems in public transport have been gaining importance since a large amount of money can be saved if the available resources are employed efficiently, or wasted if not. The same challenges are faced by the private transportation companies. As a consequence, there is an increasing need for computerized tools to aid planners in public and private companies.

Several projects have been developed, or are undergoing development, to design a Transportation Planning System, as for example HASTUS, Rousseau et al.(1985), IMPACTS, Smith and Wren (1988), HOT, Daduna and Mojsilovic (1988) and TRACS II, Kwan, Kwan, Parker and Wren (1997). All these systems are used by transportation companies in several countries. This work is also a part of a large project in transportation planning, designated by GIST, developed in INEGI and ICAT, in coordination with six bus transportation companies: Carris, STCP, Horarios do Funchal, Vimeca, Barraqueiro and Rodoviaria. GIST is a Decision Support System to assist the planning department of public and private transportation companies or transit authorities in the operations management. The system includes the production of timetables, the scheduling of vehicles, the generation of daily duties for drivers, and the construction of duty rosters of individual drivers for a certain period.

In general, the transportation planning process is decomposed into several subproblems due to its complexity: timetabling, vehicle scheduling, crew scheduling (in this case driver scheduling) and duty rostering, with relations between them as it can be seen in Figure 1, Freling (1997).

The bus network structure includes all the information on the company operational network, well known information which usually does not change for large periods of time. The transport service is composed by a set of bus lines, usually identified by a number, that correspond to a bus traveling between two points in town or between two towns. For each bus line the respective frequency is determined based on demand. Afterwards, a timetable is constructed, resulting in trips that correspond to a start and to an end point as well as a start and end time. The vehicle scheduling assigns vehicles to trips. The driver scheduling problem generates daily duties for drivers. The duty rostering process is a long term driver planning, e.g. a month or half year, for constructing rosters from the drivers duties. For a survey in transportation planning see Wren and Rousseau (1995), Wren (1996), Odoni et al.(1994), Daduna et al.(1995), and for a recent survey in vehicle and crew scheduling, see Freling (1997).

Traditionally, the crew scheduling problem (and more specifically regarding this paper the driver scheduling problem) has been formulated as a single objective linear program with several Linear Programming methods proposed to solve it. These methods have been widely applied by the previously mentioned systems. The DSS GIST used, until recently, an algorithm based on Linear Programming, Beasley (1987), and the Vasko and Wolf (1988) and Beasley (1987) heuristics to obtain lower and upper bounds for the bus-driver scheduling problem, formulated as a Set Covering model. However, the bus companies involved in the GIST project were not satisfied with the bus-driver module based on single-objective LP, even if it obtained the optimal cost solution. The transportation companies may have several different objectives when planning besides the cost function, as for example the service quality, which can be measured in different ways in each company. We asked the companies to give us a good solution for them, finding the cost to be high when we plugged in it into the single objective model. To be able to produce a more realistic model of the problem we have to consider several objective functions linked to the service quality that usually are in conflict with the traditional cost function.

In this work, we put emphasis in taking into account the environment of the users of the GIST system when developing the solution methods. In everyday planning, these companies need fast methods to obtain several good scenarios that can help the decision maker. Therefore, the aim of this work is to develop methods to solve real-world crew scheduling problems that can be used in a transportation planning system for these companies, in a user-friendly environment. The methodology followed to develop such a solution approach was the multiobjective metaheuristics since they can obtain efficient solutions in a short time considering the multiobjetive approach, allowing, after the decision maker to do what he or she does best, making judgments in the face of the set of several good scenarios. The ultimate objective is improvement of real managerial and operations effectiveness.

Our main applications are for bus transportation companies, where all our data came from considering the crew of the buses are drivers, we will designate the problem as the Bus-Driver Scheduling Problem (**BDSP**). As well, the metaheuristics presented here can be applied to other crew scheduling problems in different sectors, such as, train and airline company problems and, in general, to other combinatorial optimization problems.

In the next chapter, we present more details of the bus-driver scheduling problem and the multiobjective combinatorial optimization model for the problem. In chapter 2, we propose a GRASP, multiobjective Tabu Search and multiobjective Genetic Algorithm, respectively, to solve the BDSP. In chapter 3, a summary of the computational experiment is presented followed by the conclusions and description of future work.

1. The bus-driver scheduling problem

The bus-driver scheduling problem can be stated as finding a set of feasible daily duties that cover all trips or vehicle blocks. A vehicle block is the itinerary of a vehicle between its departure from the garage and its return to the same garage. Any vehicle block can be split into pieces of work, such that a split occurs only at a relief point, i.e. a time and a place at which change of drivers is possible. A driver’s duty is a set of pieces of work that can be assigned to a driver.

Several formulations have been proposed for the single objective crew-scheduling problem. We will consider an approach based on the set covering formulation of the problem, considering multiple objective functions. One of the advantages of the set covering formulation is that it is independent of labor contract and specific company rules. Therefore, the duty generation module is separated from the duty selection module where the minimal cost or best quality driver duties are chosen such that all pieces of work have a driver. In this case, the set of the driver duties is generated by an algorithm presented in Agra (1993). The method iteratively generates combinations of pieces of work that complies with labor contracts and company rules, specific for each company, as for example no duties with more than four hours lunch break. This procedure is tailored to each company since each company has its own rules mostly based on union agreements and operational rules. The separation between duty generation process and duties selection allows us to adjust only the first module for each transportation company and therefore, is very convenient for implementation reasons.

