Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling (ICAPS 2020)

Solving Large Real-Life Bus Driver Scheduling Problems with Complex Break Constraints

Lucas Kletzander, Nysret Musliu
Christian Doppler Laboratory for Artificial Intelligence and Optimization for Planning and Scheduling
DBAI, TU Wien, Karlsplatz 13, 1040 Vienna, Austria
{lkletzan, musliu}@dbai.tuwien.ac.at

Abstract

When scheduling drivers for public transport, in addition to covering the demand and dealing with the spatial dimension, a range of legal requirements, collective agreements and company policies need to be respected. The level of concentration required while driving leads to strict rules for break assignments. This results in a complex problem where creating cost-efficient and employee-friendly schedules is challenging. This paper deals with bus driver scheduling using the rules of the Austrian collective agreement for private omnibus providers. The contributions are the formalization of the complex Austrian rules for bus drivers, a new set of publicly available instances based on the characteristics of real-life instances, and a metaheuristic solution approach for the problem. The algorithm was able to significantly improve the solutions of real-life instances and is evaluated on the generated instances. Further we provide insight in the necessity of objectives for employee satisfaction and their effects. Our method can even be successfully applied to improve results on a problem with very different constraints from Brasil.

Introduction

When there is varying demand for employees at different times of the day, it is important to have efficient schedules in order to cover the demand with minimal cost. On the other hand, there is a range of legal requirements, collective agreements and company policies that need to be taken into account to create feasible schedules. Further, not every schedule that is feasible will be readily accepted by the employees, purely optimizing cost might result in reduced employee satisfaction and potential conflicts with labour unions.

An area that is especially restricted by various constraints is scheduling for drivers in public transport. As these employees have a great responsibility keeping their passengers safe, legal requirements enforce strict break assignments in order to maintain concentration. In addition to that a spatial component needs to be considered. This makes the goal to create cost-efficient and employee-friendly schedules even more challenging. This paper deals with optimizing schedules for bus drivers in Austria, using the regulations from the Austrian collective agreement for employees in private omnibus providers serving regional lines.

The contributions of this work are as follows. Based on real-life problems from different cities in Austria we generate a publicly available set of benchmark instances that can be reused for further research. We formalize and optimize the rules from the Austrian collective agreement which are more complex than most other regulations presented in literature so far. We present a metaheuristic based on Simulated Annealing that was able to significantly improve the results in real-life scenarios and evaluate this method on the newly generated instances. We explain the practical relevance of optimizing not just cost, but also objectives for employee satisfaction. Finally, we also include a comparison with work on Brasilian bus driver scheduling, highlighting that our approach can be adapted to driver scheduling problems with completely different objectives and still improve some of the best known results.

Related Work

Due to its high practical relevance, the topic of employee scheduling has seen tremendous research for many years. Several surveys (Ernst et al. 2004; Van den Bergh et al. 2013) provide a good overview of work in different areas. A survey for the different objectives in operating bus transport systems is provided by (Ibarra-Rojas et al. 2015). Driver scheduling is located between vehicle scheduling and driver rostering in a six step process. Driver scheduling belongs to the area of crew scheduling problems (Ernst et al. 2004) that is also frequently applied to airline (Gopalakrishnan and Johnson 2005) and train crew scheduling.

Research on Bus Driver Scheduling Problems started decades ago (Wren and Rousseau 1995). Previous work explored different solution methods. Exact methods mostly use column generation with a set covering or set partitioning master problem and a resource constrained shortest path subproblem (Smith and Wren 1988; Desrochers and Soumis 1989; Portugal, Lourenço, and Paixão 2009; Lin and Hsu 2016). Heuristic methods like greedy (Martello and Toth 1986; De Leone, Festa, and Marchitto 2011; Tóth and Krész 2013) or exhaustive (Chen et al. 2013) search, tabu search (Lourenço, Paixão, and Portugal 2001; Shen and Kwan 2001), genetic algorithms (Lourenço, Paixão, and Portugal 2001; Li and Kwan 2003) are used in different variations. A problem in Brasil (Constantino et al. 2017) uses a method based on iterated assignment problems on published

Table 1: Example bus tour

ℓ	$tour_\ell$	$start_\ell$	end_ℓ	$startPos_\ell$	$endPos_\ell$
1	1	360	395	0	1
2	1	410	455	1	2
3	1	460	502	2	1
4	1	508	540	1	0

benchmarks that we compare with.

