Public Transp (2009) 1: 189–210
DOI 10.1007/s12469-009-0013-x

ORIGINAL PAPER

Bi-objective evolutionary heuristics for bus driver rostering

Margarida Moz · Ana Respício · Margarida Vaz Pato

Published online: 1 August 2009

Abstract The Bus Driver Rostering Problem (BRP) refers to the assignment of drivers to the daily crew duties that cover a set of schedules for buses of a company during a planning period of a given duration, e.g., a month. An assignment such as this, denoted as roster, must comply with legal and institutional rules, namely Labour Law, labour agreements and the company's regulations. This paper presents a new bi-objective model for the BRP, assuming a non-cyclic rostering context. One such model is appropriate to deal with the specific and diverse requirements of individual drivers, e.g. absences. Two evolutionary heuristics, differing as to the strategies adopted to approach the Pareto frontier, are described for the BRP. The first one, following a utopian strategy, extends elitism to include an infeasible (utopic) and two potential lexicographic individuals in the population, and the second one is an adapted version of the well known SPEA2 (Strength Pareto Evolutionary Algorithm). The heuristics' empirical performance was studied through computational tests on BRP instances generated from the solution of integrated vehicle-crew scheduling problems, along with the rules of a public transit company operating in Portugal. This research shows that both methodologies are adequate to tackle these instances. However, the second one is, in general, the more favourable. In reasonable computation

M. Moz · M.V. Pato
ISEG, Technical University of Lisbon, Rua do Quelhas, 6, 1200-781, Lisboa, Portugal

M. Moz
e-mail: mmoz@iseg.utl.pt

M.V. Pato
e-mail: mpato@iseg.utl.pt

A. Respício (✉)
FC, University of Lisbon, DI, Bloco C6, piso 3, Cidade Universitária, 1749-016 Lisboa, Portugal
e-mail: respicio@di.fc.ul.pt

M. Moz · A. Respício · M.V. Pato
Centro de Investigação Operacional, University of Lisbon, Lisboa, Portugal

times they provide the company's planning department with several rosters that satisfy all the constraints, an achievement which is very difficult to obtain manually. In addition, among these rosters they identify the potentially efficient ones with respect to the BRP model's two objectives, one concerning the interests of administration, the other the interests of the workers. Both heuristics have advantages and drawbacks. This suggests that they should be used complementarily. On the other hand, the heuristics can, with little effort, be adapted to a wide variety of rostering rules.

Keywords Bus driver rostering · Bi-objective problems · Evolutionary algorithms

1 Introduction

Rostering is mainly a human resources management problem that may arise in several contexts, e.g., in the transport and in the health care sectors. Transport operators in particular require very careful management of human resources in view of the increasing burden of costs and the need to provide a high quality service. The rostering problem supports the production of work schedules for a set of workers—called roster—that must satisfy institutional and legal regulations, during a planning period. A good roster should also comply with workers' preferences. These are not only important for the workers themselves, but also for the employer, insofar as worker satisfaction lowers the incidence of professional illness, accidents and absenteeism. Consequently, rostering naturally calls for a multi-objective model to reconcile conflicting interests.

Personnel scheduling and rostering has been in the operations researchers' agenda since Dantzig's paper on scheduling collectors at toll booths (Dantzig 1954). There are many bibliographical references on rostering, as surveyed by Ernst et al. (2004). However, only a few studies have been published on transit crew rostering, with the same connotation as bus driver rostering studied here.

It should be noted that the bus driver rostering problem, as well as the vehicle and crew scheduling problems are the most important issues of the operational planning phase in public transit companies. However, bus driver rostering differs from crew scheduling insofar as rostering per se deals with the assignment of duties to a particular set of drivers of the company during the planning period, whereas crew scheduling builds a set of duties that cover the bus schedules for a short period, usually a single day, which are the outcome of the vehicle scheduling problem. In fact, the literature on crew scheduling is considerable, whereas that on bus driver rostering is limited. This may be so because, as opposed to the bus driver rostering process, crew scheduling is more standardised among public transit companies.

Previous approaches to bus driver rostering are surveyed by Odoni et al. (1994) and the most recent studies have been based on models designed to tackle the cases where drivers perform cyclic rosters (Emden-Weinert et al. 2001; Pedrosa and Constantino 2001). Cyclic rostering easily embeds the balancing of the workload among drivers, but does not cope with certain real world specifics, namely the particular requirements of each driver. We argue the importance of personalised schedules, adapted to satisfy the different interests and needs of drivers, such as absences for holidays, scheduled

medical assistance, trade union service, training, etc. In fact, in a non-cyclic rostering strategy, as drivers are not obliged to cycle from week to week within a set of predefined work schedules, their work schedules cope with those conditions.

As to other rostering problems within the transport sector, those arising in air transport companies share few similarities with bus driver rostering (see, for instance, Cappanera and Gallo 2004; Kohl and Karisch 2004). Though the cases of train or underground crew rostering are not that different from bus driver rostering, they have also been tackled on a cyclic basis (Caprara et al. 1997, 1998; Dornberger et al. 2008; Hartog et al. 2009; Lezaun et al. 2006; Sodhi and Norris 2004).

The mathematical models for bus, train or underground driver rostering found in the literature are either based on networks (Bianco et al. 1992; Caprara et al. 1997; Carraresi and Gallo 1984); or within set covering/partitioning (Caprara et al. 1997, 1999; Catanas and Paixão 1995; Freling et al. 2004; Portugal et al. 2009). In any case, the exact methodologies do not handle all real world applications of driver rostering problems, namely those that give rise to high dimension instances. Hence, the solution approaches have been a mix of exact, heuristic and relaxation based methods or heuristic methods alone.

The multi-objective nature has been tackled in various rostering issues, such as train and/or air crew rostering (Caprara et al. 1998; Dornberger et al. 2008; Lucic and Teodorovic 1999, 2007), nurse rostering (Moz and Pato 2007; Silva and Le 2008) and airport staffing (Chu 2007). Catanas and Paixão (1995) proposed a bi-objective set covering formulation for bus driver rostering. The model considers minimising both the maximum workload and the total roster cost. The first objective is implicitly taken into account by introducing side constraints. In order to reconcile the interests of drivers and the employer in bus transit companies, Emden-Weinert et al. (2001) proposed a decision support system for rostering with multiple objectives, as well as algorithms combining integer linear programming and local search. For a general survey on multi-objective issues for personnel scheduling in different contexts see Silva et al. (2004).

The above reasoning supports the need for new methodologies to efficiently tackle real instances of highly constrained multi-objective non-cyclic driver rostering problems. The Bus Driver Rostering Problem is presented through a bi-objective model in Sect. 2 of this paper, along with a mathematical formulation in Sect. 3. Section 4 proposes two evolutionary heuristics, followed by Sect. 5 which provides computational results. Final comments can be found in Sect. 6.

2 Problem description

The Bus Driver Rostering Problem (BRP) consists of building a work schedule for each of the company's bus drivers who are assigned to a specific depot, over a planning period of 28 days, the rostering period, so as to cover the driving demand for the period, i.e., the entire set of (daily crew) duties required to perform all the trips of the depot's vehicles. The rostering period is, as usual, assumed to begin on a Monday, thus consisting of four complete weeks, i.e. a month.