Let $M = \{1,2,\ldots,m\}$ and $N = \{1,2,\ldots,n\}$ be the index sets for the pieces of work (rows) and feasible duties (columns), respectively. Define the matrix A as follows:

$$a_{ij} = \begin{cases} 1 & \text{if the duty index } j \text{ includes the } i\text{-th piece} \\ 0 & \text{otherwise.} \end{cases}$$

Consider the following variables:

$$x_j = \begin{cases} 1 & \text{if } j\text{-th duty is in the solution;} \\ 0 & \text{otherwise.} \end{cases}$$

For each duty, a cost is associated which represents real cost as extra hours, night hours and meal costs, and artificial costs, such as, vehicle change, type of service, and number of hours over the average. Other components of the cost can be considered if they are requested by the transportation company. Let c_j be the cost associated with duty j and define the cost function as $f_1(x) = \sum_{j=1}^{n} c_j x_j$.

In the previous formulations the objective was to minimize the total cost of the driver's duties. However, the single objective approach had been object of criticism by the bus companies in the GIST project since during planning they also want to take into account other objectives like service quality. Different companies may have different measures of the service quality, but all of them agree in the importance of taking other objectives besides cost when planning. Moreover, they want to be able to analyze several scenarios before taking a decision. Therefore, a multiobjective combinatorial optimization model is appropriate to formulate the BDSP. Some examples of measures of the quality of the service are:

- The number of pieces of work not covered. Some companies allow pieces of work or trips to be uncovered. However, a good schedule should have a small number of these pieces. The objective function is $f_2(x) = \sum_{i=1}^{m} max\left\{1 - \sum_{j=1}^{n} a_{ij} x_j , 0\right\}$.
- The unfitness value, which measures the amount of infeasibility with respect to the set partitioning formulation, Chu and Beasley (1995). We define the unfitness by $\sum_{i=1}^{m} \left| w_i - 1 \right|$, where $w_i = \sum_{j=1}^{n} a_{ij} x_j$ as the number of columns in the current solution x that cover row i. In practice, this value is quite important for the users. Some degree of overcovering is desired. However, if a solution has some pieces of work with too many drivers assigned, the planner will have to adjust manually this solution until he or she obtains one with smaller unfitness value. Therefore, we have defined the unfitness function as $f_3(x) = \sum_{i=1}^{m} \left| \sum_{j=1}^{n} a_{ij} x_j - 1 \right|$.
- The total number of duties. Some companies claim that a small number of duties respecting the labor and company constraints allow a better planning and easier implementation. Define the “number of duties” function as $f_4(x) = \sum_{j=1}^{n} x_j$.
- The total number of duties with only one piece of work (single piece-of-work duty). If we allow duties with only one piece of work, these duties will be very expensive in terms of

labor cost. Therefore, some companies use them only in special cases and want to minimize the use of these type of duties. The objective function is $f_5(x) = \sum_{\{j:\sum_i a_{ij}=1\}} x_j$.

- The number of vehicle changes. The change of a vehicle driver can disrupt the operations of the company and cause complaints from the drivers. Therefore, some companies are mostly worried about minimizing the number of changes. Supposing the columns are ordered such that the last $\bar{n}$ are the duties with vehicle changes, the objective function is $f_6(x) = \sum_{j=n-\bar{n}+1}^{n} x_j$.

Other objective functions can be considered as well as some combination of any of the above functions.

The BDSP can be formulated as a multiobjective set-covering problem, where K is the number of objective functions to be considered:

$$\text{Min} \quad z(x) = \left(f_1(x), f_2(x), \ldots, f_K(x)\right) \qquad (1)$$

$$\text{s.t.} \quad \sum_{j=1}^{n} a_{ij} x_j \geq 1, \quad i \in M, \qquad (2)$$

$$x_j \in \{0,1\}, \quad j \in N. \qquad (3)$$

Columns correspond to duties and the rows correspond to the pieces of work. We say that row *i* is covered if there exists a column *j* in the solution such that $a_{ij} = 1$. Constraint (2) means that every piece of work has to be covered by at least one duty. This means that there exists a driver's duty *j* in the solution that contains the piece of work *i*.

The single-objective set-covering problem (SCP) is NP-hard, Karp (1972), Garey and Johnson (1979). Several approaches have been proposed to solve the SCP, based on heuristics, column generation, lagrangian and linear programming relaxations and state space relaxation, see for example Caprara, Fischetti and Toth (1999) as one of the last proposed heuristics for the SCP. Other surveys can be found in Odoni et al.(1994), Freling (1997) and Daduna et al.(1995). Several greedy heuristics, based on different priority or greedy functions presented in Vasko and Wolf (1988). Beasley and Chu (1996), Al-Sultan et al.(1996), Clement and Wren (1993). Wren and Wren (1995), Kwan and Wren (1997), Kwan, Kwan and Wren (1997), Portugal (1998), and Galvão, Sousa and Cunha (1998) have proposed some approaches based on genetic algorithms (GA).