Our solution method is based on Simulated Annealing (Kirkpatrick, Gelatt, and Vecchi 1983). We focus on a combined objective that includes aspects for generating practically usable solutions, while most work so far focuses mainly on cost, only sometimes minimizing idle time and vehicle changes (Ibarra-Rojas et al. 2015), (Constantino et al. 2017). Further, break constraints are mostly simple, often just one meal break. However, break scheduling within shifts has been considered by authors in different contexts (Beer et al. 2008; 2010; Widl and Musliu 2014). There is not much work on multi-objective bus driver scheduling (Lourenço, Paixão, and Portugal 2001), but multi-objective approaches are used in other bus operation problems (Respício, Moz, and Vaz Pato 2013).

Problem Description

The Bus Driver Scheduling Problem deals with the assignment of bus drivers to vehicles that already have a predetermined route for one day of operation. The shifts that are generated need to respect a range of constraints regarding length and complex break assignment rules.

The specification presented here is taken from the Austrian collective agreement for employees in private omnibus providers (WKO.at 2019), using the rules for regional lines (up to 50 km per line). In case of ambiguities we use an interpretation from practice.

Problem Input

The bus routes are given as a set of individual bus legs $\mathbf{L}$, each leg $\ell \in \mathbf{L}$ is associated with a tour $tour_\ell$ (corresponding to a particular vehicle), a start time $start_\ell$, an end time end_ℓ, a starting position $startPos_\ell$, and an end position $endPos_\ell$. The amount of time within the leg that is actually spent actively driving is specified as $drive_\ell$. This problem uses $drive_\ell = length_\ell = end_\ell - start_\ell$.

Table 1 shows a short example of one particular bus tour. The vehicle starts at time 360 (6:00 as our time units are minutes) at position 0, which could be the bus depot. 35 minutes later it arrives at position 1. Before the next leg of the bus tour there is a 15 minutes waiting time which might qualify as a break for the employee depending on the constraints explained later. After four legs, the bus returns to the depot at time 540. Valid input never has overlapping bus legs for the same tour and consecutive bus legs i, j of the same tour always respect $endPos_i = startPos_j$.

Further input is a distance matrix, which, for each pair of positions i and j, denotes a time $d_{i,j}$ it takes a driver to get from i to j when not actively driving a bus. If no transfer is possible, we set $d_{i,j} = \infty$. $d_{i,j}$ with $i \neq j$ is called passive ride time. $d_{i,i}$ represents the time it takes to switch tour at the same position, but is not considered passive ride time. We define the occurrence of a tour change as when a driver has an assignment of two consecutive bus legs i and j with $tour_i \neq tour_j$.

Finally, for each position i an amount of working time for starting a shift at that position $startWork_i$ and for ending a shift $endWork_i$ are given. At any depot i preparing the bus ($startWork_i = 15$) and finalizing the bus ($endWork_i = 10$) are considered, for other positions the value is 0.

Solution

A solution to the problem is an assignment of exactly one driver to each bus leg. A feasible solution must satisfy the following criteria:

- No overlapping bus legs are assigned to any driver.
- Whenever tour or position changes for a driver between assigned bus legs i and j, then $start_j \geq end_i + d_{i,j}$.
- Each shift respects all hard constraints regarding work regulations as specified in the next section.

Within the set of feasible solutions, different criteria might be optimized as explained later.