Here, by 'work schedule' one means a sequence of duties and days off, whose length is 28, to be assigned to a single driver for the period; while 'duty' refers to a

set of pieces of work with breaks and idle times, to be performed by a single driver during a specific day. The set of all the drivers' work schedules in a rostering period is called a roster. Note that some authors use roster in the same sense as the above-mentioned work schedule. Hence, this problem takes the duties for each day of the rostering period, i.e., the output of 28 crew scheduling problems, and produces a roster, i.e., a set of work schedules, one per driver.

Rosters must comply with the conditions and rules imposed by labour union contracts, institutional and legal requirements that establish the hard constraints, such as the number of days off per week, specific days off per week, a minimum/maximum number of days for the length of a rest period, a minimum number of Sundays/weekends off in the planning period, a minimum number of rest hours between two consecutive work periods and a minimum/maximum number of days for the length of a work period. Rosters that cover the demand and satisfy the established conditions and rules (the hard constraints) may be distinguished by the different features that qualify them as good or bad, according to distinct interests or points of view. For example, good rosters may be characterised by equity in the distribution of Sundays/weekends off among drivers, equity in the distribution of overtime work, equity in the distribution of late duties and, where administration goals are concerned, the low cost of assignments, reduction of part-time workloads, and small gaps between each driver's overall scheduled hours and the respective contracted hours per period. These requirements are generally taken as soft constraints, as distinct from hard ones, or even as objectives to attain.

The rostering problem studied throughout this paper considers a set of hard constraints and two optimisation objectives that will be detailed next. However, both the formalisation, described in Sect. 3, and the heuristics, described in Sect. 4, can be adapted to any of the above-mentioned or other situations, always within the framework of non-cyclic rostering.

For each rostering period of 28 days, the set of duties to be covered, and the respective starting time and length, are known in advance. The hard constraints of the BRP may be summarised as follows:

(h1) each duty must be assigned to one and only one driver;
(h2) each driver must be assigned one and only one duty or a day off, each day;
(h3) some drivers must be granted specific days off due to planned absences (for instance, holidays), or all weekends off, in view of seniority;
(h4) each driver must enjoy a minimum rest period between consecutive duties, thus defining compatible consecutive duties per driver;
(h5) drivers must not work more than g consecutive days;
(h6) drivers must have at least d_w days off a week;
(h7) drivers must have at least d_s Sundays off in each rostering period;
(h8) drivers must work no more than b_1 hours per week and b_2 hours per rostering period.

The rostering within the BRP is guided by the two following objectives:

(o1) minimise the maximum total overtime during the rostering period, per driver;
(o2) minimise the number of drivers whose workload is positive but less than q duties, the contractual number of duties per rostering period.

The total overtime per driver, in objective (o1), is the sum of the daily excess workload over the contractual workload, $\bar{t}$. Note that the two objectives in fact represent the interests of both entities involved in the problem: bus drivers and the company's administration. By minimising the number of drivers working below the contractual workload, q duties—objective (o2)—more drivers of the depot are left with no duty assignments. Hence, those drivers can be assigned to other depots of the company. Objective (o2) at the same time increases the scheduled workload of the other drivers—fewer days off—thus raising total overtime per driver. On the other hand, objective (o1) not only reduces total individual overtime, and distributes it equitably among drivers, but also enlarges the set of drivers assigned to work. Therefore, it also enlarges the set of the drivers that work less than q duties per rostering period.

In short, the BRP sets out to determine a roster capable of fulfilling the demand expressed by the set of duties for the rostering period, besides satisfying the hard conditions, (h1) to (h8), and optimising objectives (o1) and (o2) defined from the most relevant soft constraints faced by driver rostering in the company.

3 Mathematical formulation

The BRP can be modeled as a bi-objective binary non-linear programming problem as follows.

First, the parameters and the sets necessary for the formulation are introduced:

$V =$ set of drivers of the company;
$g =$ maximum number of consecutive days without a day off;
$T_h^w =$ set of duties of day $h, h = -g+1, \ldots, 0, 1, \ldots, 28$;
$T_h^\vartheta = \{\vartheta\}$, the set with the day off for each day $h, h = -g+1, \ldots, 0, 1, \ldots, 28$;
$T_h =$ set whose elements are the duties of day h and the day off taken on day h, that is $T_h^w \cup T_h^\vartheta, h = -g+1, \ldots, 0, 1, \ldots, 28$;
$T_{ih}^v =$ set including the day off and the duties of day h that are compatible for driver v with duty i of day $h-1, v \in V, i \in T_{h-1}, h = 1, \ldots, 28$;
$t_{ih} =$ length of duty i on day $h, i \in T_h^w, h = 1, \ldots, 28$;
$\bar{t} =$ contractual daily working time of a driver;
$t'_{ih} = \max\{0, t_{ih} - \bar{t}\}, i \in T_h^w, h = 1, \ldots, 28$;
$b_1 =$ maximum total work time per week per driver;
$b_2 =$ maximum total work time per rostering period per driver;
$d_w =$ minimum number of days off per week per driver;
$d_s =$ minimum number of Sundays off per rostering period per driver;
$q =$ contractual number of duties (working days) of a work schedule;
$F^v =$ set of obligatory days off for driver v during the rostering period, defined by hard constraint (h3), $v \in V$;

$$e_{ih}^v = \begin{cases} 1, & \text{if driver } v \text{ performed duty } i \text{ on day } h \text{ of the previous rostering period} \\ 0, & \text{otherwise,} \quad v \in V,\ i \in T_h^w, h = -g+1, \ldots, 0. \end{cases}$$

Three types of binary variables are used to formulate the BRP:

$y_{ih}^v =$ 1—if driver v performs duty i on day h or, in case $i = \vartheta$, gets a day off on that day, 0 otherwise, $v \in V, i \in T_h, h = 1, \ldots, 28$;

$\omega^v =$ 1—if driver v does not work or is assigned to less than q duties during the rostering period, 0 otherwise (driver v works q or more duties), $v \in V$;

$\eta^v =$ 1—if driver v is assigned to at least one duty during the rostering period, 0 otherwise, $v \in V$.

The formulation follows:

$$\text{minimise} \quad \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} \max_{v \in V} \sum_{h=1}^{28} \sum_{i \in T_h^w} t'_{ih} y_{ih}^v \\ \sum_{v \in V} (\omega^v + \eta^v) - |V| \end{bmatrix}, \tag{1}$$

subject to:

$$\sum_{v \in V} y_{ih}^v = 1, \quad i \in T_h^w, \ h = 1, \ldots, 28, \tag{2}$$

$$\sum_{i \in T_h} y_{ih}^v = 1, \quad v \in V, \ h = 1, \ldots, 28, \tag{3}$$

$$\sum_{v \in V} \sum_{h \in F^v} \sum_{i \in T_h^w} y_{ih}^v = 0 \tag{4}$$

$$y_{i,h-1}^v + \sum_{j \in T_h \setminus T_{ih}^v} y_{jh}^v \le 1, \quad i \in T_{h-1}, v \in V, h = 2, \ldots, 28, \tag{5}$$

$$e_{i0}^v + \sum_{j \in T_1 \setminus T_{i1}^v} y_{j1}^v \le 1, \quad i \in T_0, v \in V, \tag{5'}$$

$$\sum_{l=0}^{g} \sum_{i \in T_{h+l}^w} y_{i,h+l}^v \le g, \quad v \in V, h = 1, \ldots, 28 - g, \tag{6}$$

$$\sum_{l=h}^{0} \sum_{i \in T_l^w} e_{il}^v + \sum_{l=1}^{h+g} \sum_{i \in T_l^w} y_{il}^v \le g, \quad v \in V, h = -g + 1, \ldots, 0, \tag{6'}$$