Multiobjective optimization has been vastly studied, see Steuer (1986) for a basic reference. One property that is commonly considered as necessary for any candidate solution to the multiobjective optimization problem is that the solution is not dominated.

A feasible solution x is dominated by a second feasible solution y if and only if $f_k(y) \leq f_k(x)$ for $k = 1, \ldots K$, and $f_k(y) < f_k(x)$ for at least one k. A feasible solution is efficient if, and only if, there is no other feasible solution which dominates it.

In classical literature on muldimensional optimization problem, the usual approaches are to aggregate all the objectives in a utility function and solve the problem as a single objective problem or use interactive method, see Steuer (1986). However, knowing the set of efficient (Pareto-optimal) solutions, or an approximation of it, gives greater freedom to the decision maker when selecting solutions. However, finding this set is a complex task. If several efficient solutions are known by the user, this one can compare them even with respect to additional criteria which have not been formalized, e.g. the acceptance by management or the public. Note that this is exactly the objective that the transportation companies involved in the GIST project are looking for. Moreover, since we are working with different companies, the criteria to choose one solution over another can be different amongst the various companies. The possibility of analyzing several scenarios of efficient solutions is a very important aspect for the companies in order to accept the transportation planning system. For all these reasons, some of the classical methods are not adequate. The recent advances and success in multiobjective metaheuristics applied to other problems advocate the use of these techniques as a good way to proceed.

2. Multiobjective Metaheuristics for the BDSP

Metaheuristics have been object to extensive studies and applied with success to several single-objective optimization problems, including the crew-scheduling problem, see Clement and Wren (1993), Wren and Wren (1995), Kwan and Wren (1997), Kwan, Kwan and Wren (1997) and Galvão, Sousa and Cunha (1998). For a large list of applications and recent developments see the two volumes published as proceedings of the International Metaheuristics Conferences, Osman and Kelly (1996) and Voss et al.(1998), and see Osman and Laporte (1996) for a list of publications. Only recently there has been an increasing interest in applying metaheuristics to multiobjective problems, see Viana and Sousa (1999), Hansen (1997) and Fonseca and Fleming (1995). The aim of multiobjective metaheuristics is to obtain a good approximation of the set of efficient solutions. Due to the computational efficiency of these methods, we propose a multiobjective tabu search and multiobjective genetic algorithm to solve the BDSP.

In the next sections, we will present several metaheuristics for the BDSP. We will start by describing a single objective GRASP procedure and some common aspects of the proposed metaheuristics, like the candidate list and penalization function for infeasible solutions. The aim of the GRASP procedure is not to be used as a solution method to solve the BDSP, but as a procedure inside the multiobjective tabu search and genetic algorithm described in the following sections. It is presented first because of its simplicity.

2.1 GRASP

The Greedy Randomized Adaptive Search Procedure, GRASP, was proposed by Feo and Resende (1989), and since then it has been applied to several Combinatorial Optimization problems with success.

The basic idea of GRASP is to combine constructive methods with local search. GRASP can be seen has a multi-start sampling algorithm, rather than starting from purely random solutions, constructing the initial solution using a greedy adaptive probabilistic heuristic. In the first phase, the construction phase, a feasible solution is built in a greedy fashion, introducing one element at each step. At each step of the greedy heuristic a restricted list of the best elements to be included in the solution is created, follow by the random selection of one of these elements that is inserted into the solution. The process is repeated until a feasible solution is found. In the second and last phase, a local search method is applied to try to improve the solution found in the first phase. Both phases are repeated until a certain stopping-criterion is verified.

The first phase of a GRASP for the set covering problem corresponds to a greedy adaptive randomized heuristic, where at each iteration a column is added to the solution; we keep adding columns until all rows have been covered. The greedy function used is the following one: c_j / k_j where c_j is the cost of column *j* and k_j is the number of uncovered rows that column *j* covers if added to the solution. The greedy heuristic is adaptive since at the end of each iteration the value of k_j is adapted taking into account the columns already included in the solution.

The second phase of the GRASP is a first-improvement local search heuristic, where the objective function is $\bar{z}(x) = f_k(x) + g(x)$, where $f_k(x)$ is one of the objective functions, and $g(x)$ is a penalty function, defined next. The most important concept in a local search is the definition of the neighborhood for the problem in consideration. We propose an exchange neighborhood, i.e. remove a column of the solution and add a new column that covers at least one of the uncovered rows. It is important to notice that, for this neighborhood, the number of columns of the neighbor solution is always equal to the number of columns of the initial solution obtained in the construction phase of GRASP. However, since both phases are repeated several times, the search visits solutions with different number of columns.