Work and Break Regulations

Valid shifts for drivers are constrained by work regulations and require frequent breaks. First, we need to distinguish different measures of time related to a shift s containing the set of bus legs $\mathbf{L}_s$:

- Driving time $D_s = \sum_{i \in \mathbf{L}_s} drive_i$: The total amount of driving time.
- Total time $T_s = end_{\ell_s} + endWork_{\ell_s} - (start_{f_s} - startWork_{f_s})$, where $f_s = \arg\min_{i \in \mathbf{L}_s}\{start_i\}$ and $\ell_s = \arg\max_{i \in \mathbf{L}_s}\{end_i\}$: Span from the start of work until the end of work for shift s.
- Working time $W_s = T_s - unpaid_s$: Actual working time which does not include certain unpaid breaks.

Corresponding to those definitions there are also different notions of breaks called driving breaks and rest breaks as defined in the remainder of this section. Note that a particular break might qualify as both of those types at the same time or just one of them.

Driving Time Regulations. The maximum driving time is restricted to $D_{max} = 9$ hours. Breaks from driving need to be enforced at the latest after 4 hours of driving time. In case of splitting the break, all parts of the break need to occur before a driving block exceeds the 4 hour limit. Once the required amount of break time is reached, a new driving block starts. The following options are possible:

- One break of at least 30 minutes
- Two breaks of at least 20 minutes each
- Three breaks of at least 15 minutes each

Total Time Regulations. A hard limit $T_{max} = 14$ hours is enforced.

Working Time Regulations. The working time W_s has a maximum of $W_{max} = 10$. The minimum paid working time W_{min} is set to 6.5 hours for full time employees. If the employee is working for a shorter period of time, the difference has to be paid anyway. It is possible for part time employees to set this limit to 3 hours, however, due to limited availability of part time bus drivers in practice we use the limit for full time employees. We define $W'_s = \max\{W_s; 390\}$ to denote the actual paid working time.

A minimum rest break is required according to the following options:

- $W_s < 6$ hours: no rest break
- 6 hours $\leq W_s \leq 9$ hours: at least 30 minutes
- $W_s > 9$ hours: at least 45 minutes

The rest break might be split into one part of at least 30 minutes and one or more parts of at least 15 minutes. The first part of the rest break has to occur after at most 6 hours of work.

A rest break is unpaid as long as it not located within the first 2 or the last 2 hours of the shift. More precisely, the area of a (partial) rest break not located within the first 2 or last 2 hours is unpaid as long as this area itself is at least 15 minutes long.

The maximum amount of unpaid rest is limited:

- If the 30 minute part is located within the first 3 or the last 3 hours of the shift: at most one hour of unpaid rest
- Otherwise: at most 1.5 hours of unpaid rest

Rest breaks beyond this limit are paid. Regarding the 30 minute part, our interpretation is as for the 15 minute breaks, e.g., as long as 30 minutes of a longer rest break are not within the first 3 or last 3 hours, the requirement is met.

Split Shifts. A shift might contain up to two shift splits. Each of them must be at least three hours long, is unpaid and does not count towards W_s. However, such splits are typically regarded badly by the drivers. A break less than 3 hours long is considered a rest break. A shift split resets the driving time (i.e., counts as a driving break), but does not contribute to rest breaks.

Objectives

There are several optimization criteria, setting a different and often conflicting focus on the resulting schedules. Typical minimization objectives regarding operation cost are:

- Number of employees
- Sum of working times $\sum_s W_s$ or $\sum_s W'_s$
- Sum of extra hours

However, there are additional minimization objectives that need to be considered in order to get schedules that are actually workable in practice:

- Sum of total times $\sum_s T_s$
- Sum of passive ride times
- Number of tour changes
- Number of shift splits

While some of these additional objectives correlate with operational objectives (e.g., reducing the number of tour changes typically requires less change time, contributing to the working time objective), others like the number of shift splits actually reduce options like the possibility to use the same driver at two different peak times. However, schedules without such objectives typically are not acceptable in practice as they completely disregard driver needs.