$$\sum_{h=7(l-1)+1}^{7l} y_{\vartheta h}^v \ge d_w, \quad v \in V, l = 1, \ldots, 4, \tag{7}$$

$$\sum_{l=1}^{4} y_{\vartheta,7l}^v \ge d_s, \quad v \in V, \tag{8}$$

$$\sum_{h=7(l-1)+1}^{7l} \sum_{i \in T_h^w} t_{ih} y_{ih}^v \le b_1, \quad v \in V, l = 1, \ldots, 4, \tag{9}$$

$$\sum_{h=1}^{28} \sum_{i \in T_h^w} t_{ih} y_{ih}^v \le b_2, \quad v \in V, \tag{10}$$

$$-\sum_{h=1}^{28}\sum_{i\in T_h^w} y_{ih}^v + q - 28\omega^v \le 0, \quad v \in V, \tag{11}$$

$$\sum_{h=1}^{28}\sum_{i\in T_h^w} y_{ih}^v - 28\eta^v \le 0, \quad v \in V, \tag{12}$$

$$y_{ih}^v \in \{0,1\}, \quad v \in V, i \in T_h, h = 1,\ldots,28, \tag{13}$$

$$\omega^v \in \{0,1\}, \quad v \in V, \tag{14}$$

$$\eta^v \in \{0,1\}, \quad v \in V. \tag{15}$$

Sets (2), (3) and (4) of equalities impose hard constraints (h1), (h2) and (h3), respectively. Moreover, constraints (5) and (5′) force hard constraints (h4) as a result of the definition of compatible duties. Inequalities (6) and (6′) formalise the hard constraint (h5), whereas (h6) to (h8) are forced by (7) to (10).

Inequalities (11), along with minimisation of the objective function f_2 force the variable ω^v to be equal to 1, when driver v is assigned to less than q duties per rostering period or does not work at all during the period, for each $v \in V$; otherwise ω^v is set equal to 0. Meanwhile, inequalities (12) along with minimisation of f_2 impose the value 1 for the variable η^v, when driver v is assigned to driving work during the period, otherwise it is set equal to 0, for each $v \in V$.

Finally, conditions (13) to (15) impose the domains of the variables.

The two objectives of BRP, (o1) and (o2), are formalised by minimising the vectorial function in (1). In fact, the first component f_1 represents the maximum overtime per driver and f_2 the total number of drivers assigned to work but with an incomplete workload, less than q duties, during the rostering period. This can be seen from the following rewriting of the formula for this objective function in (1):

$$f_2 = \sum_{v\in V}(\omega^v + \eta^v) - |V| = \sum_{v\in V}\omega^v - \left(|V| - \sum_{v\in V}\eta^v\right)$$

where the second expression in brackets represents the number of drivers at the depot who are not assigned to driving during the period.

The BRP can be proved to be an NP-hard problem on the strength of arguments similar to those used to prove NP-hardness in the case of the nurse rerostering issue in Moz and Pato (2007).

Computational experiments revealed the difficulty in reaching optimality for some real BRPs. In fact, the experience described in Mesquita et al. (2008), with a set of 50 instances and a corresponding number of duties from 420 to 1404, has shown that even for the single objective problem minimise f_1, after linearisation, the CPLEX 11 software optimiser (CPLEX 2007) did not achieve exact solutions for most cases, within four CPU hours. This experience, already anticipated in view of the high complexity of the problem, besides the dimension of its real life instances, motivated the authors to use evolutionary heuristics. As a rule, they are well equipped to deal with high complexity problems and have been successfully applied to other bi-objective

problems, including nurse (re)rostering issues (Aickelin and Dowsland 2004; Pato and Moz 2008).

4 Evolutionary heuristics

Two evolutionary heuristics were developed for the BRP: the Utopic Genetic Heuristic (UGH) and the Adapted SPEA2 (ASP), which differ as to the strategy used to approximate the (exact) Pareto frontier for BRP.

4.1 Common features

The encoding of individuals of the population and the decoder (to obtain solutions from the individuals) are identical for both evolutionary algorithms being tested.

An individual is characterised by a pair of chromosomes: one associated with a list of duties—duty chromosome—and the other with a list of drivers—driver chromosome. The first one is a permutation of all the duties to be assigned during the rostering period—with length $r = \sum_{h=1}^{28} |T_h^w|$—and the second one a permutation of the drivers of the depot—with length $|V|$. Both are coded by integer vectors.

The main aim of the evolutionary heuristic is to determine feasible solutions for the BRP, i.e., rosters. Consequently, each individual of the population must be decoded so as to produce a roster. As the rostering problem, in fact, includes many and different hard constraints, namely those concerning a non-homogeneous workforce, the problem of finding feasible solutions is very difficult. Thus, a constructive heuristic (the Decoder algorithm) was carefully designed to ensure that the solutions produced do satisfy all the hard constraints as much as possible. In fact, simple forward planning heuristics cannot easily deal with all the rostering rules of the BRP. Therefore, to obtain a roster one must take the pair of (different type) chromosomes characterising a specific individual and run the algorithm Decoder in its CONSTR version. This process corresponds to a specially devised constructive heuristic using the two chromosomes as input.

The algorithm Decoder(CONSTR), described in Fig. 1, starts from the two lists, represented respectively by the vectors $duties[i]$, with $i = 1, \ldots, r$, and $drivers[j]$, with $j = 1, \ldots, |V|$, considering that all duties are set free (non-assigned) and all drivers are also set free—STEP 1. In the main step, STEP 2, the algorithm tries to attribute each non-assigned duty to a driver, while sequentially testing the hard constraints. The testing order, shown in STEP 2.1, comes from the need for computational efficiency, the last constraints being the most difficult ones to satisfy. It is important to stress that this constructive heuristic may yield an infeasible solution, i.e., a 'roster' which does not satisfy all the hard constraints. In STEP 3, if a feasible solution is found, in order to evaluate the fitness of the respective individual, one calculates the values of the two objective functions f_1 and f_2—in the case of the bi-objective algorithms UGH (main iteration) or ASP—or the value of one objective function alone—in the case of the single objective auxiliary genetic procedure to determine the first generation's population or the utopic individual, see Sect. 4.2. Otherwise, the solution is infeasible and a penalised fitness value is assigned to the corresponding individual.

ALGORITHM Decoder(CONSTR or IMPROV)

STEP 1. {initialisation}
1.1 Set the list of *duties* equal to the duty chromosome.
1.2 Set the list of *drivers* equal to the driver chromosome.
1.3 All the drivers for all days are set free and all the duties are set free.
1.4 Set the index of the first duty to assign $i = 1$.

STEP 2. {assignments}
2.1 Search for the minimum j such that *drivers*[j] is a free driver to assign *duties*[i], verifying the hard constraints in the order (h3), (h5), (h6), (h4), (h7) and (h8).
IF a free driver is found, *drivers*[j], THEN
2.2 Assign *duties*[i] to *drivers*[j], set *duties*[i] as non free and *drivers*[j] as non free for the corresponding day;
$i = i + 1$;
IF $i \leq r$ THEN GO TO 2.1;
ELSE {it is not possible to assign *duties*[i] to any driver}
2.3 IF version=CONSTR THEN GO TO 3.
IF version=IMPROV THEN
REPEAT
Backtrack to the last duty already assigned, *duties*[$i - 1$];
Insert *duties*[i] in position $i - 1$, pushing *duties*[$i - 1$] and its successors one position forward;
FOR all $k = i$ to r DO set *duties*[k] free and the driver assigned to *duties*[k] free for the corresponding day;
$i = i - 1$;
Search for the minimum j such that *drivers*[j] is a free driver to assign *duties*[i] verifying: (h3), (h5), (h6), (h4), (h7) and (h8);
UNTIL (j is found OR $i = 1$).
IF j is found {*duties*[i] can be assigned to *drivers*[j]} THEN GO TO 2.2;
ELSE {*duties*[i] cannot be assigned to any driver} GO TO 3.