More precisely, let *x* be a solution of the BDSP, and let N_x^* be the set of columns in the solution *x*. Then, a move is defined by removing a column *l* from the current solution *x* and a new column *k*, not in *x*, is added such that it covers at least a row left uncovered by removing *l*. The neighborhood can be formally defined as follows:

$$N(x)=\{y: \exists\ l,k \in N, \text{such that } l \in N_x^*\ (x_l = 1) \text{ and } l \notin N_y^*\ (y_l = 0) \text{ and } k \notin N_x^*\ (x_k = 0)$$
$$\text{and } k \in N_y^*\ (y_k = 1),\ \text{also} \exists\ i \in M\ \text{(row)}: \sum_j a_{ij} x_j \leq 1 \text{ and } a_{il} x_l = 1, \text{and } \sum_j a_{ij} y_j \geq 1\}$$

This exchange neighborhood can lead to infeasible solutions, i.e. some rows are not covered since we only require that the entering column covers at least one uncovered row, even if the leaving column can leave one or more rows uncovered. Allowing these extra (infeasible) solutions is known as strategic oscillation, Glover and Laguna (1997), and provides escapes from local optimal solutions and drives the search to good solutions. To control the trajectories and avoid visiting too many infeasible solutions, we define a penalty function that penalizes the infeasibility and is defined as follows: $g(x) = -K \times \sum_{i=1}^{m} \min(0, w_i - 1)$ where $w_i = \sum_{j=1}^{n} a_{ij} x_j$, and *K>0* is a parameter representing the "cost" of an uncovered row. This penalty function is added to the objective function considered in the GRASP procedure.

Another aspect of the local search is related to the candidate list of neighbors. The feasible solutions, for most of our real problems, contain a very small ratio between the number of columns in the solution and the total number of columns. Therefore, the number of candidate columns to enter is quite large. To avoid considering all possible columns, we start with the ones of smaller penalized cost, defined next. Considering a current solution *x*, after removing the column *l*, the penalized cost is obtained by taking into account the uncovered rows and overcovered rows that a column covers after entering the solution. The penalized cost of column j, not in the solution, is defined as follows: $p_j = c_j + \mathrm{U} \times \sum_{i=1}^{m} u_{ij} + Q \times \sum_{i=1}^{m} q_{ij}$, where $u_{ij} = a_{ij} * \min(0, w_i - a_{il} - 1)$, $q_{ij} = a_{ij} * \max(0, w_i - a_{il})$, *U>0* and *Q>0* are parameters associated with rows uncovered and overcovered, respectively, and w_i defined as above. If a column covers many uncovered rows the cost is reduced. On the other hand, if a column covers many rows, already covered, the cost increases.

As mentioned, the reasons for using such a simple local search and neighborhood are due to the aim of repeating several times the GRASP iteration, each time starting with a different initial solution. The application of the GRASP is intended as a subroutine of the tabu search and genetic algorithm as described next.

2.2. Tabu search algorithm for the BDSP

Tabu Search is an adaptive procedure originally proposed by Glover (1986). Recently, this metaheuristic has been gaining importance as a very good search strategy method to solve combinatorial optimization methods. For a survey see Glover and Laguna (1997).

The basic idea of tabu search is to escape from a local optimum by means of memory structures. Each neighbor solution is characterized by a move and short term memory is used to memorize the attributes of the most recently applied moves, incorporated via one or more tabu list. Therefore, some moves are classified as tabu and consecutively some neighbor solutions are not considered. To avoid not visiting a good solution, an aspiration criterion can be considered. At each iteration, we choose the best neighbor of the current solution that it is not tabu or verifies an aspiration criterion. The aspiration criterion used here was the most common one; the tabu status is overruled if the neighbor solution has an objective function value smaller than the best found up to that iteration. The algorithm stops when a certain stopping-criterion is verified. The best solution found during the search is then the output of the method.

Initially, we will present the basic features of the tabu search algorithm for the bus driver scheduling problem; afterwards we will describe some new aspects of the heuristic.

The main steps of the tabu search algorithm are:

1. Obtain an initial solution *x*.
2. While a certain stopping-criterion is not verified:
   2.1. Obtain a neighbor of *x*, *x'*, not tabu or satisfying an aspiration criteria with minimal value of the objective function (or weighted sum function, in case of multiobjective) among the neighbors of *x;*
   2.2. Set *x*=*x'* and update the tabu list and aspiration criteria.
3. Return the best solution found.

The initial solution can be obtained by two methods: a random initial heuristic and a greedy heuristic. The random initial works as follows: For each row, randomly select a column between the ones that cover it. When all the rows have been considered, the redundant columns are removed using a greedy type algorithm. The greedy heuristic builds a solution in a greedy fashion, at each step, selecting a column to enter the solution following some greedy function; repeat this step until all rows have been covered. This is the deterministic version of the greedy adaptive probabilistic heuristic of the previous section.

We considered three neighborhoods, the exchange neighborhood, presented before, the remove neighborhood and the insert neighborhood. The insert neighborhood considers all solutions that can be obtained from the current solution by the introduction of one column. In the opposite way, the remove neighborhood considers all solutions that are obtained from the current one by removing one column. Most of the solutions consist of a small number of columns. Therefore, the insert neighborhood is larger than the remove neighborhood. Due to the size of the insert neighborhood, we consider a candidate list strategy. The strategy makes the search more aggressive and avoids visiting solutions that are not attractive. The candidate list is built considering the following rule: a column is considered for entering if it covers one uncovered and if its penalized cost, p_j, is less or equal than the average cost.

For each of the three neighborhoods a certain number of iterations are performed which depend on a parameter related to the size of the neighborhood. The order of the neighborhood search is as follows: insert neighborhood, exchange neighborhood, remove neighborhood, exchange neighborhood, and repeat.