We use a linear combination of several objectives in this work. Note that there is hardly a universal notion which objectives to consider, much less a perfect set of weights. In practice, depending on the scenario, often many different schedules are evaluated to figure out what works best. In this evaluation, we propose the following objective function that stems from practical experience.

$$objective = \sum_s 2 \cdot W'_s + T_s + ride_s + 30 \cdot change_s + 180 \cdot split_s \quad (1)$$

In Equation (1) we use the sum of working times as the main objective, but also the sum of total times to reduce long unpaid periods for employees. The further objectives aim to reduce passive ride time $ride_s$ and the number of tour changes $change_s$ that are beneficial for employees, but also for efficient schedules. The last objective aims to reduce the number of split shifts $split_s$ as they are very unpopular. The weight of 30 for tour changes corresponds to 10 minutes of paid working time (paid working time contributes to W_s and T_s), the weight of 180 for a shift split corresponds to 1 hour of paid working time.

Instances

This section describes the kind of instances that are typically seen in real-life applications and highlights their most relevant features for the optimization. The features were drawn from analysing different real-life scenarios in Austria, looking at local bus lines both in urban and rural areas.

While the original instances cannot be shared in public, an instance generator was created to generate instances with the same characteristics that are shared along with this paper to allow comparison in further work[1]. There are 50 instances, 5 each for 10 different sizes ranging from 10 tours (about 70 legs) to 100 tours (almost 1000 legs).

Instance Size

Real-life instances come in a variety of sizes. This is due to different scenarios that are considered. Small instances are typically composed of a few lines that have one or more common or at least closely located stations. When bus companies have to compete for providing service in an area, they often have to apply with a cost calculation for such line clusters. Typically, the vehicle routes are already predetermined, leaving the personnel calculation to the competitors. Therefore, the small instances reflect such clusters of lines.

On the other hand, some scenarios involve larger bus companies optimizing their whole bus network for a town or city,

[1] https://cdlab-artis.dbai.tuwien.ac.at/papers/sa-bds/

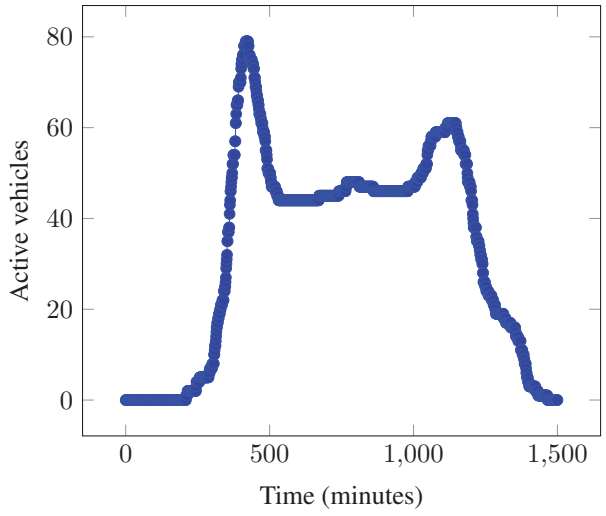

Figure 1: Demand distribution for instance 100_50.

giving rise to very large instances with more than 1000 legs, making the problem very challenging to solve. While the problem is very large and there are many constraints to take into account, in practice still much work is done manually.

Demand Distribution

The instances encountered in real-life scenarios have a very typical shape highlighted in Figure 1. There is a large demand peak in the morning, where both employees and pupils require many buses in short amount of time, while there is a major depression until lunch time. This results in a problematic conflict with the constraints for valid shifts as a minimum work time of 6.5 hours needs to be paid, while after the peak there is not enough demand.

The other parts of the demand distribution are less of a problem for the optimization. There are often a few shifts that start very early and end very late, this might rarely also include night buses running the whole night. We do not include night buses in the generated instances. At lunch time there is often a slight peak, in particular when schools need to be served, further there is a significant peak in the evening, however, far less pronounced compared to the morning peak.

Waiting Times

Figure 2 shows the bus legs for one of the small instances. Depending on the instance, real-life scenarios might include significant waiting times within the tours. This typically occurs when the bus arrives at the final stop of a line and has to wait before starting the next trip. There might be longer waiting times on lines with less demand, especially in rural areas, while some lines have only very short waiting times. E.g., tour 2 in the example repeatedly shows long waiting times, while tour 3 has mostly short waiting times.