STEP 3. {stopping criterion}
IF all the duties are non-free THEN {a roster is found}
Compute the objective value(s) associated with the roster;
ELSE {the solution is infeasible}
Assign an adequate penalty to the objective value(s).

STOP

Fig. 1 Description of the Decoder algorithm

Either in UGH or in ASP the initial population is randomly generated, but within UGH a genetic initialisation phase follows, searching for feasibility in the population, in accordance with the description of the respective algorithm.

In both cases, if after the first *NINIC* generations no feasible solution is found by the genetic search, then one individual is randomly chosen from the population of generation *NINIC* and feasibility is enforced by using an improved constructive

heuristic, denoted as Decoder(IMPROV), acting as the Decoder(CONSTR) but in a more sophisticated way. The Decoder(IMPROV) is a constructive heuristic enhanced with a kind of simplified local search that tries to find a feasible solution for BRP, searching for feasibility within local neighbours, as briefly described in STEP 2.3 of Fig. 1 for algorithm version IMPROV. This procedure is called for when, at some point of the assigning process, a duty cannot be assigned to any free driver, meaning that the assignment by the specific order cannot reach a feasible solution. The main idea behind IMPROV is to move back in the process by cancelling some previous assignments, then to insert that duty in a lower position in the list and resume the process from that point forward.

In each of the *NGER* generations of both bi-objective evolutionary approaches, UGH and ASP, the cardinality of the population is fixed at p individuals, equal for all generations. The selection operator randomly picks p individuals from the respective population, thus forming a mating pool with cardinality equal to p.

Crossover and mutation operators act independently on the two types of chromosomes of the individuals taken from the mating pool. The crossover operator is applied in the same way on the chromosomes of the same type (duty or driver) of each pair of individuals randomly chosen for recombination. After the crossover operation, two new duty chromosomes are produced, as well as two new driver chromosomes. By matching each chromosome of one type with a chromosome of the other type an offspring is obtained. The two offspring replace the two parents in the population. The mutation operator is also applied in the same way on both duty and driver chromosomes of an individual randomly picked for mutation. This operator acts on that chromosome by swapping the positions of two of its randomly chosen genes. In the sequence, the Decoder(CONSTR) acts independently on the new individuals (to produce new rosters) and computes the respective fitness values.

4.2 Utopic genetic heuristic

In evolutionary heuristics, because of the optimisation objective(s), the fitness of each individual must reflect the quality of the corresponding solution. In the case of UGH the fitness of an individual is calculated according to the position of the respective point in the objective space in relation to the Pareto frontier of the current generation's population, called Pareto rank (Goldberg 1989). Here, the fitness is simply defined as the inverse of the Pareto rank. For individuals corresponding to rosters that do not satisfy all the hard constraints, the respective objective values are penalised, as mentioned above. As a result, these individuals belong to the last rank and obtain the lowest fitness value.

Below, Fig. 2 outlines the pseudo-code of the evolutionary heuristic UGH. The initial population is generated, in STEP 1.1, by running a single objective evolutionary algorithm, guided by objective (o1) alone, sharing all the other features with algorithm UGH, with the obvious exception of the fitness evaluation and number of iterations, here *INGER*.

The selection operator used to obtain the mating pool is based on the roulette wheel procedure, which states that the probability of an individual's entering the mating pool is directly proportional to its fitness. In generation t, the genetic operators act

ALGORITHM UGH

STEP 1. {initialisation}
1.1 Create the initial population $Pop(1)$.
1.2 Determine the utopic individual.
1.3 $t = 1$.

STEP 2. {evaluation and selection}
2.1 FOR all individuals $i \in Pop(t)$ DO call algorithm Decoder(CONSTR) to compute objective values for i.
2.2 Compute Pareto ranks and assign the fitness value to each $i \in Pop(t)$.
2.3 Build a mating pool by selecting p individuals from $Pop(t)$.

STEP 3. {recombination and mutation}
3.1 Perform crossover over individuals in the mating pool and submit them to mutation.
3.2 Update $Pop(t + 1)$.
3.3 Perform elitism within $Pop(t + 1)$.
3.4 $t = t + 1$.

STEP 4. {stopping criterion}
4.1 IF $t \leq NGER$ THEN GO TO STEP 2.

STOP

Fig. 2 Description of the UGH algorithm

over the individuals of the mating pool, thus creating the population for the next generation, $Pop(t+1)$. The crossover operator used here is a standard PMX crossover (Goldberg 1989), specific for permutation encoding.

Elitism is performed by forcing three individuals into the population of generation $(t + 1)$, $Pop(t + 1)$:

- one individual corresponding to a probably infeasible solution, called utopic individual;
- two individuals, *lexic1* and *lexic2*, corresponding to the lexicographic points of the previous generation, t, i.e., those non-dominated individuals that register the best value for the objective functions, f_1 and f_2, respectively;

to substitute three individuals randomly chosen from the mating pool.

The utopic individual corresponds to a feasible solution of the relaxation of BRP obtained by eliminating three hard constraints (h4), (h6) and (h7). This individual is the best suited individual resulting from the application of an auxiliary single objective genetic algorithm, in STEP 1.2. Such an algorithm operates as the one that initialises the UGH population, the exception being that here the decoder ignores constraints (h4), (h6) and (h7). Consequently, the utopic individual might correspond to an infeasible solution for the original problem, BRP. The fitness value of the utopic individual is set equal to 1. In this way, it will always have the highest probability of being selected for the mating pool.

The elitist strategy of UGH was devised to both enforce the search towards the exact Pareto frontier of the bi-objective optimisation problem, BRP and increase diversity in the approximate Pareto frontier. The utopic individual plays the role of an attractor for good quality solutions of the bi-objective problem. It pushes the solutions generated by the evolutionary algorithm in the direction of the lowest value for (o1), which is the most difficult objective as revealed by past experience with similar models (Mesquita et al. 2008). Eventually, the inclusion of the utopic favours the lowest value for (o2) as well. As for diversity, inclusion of the lexicographic and the utopic individuals promotes the spread of the individuals along the approximation of the exact Pareto frontier.

It should be noted that the algorithm UGH was developed by taking the specific characteristics of the BRP—the hard constraints, (h1) to (h8), and the objectives, (o1) and (o2)—following the general features of the genetic heuristic for the single objective nurse rerostering problem by Moz and Pato (2007). The fitness of the individuals is derived from a Pareto genetic approach, enhanced with the utopic elitism adapted from Pato and Moz (2008) for a nurse rerostering problem involving constraints and objectives that differ from those tackled here.

4.3 Adapted SPEA2

The algorithm ASP is an adaptation of the SPEA2—the improved Strength Pareto Evolutionary Algorithm (Zitzler et al. 2002). SPEA2 has been compared with other state of the art multi-objective heuristic techniques, and is well known for producing excellent results. The main features of ASP, whose brief pseudo-code is presented in Fig. 3, include elitism, a fine-grained fitness assignment scheme and diversification of solutions in each generation. As stated above, ASP embeds the main components of UGH, including two chromosomes characterising each individual, the decoder procedure, the fixed cardinality of the population and the mutation operator.