Multiobjective Tabu Search usually considers a set of weights assigned to each objective function and a utility function that is the weighted sum function, see Viana and Sousa (1999) and Hansen (1997). In our implementation, we apply initially a tabu search considering one objective function at each time and try to obtain the best solution with respect to this objective function. Afterwards, we apply the tabu search using the weighted sum function to obtain additional efficient solutions. In the diversification steps, the weights are modified in such a way that the search tries to look for new nondominated solutions by using the information obtained in previous runs and the values of the objective functions for each individual run at first step. All nondominated solutions found are stored and output at the end of the search.

Two tabu lists were considered: the insert tabu list and the remove tabu list. The remove tabu list contains all columns that have been inserted recently in the solution and therefore they cannot be removed. The insert tabu list contains the columns that have been removed in the most recent iterations and therefore, it is tabu to insert them again in the solution. The two tabu lists have different size because the candidate columns for each one are quite different. The size of the insert tabu list is a percentage of number of columns in the first solution, and smaller than the size of the remove list, which is defined by the user as a percentage of the total number of columns.

Next we present an innovation aspect considered in the tabu search, the optimized intensification strategy. The inclusion of this new feature has the objective of improving the search and can be seen as intensification strategies based on classical optimization and hybrid methods.

The optimized intensification strategy consists in applying the tabu search for several iterations using only the insert neighborhood. The resulting solution will have a large number of columns and each row will be covered by several columns. To obtain good solutions with fewer columns and such that each row is not overcovered, we can apply an exact method to the set covering subproblem using any objective function. Note that the subproblem has a smaller dimension compared with the full problem since we are only considering the columns in the current solution. For small instances, we can apply the exact method using the cost function. For larger instances, we apply the GRASP method described previously since the exact method takes too much time. This permits us to find good solutions, eliminating the most expensive columns, which are inserted in the insert tabu list. Moreover, this intensification strategy allows us to obtain solutions that would be difficult to find by the usual search, and the computational time did not increase significantly. The optimized intensification strategies combined with tabu search have been applied with success to several scheduling problems, Lourenço (1995).

2.3. Genetic algorithm for the BDSP

Genetic Algorithms (GA) were originally developed by Holland (1975) and are intelligent search heuristics based on evolution. The basic idea of the genetic algorithms is that, during the course of evolution, the best fitted individuals have better chance to survive and reproduce, meanwhile the least fit individuals will be eliminated. A GA simulates this behavior by taking into account an initial population of solutions (individuals) and a fitness function usually associated with an objective function. By means of some selection techniques and operators, this population is replaced by a new one with higher fitness. This cycle is repeated until a satisfactory solution is found. The population-based search of the GA makes this approach tailored to solve multiobjective optimization methods. See Fonseca and Fleming (1995) for a discussion in the use of genetic algorithms to multiobjective problems.

GA have been applied to a wide range of problems in several areas. For a survey on GA see Davis (1996). The GA proposed to solve the BDSP is based on the work of Beasley and Chu (1996), but considering the multiobjective aspects of the problem.

The fundamental aspects of genetic algorithms are: the representation of the solutions, parent selection, population replacement and the genetic operators, crossover and mutation.

The main steps of the genetic algorithm are:

1. Generate a family of initial solutions.
2. Calculate the fitness of each solution.

3. While a certain stopping-criterion is not verified:
   3.1. Select elements from the population
   3.2. Crossover these elements
   3.3. Mutate some elements
   3.4. Population replacement
4. Return the best solution found.

Solutions are represented as a binary vector of dimension *n*, indicating if the column (driver duty) is or is not in the solution. This is the obvious representation of a covering solution, but others have been suggested, see Beasley and Chu (1996). The major problem of this representation is that the application of a crossover or a mutation operator frequently produces infeasible solutions. To control the infeasibility, we use the penalty function described previously. Also, if necessary, a greedy heuristic can be applied to restore feasibility.

The initial population is generated by different methods to guarantee some diversification. The first method generates most of the population, except for ten solutions, and can be described as follows: for each row, choose randomly a column that covers it, and then apply a simple heuristic to eliminate columns that cover already covered rows. The remaining ten solutions are obtained by the heuristics described in Vasko and Wolf (1985), so we guarantee the presence of some good solutions in the initial population in order to make the search converge faster. The initial population has 100 solutions and, as the offsprings are introduced into the population, it can grow until 200. When this limit is obtained, we choose the 100 best solutions and eliminate the remaining ones. The final population will be an approximation of the set of efficient solutions since the nondominated solutions within a population are maintained from generation to generation. At the end, the system presents these "near" nondominated solutions to the users to be considered in the planning process.

The parents are selected by a tournament selection based on the uniform probability function and on the objective functions. In this method, two groups of T solutions are selected uniformly from the population and the best solution of each group is selected as a parent. Beasley and Chu (1996) state that this method is one of the more efficient methods and suggest the value 2 for T. In the multiobjective approach different objective functions are used to determine the best solution of each group. To diversify the search, at some stages, we randomly choose one of the objective functions. The reason for using this approach is that the main objective is to determine an approximation of the set of efficient solutions.

The crossover operator defines the way of combining two or more parent solutions to an offspring solution. Several methods have been considered such as the one-point and two-point crossover, the uniform crossover and the generalized fitness-based crossover or fusion operator, see Beasley and Chu (1996). We consider the two-point crossover where two crossover points are selected randomly and the segments of the parents are swapped to produce two offspring solutions.

An improvement in the GA was obtained by defining a new crossover operator that we denominate by *perfect offspring*. This operator considers two parents and tries to obtain the best offspring of these parents by solving a set-covering sub-problem, where all the rows are considered but only the columns present in the parent solutions are taken into account. We follow the optimized intensification strategy described in the previous section, applying exact methods for small instances and the GRASP method for larger ones. This approach follows the methodology known as memetic algorithms, Moscato (1989) and a similar approach based on Integer Linear Programming was used by Clement and Wren (1993). Also, this solution combination method can be found in scatter search, Glover and Laguna (1997), optimized search heuristics, Lourenço (2001) and in optimized crossover of Aggarwal et al.(1997). Our GA, not only combines aspects of different search methods, but also optimization methods based on the structure of the problem. The perfect offspring crossover allows the search to converge rapidly to efficient solutions and this idea can be applied to other combinatorial optimization problems.

The mutation operator permits to introduce random variations in the solutions and plays an important role in the capacity of the GA to diversify the search, especially when all the solutions in the population become similar. A mutation is applied to each offspring solution after crossover. We use the mutation operator proposed by Beasley and Chu (1996) based on removing or including a column.

After the crossover and the mutation, an offspring solution can be included in the population or not, depending on the dominance criteria. Iteratively, compare the child solution with the solutions in the population starting at a random position. Three situations can occur:

1. A solution having better values for all objective functions is found. Discard this child solution and go back and obtain a new one (the child solution is dominated by another solution in the population).
2. A solution in the population having worst values for all objective functions is found. The child solution dominates this solution and so, substitutes this last one in the population.

3. If none of the above happens, then the child solution is not dominated by any solution in the population, so add the child solution to the population without removing any solution.

The just added offspring can be a parent in the next iteration.

3. Summary of the computational results

The motivation for this research was to develop solution methods to solve real-world driver scheduling problems that can be used in a transportation planning system in a user-friendly environment. Therefore, the computational experiment was designed to analyze the performance of the different metaheuristics described previously when applied to real instances. Our main objectives are to gain understanding of the behavior of the different methods and compare the performance of these ones with the method actually used by GIST. We present a summary of the extensive computational testing reported in Portugal (1998), where the characteristics of bus-driver problems of each of the transportation companies with all of the GIST Project results are presented.

All numerical tests were carried out on a Workstation IBM Risc 6000. The algorithms were coded in C. The real data sets were obtained directly from the companies associated with the GIST project, see Table 1. The density of matrix A is between 2% and 6%.

In Tables 2 to 7, we present a summary of the solutions chosen by the companies in their final process, where LP stands for the linear programming based algorithm previously used by the GIST system which could obtain the optimal solution for almost all the instances except a very few large ones. We also present the efficient solutions, ordered by the cost, obtained by one run of the multiobjective tabu search (TS) and two versions of the multiobjective genetic algorithms, a simple GA with a two point crossover (GA2) and a sophisticated genetic algorithm with the perfect offspring crossover (GAS). For each test problem, we present the value of the following objective functions for each efficient solution: cost function, unfitness, total number of duties, the number of single piece-of-work duties, and the time in seconds of one run.

The usual practice before the introduction of the multiobjective metaheuristics in the GIST system was to run the LP-based algorithm several times with different objective functions to obtain several scenarios. The times, indicated in the tables, for LP approach algorithm is the average time of one run to obtain one scenario, whereas, one run of the multiobjective tabu search and genetic algorithms obtains several efficient solutions.

The tabu search outperformed the remaining methods since overall it obtained the best results for all the evaluation functions. However, a few final solutions where infeasible and it was necessary to apply a heuristic to restore feasibility.

Comparing the LP-based solutions with the TS-solutions, we can observe that the TS-solutions have better values with respect to other functions besides cost; meanwhile, the cost is a little higher, as expected. On average, the running times are similar for both methods, but the tabu search algorithm may obtain several efficient solutions. Showing solutions to the final users, they commented that the TS-solutions have the characteristics they are looking for: less number of single piece-of-work duties, smaller unfitness, less or approximately the same number of services and similar cost.

Considering the GA and the LP-based solutions, we can observe that if the GA runs for a short time we can obtain similar solutions to the LP ones (one run). The GA-solutions have small values for the number of single piece-of-work duties and smaller unfitness value, approximately the same number of services, but higher cost. For larger running times, the results improve for other objective functions.

Between the two versions of the genetic algorithms, we can observe that the perfect offspring crossover version (GAS) obtains slightly better results but at the expensive high computational running times. Therefore, more work is needed to obtain a faster perfect offspring crossover.

Users of different companies were asked to evaluate the performance of the different methods. They point out the importance of the consideration of several objective functions, in addition to the cost function, and the advantage of being able to obtain several scenarios as the main benefit and utility of the new multiobjective metaheuristics. As a consequence of having good values for the unfitness and single piece-of-work duties, the need for manual adjustments that yield satisfactory solutions is reduced significantly, leading to a better acceptance of the GIST system.