In each generation t, the algorithm considers a population $Pop(t)$, with a fixed size p, and an archive $Arc(t)$, whose size is also set constant and equal to $a, a < p$, to keep non-dominated individuals found as the genetic search progresses.

In STEP 2.1, the fitness value of the individual i, represented by $F(i)$, is computed in three phases. Firstly, the force of each individual counts the number of individuals it dominates within the population and the archive. Secondly, the raw fitness of each individual sums up the forces of all the individuals that dominate it. Thirdly, an estimate for the diversity measure is added to the raw fitness whose aim is to penalise the individuals whose points are in more crowded regions of the objective space. This measure is given by $1/(\sigma^k + 2)$, where σ^k is the distance to the kth nearest neighbour with $k = \lfloor\sqrt{p+a}\rfloor$, the integer part of the square root of the total number of individuals in the current generation, t. Individuals corresponding to potentially efficient solutions receive a fitness value below 1, meaning that the genetic search is oriented to minimise fitness.

Elitism is enforced, in generation t, by keeping non-dominated individuals in the archive $Arc(t+1)$, built in STEP 2.2 of the pseudo-code. Here, the truncation operator acts in keeping with the original SPEA2 (Zitzler et al. 2002). If the number of non-dominated individuals exceeds the archive size, a, then the individual at the minimum

ALGORITHM ASP

STEP 1. {initialisation}

1.1 Create the initial population $Pop(1)$ by randomly generating p pairs of duty and driver chromosomes.
1.2 Create an empty archive $Arc(1)$.
1.3 $t = 1$.

STEP 2. {evaluation and selection}
2.1 FOR all individuals $i \in Pop(t) \cup Arc(t)$ DO
Call algorithm Decoder to compute objective values for i;
Compute $F(i)$.
2.2 Copy non-dominated individuals in $Pop(t) \cup Arc(t)$ to $Arc(t+1)$.
IF $|Arc(t+1)| < a$ THEN fill $Arc(t+1)$ with
minimal fitness individuals from $Pop(t)$.
IF $|Arc(t+1)| > a$ THEN apply the truncation operator.
2.3 Build a mating pool by selecting p individuals from $Arc(t+1)$.

STEP 3. {recombination and mutation}
3.1 Perform crossover over individuals in the mating pool and submit them to mutation.
3.2 Update $Pop(t+1)$.
3.3 $t = t + 1$.

STEP 4. {stopping criterion}
4.1 IF $t \leq NGER$ THEN GO TO STEP 2.

STOP

Fig. 3 Description of the ASP algorithm

distance to another individual in the archive is chosen for removal. Ties are broken by considering the following smallest distances. Still in STEP 2 (2.3), the mating pool is produced by performing selection through a binary tournament over individuals in the archive. Each time selection acts, this operator randomly chooses two individuals from $Arc(t+1)$. The individual with the lowest fitness value is introduced in the mating pool. Ties are solved by randomly choosing one of the two candidates.

In STEP 3, the recombination operator is a standard OX crossover (Goldberg 1989) and mutation is common to the algorithm UGH and was presented in Sect. 4.1.

5 Computational results

The proposed approaches were tested on 21 instances obtained from the solutions of the integrated vehicle-crew scheduling problem (Mesquita and Paias 2008; Mesquita et al. 2009) for the case of a single depot, along with the institutional requirements and norms of a bus transit company in Lisbon, besides the Portuguese Labour Law and the drivers' union contracts.

Here, each set of trips (see http://people.few.eur.nl//huisman/instances.htm) is used to build a daily set of vehicle schedules and a set of (daily crew) duties by running an algorithm for the integrated vehicle-crew scheduling problem. This algorithm runs for a typical week day (a Monday) and for a typical weekend (a Saturday). Afterwards, by copying these results for all the other days of the rostering period, the set of duties for the BRP is determined. In fact, following some of the company's transport demand patterns, the sets of duties are established as equal for week and for weekend days, respectively. Consequently, the total number of duties for the rostering period of four weeks results from the formula $20|T_1^w| + 8|T_6^w|$, where 1 is the index for Monday and 6 for Saturday.

In keeping with reality in most bus transit companies, two different types of duties are considered: early and late duties. In the sequel, for definition of the sets of compatible duties it is assumed that non-satisfaction of (h4) occurs when a driver works on a late duty on one day and on an early duty the following day.

The main data of these instances is as follows:

- $|V| = 45$ drivers; $g = 6$ days;
- relevant data on duties, i.e., the starting time and length (in quarters of an hour) as well as identification of the early duties, starting between 6:00 h and 15:30 h, and of the late duties, starting after 15:30 h until 24:00 h;
- T_h^w—set of duties of day h, $h = 1, \ldots, 28$, corresponding to the days of the rostering period, and $h = -5, \ldots, 0$ to the last six days ($g = 6$) of the previous period;
- $\overline{t} = 32$ quarters (8 hours); $b_1 = 192$ quarters (48 hours); $b_2 = 704$ quarters (176 hours);
- $d_w = 2$ days; $d_s = 1$ day; $q = 20$ duties or working days;
- F^v, $v \in V$—set of compulsory days off for driver v;
- e_{ih}^v, $v \in V, i \in T_h^w, h = -5, \ldots, 0$—giving the assignments of each driver in the last six days of the previous rostering period.

Next, implementation details are given. The computational experiment undertaken with the UGH ran on a Pentium IV Dual Core, 2Duo 1.8 GHz processor and the respective procedures were coded in Pascal (Borland Delphi 5.0). The ASP procedures were coded in C and ran on a 2.8 GHz and 512 Mb RAM Intel Pentium IV processor. Several possibilities for the algorithm parameters were tested and the results favoured the values used in the current study:

- $NGER = 2000$; $INGER = 400$; $NINIC = 200$; $p = 40$ for UGH; $p = 29$ and $a = 10$ for ASP;
- crossover and mutation probabilities are 0.8 and 0.2, respectively.

Tables 1, 2 and 3 present results taken from the last generation of each of the ten runs per instance identified in column 1. The BRP instances are grouped according to the dimension of the integrated vehicle-crew scheduling instances from which they are taken. One set involves 80 trips per week day (P0_80 to P10_80), and the other 100 trips per week day (P1_100 to P10_100). In order to access the dimension of the BRP instances, column 1 also displays the respective total number of duties for the rostering period.