4. Conclusions and future work

In this paper we have presented a multiobjective tabu search and multiobjective genetic algorithms for solving the bus driver scheduling problem in transportation companies, formulated as a multiobjective set covering problem. These methods have been successfully incorporated in the Decision Support System for Transportation Planning GIST, allowing solving very large-scale problems during the planning process, substituting the previous LP-based method that was criticized by the users. The LP method was able to find the optimal solutions with respect to the cost function for the large majority of the instances. However, the solutions usually had some characteristics that were difficult to implement, or were not accepted by the management. The use of the multiobjective metaheuristics lead to better acceptance by the users of the bus-driver module of the GIST system, creating real schedules that meet the requirements of the final users. Moreover, the GIST system, using these new methods to build bus driver schedules, permits the achievement of a final solution that does not need manual adjustment by the user, a common practice for the LP-based solutions.

An additional advantage of the multiobjective metaheuristics is the possibility to incorporate in an easy way the use of different objective functions. The minimization of the changes necessary for adjusting the objective functions is very important in the design of the GIST system.

Our computational testing showed that the Multiobjective Tabu Search and the Multiobjective Genetic Algorithms lead to good results within reasonable times, as well as, results that compare favorably with the LP-based solutions. The users can now choose between any of the three methods and obtain several scenarios for a final decision in the planning process.

The system GIST can be used for operational and planning decisions, helping with every day planning or as a strategic tool. During the negotiations of union contracts or changes in transportation regulation, users can successfully manage the resources gaining a sustainable competitive advantage. A similar approach is described in Campbell et al.(1997) for the airline industry.

As future work, we would like to consider more structural elements of the problems in the metaheuristics. Since the set covering problem has been very well studied, the use of this knowledge in the design of a metaheuristic can improve the search in terms of quality of the solutions and running times. Some of the ideas that we are currently working on consider the mix of column generation techniques with tabu search or genetic algorithms. This framework is the next step of the approach used to improve the tabu search and the genetic algorithms by considering strategic optimized intensifications based on exact methods and the perfect offspring in genetic algorithms for larger instances, Lourenço (2001). Also, as a future research topic, we are interested in developing metaheuristics for the integration of vehicle and crew scheduling problems.

Acknowledgments

The authors would like to thank the referees, Prof. Stefan Voss, Prof. Daniel Serra (UPF) and Dr. Ana Paias (UL) for their remarks, that helped to improve the quality of this work.

Thanks are due to PRAXIS (/2/2.1MAT/139/94) for providing funding support for the work described in this paper.

References

C.C. Aggarwal, J.R. Orlin, and R.P. Tai, "Optimized crossover for the independent set problem", *Operations Research* 45(2):226-234 (1997).

A. Agra, "A method to generate feasible duties in a transportation system", *Working Paper*, Faculdade de Ciências da Universidade de Lisboa, 1993.

K.S. Al-Sultan, M.F. Hussain and J.S. Nizami, “A genetic algorithm for the set covering problem”, *Journal of the Operational Research Society* 47: 702-709 (1996).

J.E. Beasley, “An algorithm for the set covering problem”, *European Journal of Operational Research*, 31: 85-93 (1987).

J.E. Beasley and P.C. Chu, “A genetic algorithm for the set covering problem”, *European Journal of Operational Research* 94: 392-404 (1996).

K.W. Campbell, R.B. Durfee and G.S. Hines, “FedEx Generates Bid Lines using Simulated Annealing”, *Interfaces* 27:2 1-16 (1997).

A. Caprara, M. Fischetti and P. Toth, “A Heuristic method for the set covering problem”, *Operations Research*, 47: 730-743 (1999).

P.C. Chu and J.E. Beasley, “A genetic algorithm for the set partition problem”, *Working paper*, Imperial College, London, UK (1995).

R. Clement and A. Wren, “Greedy Genetic Algorithms, Optimising mutations and Bus Driver Scheduling”, 6th. International Workshop on Computer Aided Scheduling of Public Transportation, Lisboa, 1993.

J.R. Daduna, I. Branco and J. Paixão (eds.), “Computer-Aided Transit Scheduling”, *Proceedings of the Sixth International Workshop*, Springer-Verlag, Berlin, 1995.

J.R. Daduna and M. Mojsilovic, Computer-aided vehicle and duty scheduling using HOT programme system, in *Computer-Aided Transit Scheduling*, J.R. Daduna and A. Wren (eds), 133-146, Springer-Verlag, Berlin, 1988.

L. Davis, *“Handbook of Genetic Algorithms”*, Van Hostrand Reinhold, New York (1996).

T.A. Feo and M.G.C. Resende, “Greedy randomized adaptive search heuristic”, *Journal of Global Optimization* 6:109-133 (1995).

C.M. Fonseca and P.J. Fleming, “An overview of evolutionary algorithms in multiobjective optimization”, *Evolutionary Computation*, 3:1-16 (1995).

R. Freling, “Models and techniques for integrating vehicle and crew scheduling”, Ph.D. Thesis, *Erasmus University*, Rotterdam, Nederland (1997).