Table 1 Computational results of UGH—averages from ten runs per instance

Instances/n. duties	UGH—absolute metrics				
	n. feasible individuals	Wave	Approximate Pareto frontier size	Spread	n. candidate efficient solutions
1	2	3	4	5	6
P0_80/708	39.0	12.2	4.0	17.0	4.8
P1_80/456	39.0	11.8	4.2	16.0	6.2
P2_80/352	39.0	11.6	4.4	17.2	5.5
P3_80/472	39.0	10.7	3.6	9.5	6.1
P4_80/456	39.0	12.3	3.7	13.8	4.6
P5_80/444	39.0	12.9	3.7	12.3	5.0
P6_80/416	39.0	11.6	3.8	15.9	7.3
P7_80/476	39.0	12.0	3.8	12.8	5.0
P8_80/568	39.0	11.2	3.8	13.6	6.1
P9_80/440	39.0	11.7	3.9	17.5	5.6
P10_80/592	39.0	11.2	4.2	12.0	5.3
tot. average_80	39.0	11.7	3.9	14.3	5.6
P1_100/720	39.0	11.1	4.4	14.9	5.5
P2_100/628	39.0	10.8	4.7	14.7	5.6
P3_100/648	39.0	11.6	3.9	17.0	5.2
P4_100/572	39.0	11.0	3.6	12.3	5.5
P5_100/648	39.0	11.6	3.7	15.8	5.0
P6_100/516	39.0	11.3	3.5	14.9	6.6
P7_100/624	39.0	12.7	3.7	16.6	4.6
P8_100/656	39.0	12.8	3.9	22.5	4.7
P9_100/704	39.0	11.8	4.1	15.0	5.7
P10_100/452	39.0	12.4	4.1	20.4	5.4
tot. average_100	39.0	11.7	4.0	16.4	5.4

At this point, one does not know if the solutions obtained from the evolutionary heuristics are indeed efficient ones, nor does one know the distance from the respective approximate Pareto frontiers to the exact frontier. However, the approximate frontiers built by these heuristics were studied from different standpoints, by using simple metrics defined, for instance, in Collette and Siarry (2005). The following columns found in Tables 1 and 2 show some absolute metrics for the quality of the results from the last generation of the UGH and from the ASP respectively, whereas Table 3 evaluates both evolutionary heuristics through relative behaviour metrics, as explained later.

In Tables 1 and 2, column 2 presents the average number of individuals of the last population, in case of UGH, and of the last population plus the archive, in the case of ASP, that correspond to feasible solutions (rosters) of the respective BRP instance.

Table 2 Computational results of ASP—averages from ten runs per instance

Instances/n. duties	ASP—absolute metrics				
	n. feasible individuals	n. points in the last generation	Approximate Pareto frontier size	Spread	n. candidate efficient solutions
1	2	3	4	5	6
P0_80/708	39.0	9.4	2.2	1.5	9.8
P1_80/456	39.0	6.8	2.4	2.8	11.2
P2_80/352	39.0	7.8	3.0	3.2	10.4
P3_80/472	39.0	6.8	2.4	2.8	14.2
P4_80/456	39.0	8.2	2.0	2.4	9.6
P5_80/444	39.0	13.6	2.2	1.8	13.0
P6_80/416	39.0	8.4	2.2	2.1	13.6
P7_80/476	39.0	11.4	2.4	3.8	13.2
P8_80/568	39.0	11.6	2.0	2.2	14.0
P9_80/440	39.0	9.8	2.4	2.6	12.0
P10_80/592	39.0	10.2	1.8	1.6	13.8
tot. average_80	39.0	9.5	2.3	2.4	12.3
P1_100/720	39.0	11.9	1.3	0.9	13.2
P2_100/628	39.0	9.8	1.6	1.3	12.8
P3_100/648	39.0	13.4	1.7	3.2	12.9
P4_100/572	39.0	9.6	2.2	1.4	13.6
P5_100/648	39.0	11.6	2.0	1.5	11.0
P6_100/516	39.0	12.4	1.9	1.7	15.0
P7_100/624	39.0	15.6	2.6	3.2	12.4
P8_100/656	39.0	17.2	2.6	4.0	11.0
P9_100/704	39.0	12.1	2.1	1.9	10.4
P10_100/452	39.0	8.9	2.2	1.6	14.9
tot. average_100	39.0	12.3	2.1	2.1	12.7

Column 3 in Table 1 refers to the wave metric—the average number of different ranks, again in the population of the last generation. This wave metric is only calculated for the UGH, as it is the only one to use the Pareto rank fitness function, according to the descriptions in Sects. 4.2 and 4.3. As for column 3 in Table 2, it indicates the number of different points in the objective space that correspond to the individuals of the last generation attained with the ASP.

Columns 4 and 5 of the same tables display the average size of the approximate Pareto frontier of the last generation—the number of non-dominated points in the objective space—and the average of the spread metric (see Collette and Siarry 2005). Here the spread metric is set equal to the Euclidean distance between the two lexicographic points of the last generation, the outer points of the final Pareto frontier

Table 3 Comparative computational results of the evolutionary heuristics

Instances/n. duties	Relative metrics				Size of the best frontier
	Potential non-dominated points		Contribution to the best frontier		
	UGH	ASP	UGH	ASP	
1	2	3	4	5	6
P0_80/708	0.80	1.00	0.67	0.50	6
P1_80/456	0.50	1.00	0.50	0.50	4
P2_80/352	0.25	1.00	0.25	0.75	4
P3_80/472	0.67	1.00	0.67	0.33	3
P4_80/456	0.20	1.00	0.33	0.67	3
P5_80/444	0.25	1.00	0.33	0.67	3
P6_80/416	0.00	1.00	0.00	1.00	3
P7_80/476	0.25	1.00	0.33	0.67	3
P8_80/568	0.33	1.00	0.50	0.50	2
P9_80/440	0.20	1.00	0.25	0.75	4
P10_80/592	0.50	1.00	0.67	0.33	3
tot. average_80	0.36	1.00	0.41	0.61	3.45
P1_100/720	0.75	0.50	0.75	0.25	4
P2_100/628	0.60	1.00	0.60	0.40	5
P3_100/648	0.75	0.50	0.75	0.25	4
P4_100/572	0.67	1.00	0.67	0.67	3
P5_100/648	0.33	1.00	0.33	0.67	3
P6_100/516	0.50	1.00	0.50	0.50	4
P7_100/624	0.33	1.00	0.33	0.67	3
P8_100/656	0.00	1.00	0.00	1.00	4
P9_100/704	0.00	1.00	0.00	1.00	2
P10_100/452	0.67	1.00	0.33	0.67	3
tot. average_100	0.46	0.90	0.43	0.61	3.50

approximation. Note that, in the case of UGH, these points correspond to the *lexic1* and *lexic2* individuals defined in Sect. 4.2.

Lastly, column 6 of both tables shows the average number of different solutions of each BRP instance that correspond to the points of the approximate Pareto frontier, whose cardinality is given in column 4. Such solutions are candidates for efficient solutions.

In analysing the last generation (columns 2 and 3 in these two tables), one may see that both algorithms were able to achieve a maximal set of feasible individuals. In fact, the cardinality of the population in UGH is 40 and the utopic was always infeasible, while, for ASP, the cardinality of the population plus the cardinality of the archive is 39.

In the case of UGH, the number of points in the last population is at least the value given by the wave metric, as this metric counts the number of ranks and each rank has one or more points. Therefore, although the last generations present the same number of individuals, these correspond to 11.7 or more different points for the UGH against 9.5 different points for the ASP, in the case of 80-trip instances. In the 100-trip case, on average ASP generated 12.3 different points, whereas UGH showed a wave metric equal to 11.7. Again, this may suggest that the UGH produces a more diverse set of points, because each rank seldom contains only a single point.

From the optimality perspective, the metrics evaluating the approximate Pareto frontier per run (columns 4 to 6 in Tables 1 and 2) show that, on average, the UGH produced frontiers of a larger size, 3.9 against 2.3 of the ASP for the 80-trip instances, and 4.0 against 2.1 for the 100-trip instances. On the other hand, the corresponding sets of candidates for efficient solutions were, on average, larger for ASP. This means that each point of the approximate Pareto frontier corresponds to more solutions, given the ratio of 5.3 (12.3/2.3) solutions per point for the ASP against the ratio of 1.4 (5.6/3.9) for the UGH, in the first set of instances, and the ratio of 6.0 (12.7/2.1) against 1.4 (5.4/4.0) for the second set of instances. The average spread values obtained by the ASP were smaller than the corresponding figures for the UGH (2.4 and 14.3, respectively for ASP and for UGH, in the case of 80-trip instances; 2.1 and 16.4, respectively for ASP and for UGH, in the case of 100-trip instances), thus revealing the UGH's ability to obtain more widely spread out non-exact Pareto frontiers.

Next, in Table 3, columns 2 and 3, a relative metric is used to compare the final approximate Pareto frontiers from the two heuristics. Each of these frontiers is determined by merging the approximations of the Pareto frontier obtained from the ten runs of the respective evolutionary algorithm and determining non-dominated points within the resulting sets.

Column 2 displays the percentage of points in the final approximate Pareto frontier obtained by the UGH that are not dominated by or are equal to points in the corresponding final Pareto frontier approximated by the ASP, whereas column 3 shows the same relative measure for the points obtained by the ASP. For the set of 80-trip instances, one can see that, on average, 36% of UGH's points are non-dominated by those of ASP, while all ASP's points are non-dominated by UGH's, thus revealing an average difference of 64% in favour of ASP. The relative behaviour of the heuristics for the 100-trip instances is similar: on average, 90% for ASP compared to 46% for UGH, also demonstrating a superior relative performance of the ASP of around 44%, though the figure is smaller than the one found for the 80-trip instances. Instances P1_100 and P3_100 are exceptions to this behaviour.

Columns 4 and 5 of Table 3 display another relative metric: the contribution of each heuristic to the 'best' approximation of the Pareto frontier per instance. This frontier was obtained by selecting all non-dominated points among those belonging to the 20 approximate frontiers from the ten runs of both heuristics, thus building the best approximation of the real Pareto frontier. The figures show that the two heuristics are able to effectively contribute to the production of points that are candidates to optimality. Although, on average, the ASP contributes with a larger percentage of points (61% against 41% for the first set of instances and 61% against 43% for the second one—some points were attained by both heuristics), the UGH contributes with

a significant number of potential non-dominated points which the ASP was unable to reach. This reveals that though the UGH seems to perform worse, it was able to explore regions of the objective space unexplored by the ASP. Both heuristics present complementary advantages, thus meaning that they may be used in cooperation rather than in competition.

Finally, column 6 presents the size of the best approximate Pareto frontier for each of the 21 instances. By taking four illustrative instances, P0_80, P6_80, P1_100 and P5_100, Fig. 4 displays the respective approximate final Pareto frontiers obtained for all runs of each heuristic, as well as the best approximation. Graphs P0_80 and P1_100, on the left, display two cases where UGH performs better than ASP. For the graphs on the right, P6_80 and P5_100, the opposite happens, and ASP performs better than UGH. In particular, for instance P6_80, ASP is able to build the best approximation by itself.

As for the computation times, the average executing time of the ten runs per instance was calculated. Although these times were not comparable because the two algorithms ran in different computers, the maximum figures found were six minutes

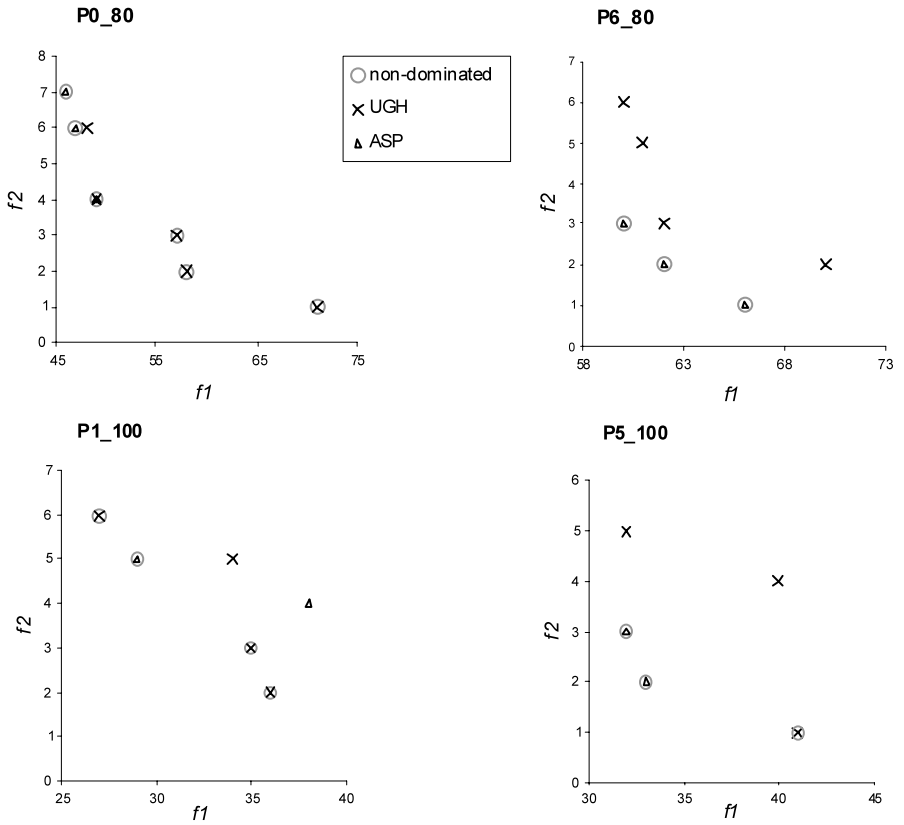

Fig. 4 Illustrations for the approximate Pareto frontiers

for UGH and eight minutes for ASP, far below the two hour limit stipulated by the planning department.

In summarising, both evolutionary approaches achieved large sets of feasible solutions of the BRP, i.e. rosters satisfying all the hard constraints. These can provide the decision-making agents with a diverse set of good alternatives from which they may choose the roster to adopt for the next planning period. The UGH attains a higher diversity within the approximate Pareto frontiers, which is an important ability in the bi-objective context. On the other hand, the ASP attained more solutions that are candidates for efficiency, although they correspond to fewer points in the objective space. Nevertheless, we should bear in mind that the exact Pareto frontier is, as yet, unknown. However, both heuristics can cooperate to approximate it, as they are able to find, in different regions, points that are candidates to be non-dominated, thus helping to build a more accurate approximation.

6 Final comments

This paper presents a model for bus driver rostering that copes with a wide variety of situations within a non-cyclic rostering context. Although non-cyclic approaches have been less common in practical bus driver rostering systems, they are more general, thus allowing for a diversity of scheduling constraints for each driver. Moreover, as the model is of a bi-objective nature it can easily tackle the contradictory interests of the parties involved in the process: the drivers and the administration of the bus transit company. Two bi-objective evolutionary heuristics, UGH and ASP, were designed for the problem addressed. Although they are based on previously devised methods, they had to be carefully adapted to meet the characteristics of the BRP. For instance, as the construction of feasible solutions for the BRP always requires an appropriate procedure, within these evolutionary algorithms an indirect encoding embedding a special purpose constructive heuristic was adopted. The computational experiment described concerns publicly available vehicle scheduling data and the specific case of a bus transit company in Portugal. This experiment enables one to evaluate the proposed heuristics' behaviour and to compare them through performance metrics.

The tests performed on two sets of instances have demonstrated that both evolutionary algorithms can obtain good results while consuming an acceptable amount of CPU time. In fact, both produced large sets of feasible solutions—rosters for the company's bus drivers that comply with all the hard constraints. It should be noted that, in practice, at the bus companies themselves non-violation of hard constraints is seldom attained. Where the objective space is concerned, the algorithms also yielded several solutions with low global violations of soft constraints, expressed by low values for the two objectives of the model. Since, for the moment, the exact Pareto frontier for this bi-objective model is not known, the solutions obtained are only candidates for efficiency.