T. Galvão, J. Pinho de Sousa and J. Falcão e Cunha, “Genetic algorithms for the crew scheduling problem: a real experiment with relaxation models”, preprint (1998).

M.R. Garey and D.S. Johnson, “*Computers and Intractability - A Guide to the Theory of NP-Completeness*”, Freeman (1979).

F. Glover, "Future paths for integer programming and links to artificial intelligence", *Computers and Operations Research* 5: 533-549 (1986).

F. Glover and M. Laguna, "*Tabu Search*", Kluwer Academic Publishers, Norwell, Massachusetts (1997).

M.P. Hansen, "Tabu search for multiobjective optimization: Mots", *Proceedings of the 13th International Conference on Multiple Criteria Decision Making*, University of Cape Town, pages 574-586 (1997).

J. Holland, "*Adaptation in Natural and Artificial Systems*", University of Michigan Press, Michigan (1975).

R.M. Karp, "Reducibility among combinatorial problems", in *Complexity of Computer Computations*, Plenum Press, New York (1972).

A.S.K. Kwan, R.S.K. Kwan, and A. Wren, "Driver scheduling using Genetic Algorithms with embedded combinatorial traits", in *Preprints of the 7th International Workshop on Computer-Aided Scheduling of Public Transportation*, Boston, U.S.A. (1997).

A.S.K. Kwan, R.S.K. Kwan, M.E. Parker and A. Wren, "Producing train driver schedules under operating strategies", in *Preprints of the 7th International Workshop on Computer-Aided Scheduling of Public Transportation*, Boston, U.S.A., 1997.

R.S.K. Kwan, and A. Wren, "Hybrid genetic algorithms for the bus driver scheduling", in *Advanced methods in transportation analysis*, L. Bianco and P. Toth (eds), 609-619, Springer, Berlin, 1996.

M. Laguna, "A heuristic for production scheduling and inventory control in the presence of sequence-dependent setup times", *IIE Transactions*, 31(2): 125-134 (1999).

H.R. Lourenço, "Job-shop scheduling: computational study of local search and large-step optimization methods", *European Journal of Operational Research*, 83: 347-364 (1995).

H.R. Lourenço, "OSH: Optimized Search Heuristics", *in preparation*, Universitat Pompeu Fabra, Barcelona, Spain, 2001.

P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms", *C3P Report 826*, Caltech Concurrent Computation Program, (1989).

A.R. Odoni, J.M. Rousseau and N.H.M. Wilson, "Models in urban and air transportation", in *Operations Research and the Public Sector*, in *Handbooks in Operations Research and Management Science*, S.M. Pollock, M.H. Rothkopf and A. Barnett (eds), vol. 6, 129-150, North-Holland, Amsterdam, 1994.

I.H. Osman and J.P. Kelly (eds.), *"Meta-Heuristics: Theory and Applications"*, Kluwer Academic Publishers (1996).

I.H. Osman and G. Laporte, “Metaheuristics: a bibliography”, in *Annals of Operations Research*, 63: 513-628 (1996).

A. Paias and J. Paixão, “State space relaxation for set covering problems related to bus driver scheduling”, *European Journal of Operational Research* 71: 303-316 (1993).

R. Portugal, “Metaheuristics for the Bus-Driver Scheduling Problems”, MSc. Thesis, Faculdade de Ciências da Universidade de Lisboa, Portugal (1998).

J.M. Rousseau, R. Lessard and J.Y. Blais, “Enhancements to the HASTUS crew scheduling algorithm”, in *Computer Scheduling of Public Transport –2,* J.M. Rousseau (ed.), 295-310, North-Holland, Amsterdam, 1985.

J.M. Rousseau (ed.), “*Computer Scheduling of Public Transport –2”*, North-Holland, Amsterdam (1985).

B.M. Smith and A. Wren, “A bus driver scheduling system using a set covering formulation”, *Transportation Science*, 22A: 97-108 (1988).

R.E. Steuer, “*Multiple criteria optimization: Theory, computation, and application*”, John Wiley (1986).

F.J. Vasko and F. E. Wolf, “Solving large set covering problems on a personal computer”, *Computers and Operations Research* 15: 115-121 (1988).

A. Viana and J. Pinho de Sousa, “Some notes on multiobjective metaheuristics – a tutorial example”, submitted for publication to the *European Journal of Operational Research* (1999).

S. Voss, S. Martello, I.H. Osman, C. Roucairol, “*Metaheuristics: advances and trends in local search paradigms for optimization*”, Kluwer Academic Publishers (1998).

A. Wren and D.O. Wren, “A genetic algorithm for public transport driver scheduling”, *Computers and Operations Research* 22: 101-110 (1995).

A. Wren and J.M. Rousseau, “Bus driver scheduling - an overview “, in *Computer-Aided Transit Scheduling*, J. Daduna, I. Branco and J.P. Paixão (eds), 173-187, Springer Verlag, 1995.

A. Wren, “Scheduling, timetabling and rostering - a special relationship?”, in *Practice and Theory of Automated Timetabling*, E.K. Burke and P. Ross (eds), 46-75, Springer Verlag, 1996.