In fact, small cardinality Pareto frontier approximations were achieved by the heuristics. Nevertheless, even for a single non-dominated point, the bi-objective optimization approach provided a set of equivalent solutions (multiple solutions in the

decision space). This allows the scheduler to consider different choices at once, in detail, thus providing an in-depth knowledge of the solution space. As for the means of supporting decision-making in a real-world context, the leading conclusion is that, in practice, such bi-objective evolutionary heuristic approaches are quite appropriate.

The results did not permit the authors to conclude that one heuristic always performs better than the other. Although ASP is able to find a larger number potentially non-dominated points, it is unable to reach regions of the objective space which are explored by UGH. This reveals that the elitism based on the utopic and lexicographic individuals provides the capability to spread out individuals along the approximate Pareto frontier during the genetic search.

As future research, it is intended to test both heuristics in a broader set of real instances. Moreover, the innovative elitism with a view to improving the SPEA2 based heuristic should be explored, thus combining the relative advantages of both strategies.

One important feature of the proposed approaches concerns the possibility of introducing, removing, or changing the rules used to produce rosters, thus revealing the flexibility whereby they can be adapted to new settings in the company or even to other rostering problems.

Acknowledgements The authors are grateful for the referees' valuable comments and suggestions. This research was partially supported by POCT/MAT/57893/2004 and POCTI/ISFL/152.

References

Aickelin U, Dowsland KA (2004) An indirect genetic algorithm for a nurse scheduling problem. Comput Oper Res 31:761–778

Bianco L, Bielli M, Mingozzi MA, Ricciardelli S, Spadoni M (1992) A heuristic procedure for the crew rostering problem. Eur J Oper Res 58:272–283

Cappanera P, Gallo G (2004) A multicommodity flow approach to the crew rostering problem. Oper Res 52:583–596

Caprara A, Fischetti M, Toth P, Vigo D, Guida PL (1997) Algorithms for railway crew management. Math Program 79:125–141

Caprara A, Fischetti M, Toth P, Vigo D (1998) Modeling and solving the crew rostering problem. Oper Res 46:820–830

Caprara A, Fischetti M, Guida PL, Toth P, Vigo D (1999) Solution to large scale railway crew planning problems: the Italian experience. In: Wilson NHM (ed) Computer aided transit scheduling. Lecture notes in economics and mathematical systems, vol 471. Springer, Berlin, pp 1–18

Carraresi P, Gallo G (1984) A multilevel bottleneck assignment approach to the bus driver's rostering problem. Eur J Oper Res 16:163–173

Catanas F, Paixão J (1995) A new approach for the crew rostering problem. In: Daduna J, Branco I, Paixão J (eds) Computer aided transit scheduling. Lecture notes in economics and mathematical systems, vol 430. Springer, Berlin, pp 267–277

Chu SCK (2007) Generating, scheduling and rostering of shift crew-duties: applications at the Hong Kong international airport. Eur J Oper Res 177:1764–1778

Collette Y, Siarry P (2005) Three new metrics to measure the convergence of metaheuristics towards the Pareto frontier and the aesthetic of a set of solutions in biobjective optimization. Comput Oper Res 32:773–792

CPLEX Manual (version 11) (2007) Using the CPLEX callable library and CPLEX mixed integer library. ILOG INC., Incline Village

Dantzig GB (1954) A comment on Edie's 'Traffic delays at toll booths'. J Oper Res Soc Am 2:339–341

Dornberger R, Frey L, Hanne T (2008) Single and multiobjective optimization of the train staff planning problem using genetic algorithms. In: Evolutionary computation, 2008. Proceedings of CEC 2008. (IEEE world congress on computational intelligence), pp 970–977

Emden-Weinert T, Kotas H-G, Speer U (2001) DISSY—a driver rostering system for public transport. Version 1.11, DISSY project of programme ESPRIT

Ernst A, Jiang H, Krishnamoorthy M, Sier D (2004) Staff scheduling and rostering: a review of applications, methods and models. Eur J Oper Res 153:3–27

Freling R, Lentink RM, Wagelmans APM (2004) A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm. Ann Oper Res 127:203–222

Goldberg D (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading

Hartog A, Huisman D, Abbink EJW, Kroon LG (2009) Decision support for crew rostering at NS. Public Transp 1. doi:10.1007/s12469-009-0009-6

Kohl N, Karisch SE (2004) Airline crew rostering: problem types, modelling, and optimization. Ann Oper Res 127:223–257

Lezaun M, Perez G, Maza ES (2006) Crew rostering problem in a public transport company. J Oper Res Soc 57:1173–1179

Lucic P, Teodorovic D (1999) Simulated annealing for the multi-objective aircrew rostering problem. Transp Res A 33:19–45

Lucic P, Teodorovic D (2007) Metaheuristics approach to the aircrew rostering problem. Ann Oper Res 155:311–338

Mesquita M, Paias A (2008) Set partitioning/covering-based approaches for the integrated vehicle and crew scheduling problem. Comput Oper Res 35:1562–1575

Mesquita M, Moz M, Paias A, Paixão J, Pato MV, Respício A (2008) Solving public transit scheduling problems. Working paper 1-2008. Centro de Investigação Operacional, Universidade de Lisboa, 31 pp (submitted)

Mesquita M, Paias A, Respício A (2009) Branching approaches for integrated vehicle and crew scheduling. Public Transp 1:21–37

Moz M, Pato MV (2007) A genetic algorithm approach to a nurse rerostering problem. Comput Oper Res 34:667–691

Odoni A, Rousseau J-M, Wilson N (1994) Models in urban and air transportation. In: Pollock S, Rothkopf M, Barnett A (eds) Operations research in the public sector. Handbooks in operations research and management science, vol 6. North-Holland, Amsterdam, pp 107–150

Pato MV, Moz M (2008) Solving a bi-objective nurse rerostering problem by using a utopic genetic heuristic. J Heuristics 14:359–374

Pedrosa D, Constantino M (2001) Days-off scheduling in public transport companies. In: Voss S, Daduna J (eds) Computer aided transit scheduling. Lecture notes in economics and mathematical systems, vol 505. Springer, Berlin, pp 215–232

Portugal R, Lourenço HR, Paixão JP (2009) Driver scheduling problem modelling. Public Transp 1. doi:10.1007/s12469-008-0007-0

Silva JDL, Le KN (2008) A simple evolutionary algorithm with self-adaptation for multi-objective nurse scheduling. In: Cotta C, Sevaux M, Sörensen K (eds) Adaptative and multilevel metaheuristics, vol 136. Springer, Berlin, pp 133–155

Silva JDL, Burke EK, Petrovic S (2004) An introduction to multiobjective metaheuristics for scheduling and timetabling. In: Gandibleux X, Sevaux M, Sörensen K, T'kindt V (eds) Metaheuristics for multiobjective optimisation. Springer, Berlin, pp 91–129

Sodhi MS, Norris S (2004) A flexible, fast, and optimal modelling approach applied to crew rostering at London underground. Ann Oper Res 127:259–281

Zitzler E, Laumanns M, Thiele L (2002) SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. In: Giannakoglou K, Tsahalis D, Periaux J, Papailiou K, Fogarty T (eds) Evolutionary methods for design, optimisation and control. CIMNE, Barcelona, pp 95–100