The crucial importance of these waiting times is the ability to use them for breaks in the schedule. Lines with waiting times make it very important to consider all break options, as splitting breaks in many parts often allows to place those naturally within the schedule of the line, reducing the need to exchange the driver on a tour.

Therefore, the generated instances exhibit a range of different waiting times that might allow usage as breaks. The evaluation section will present more detail on how that influences optimal schedules for the problem.

Solution Method

This section presents the solution methods used to handle the complex constraints and solve the problem, in particular the representation of the problem-specific constraints, a construction heuristic, and Simulated Annealing including the proposed moves and their selection.

Constraint Representation

The general idea for the representation of the complex constraints is to parse shift candidates from start to end, maintaining accumulators for several resources. These can then be used to check the constraints and, if violated, add penalties to the objective function.

Accumulators are initially assigned to 0. Assume that i and j are two bus legs assigned consecutively to shift s in the step of calculating the values at leg j. We define $diff_{ij} = start_j - end_i$.

Drive time constraints use the following accumulators:

$$dt_j = dt_i + drive_j \tag{2}$$

$$block_j = diff_{ij} \geq 30 \vee (diff_{ij} \geq 20 \wedge b20_i = 1) \vee (diff_{ij} \geq 15 \wedge b15_i = 2) \tag{3}$$

$$dc_j = \begin{cases} drive_j & \text{if } block_j \\ dc_i + drive_j & \text{else} \end{cases} \tag{4}$$

$$b20_j = \begin{cases} 0 & \text{if } block_j \\ 1 & \text{if } diff_{ij} \geq 20 \\ b20_i & \text{else} \end{cases} \tag{5}$$

$$b15_j = \begin{cases} 0 & \text{if } block_j \\ b15_i + 1 & \text{if } diff_{ij} \geq 15 \\ b15_i & \text{else} \end{cases} \tag{6}$$

Equation (2) sums up total drive time, (4) the drive time in the current driving block. The driving block resets depending on (3), formalizing the different splitting options for driving breaks. Equations (5) and (6) keep track of the number of driving breaks of length at least 20 respectively 15 minutes in the current driving block.

The problem constraints are enforced with $dt_i \leq D_{max}$ and $dc_i \leq 240$.

Next we need to specify work time and rest break constraints:

$$diff'_{ij} = \begin{cases} 0 & \text{if } diff_{ij} - r_{ij} \geq 180 \\ diff_{ij} - r_{ij} & \text{else} \end{cases} \tag{7}$$

Equation (7) denotes the length of a potential rest break between legs i and j, using r_{ij} for the passive ride time necessary between legs i and j. In case passive ride time is needed, it is always placed after the rest break and before the start of leg j. $diff''_{ij}$ denotes the potentially unpaid break

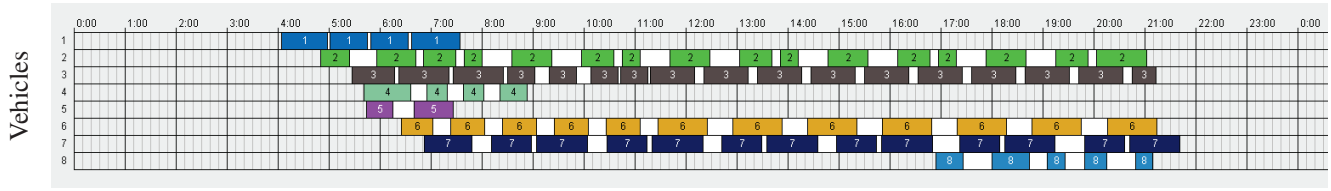

Figure 2: Bus tours for instance 10_1

time by reducing $diff'_{ij}$ by parts located within the first 2 or last 2 hours.

$$wt_j = wt_i + diff'_{ij} + r_{ij} + length_j \tag{8}$$

$$rest_j = rest_i + \begin{cases} diff'_{ij} & \text{if } diff'_{ij} \geq 15 \\ 0 & \text{else} \end{cases} \tag{9}$$

$$first15_j = first15_i \vee (diff'_{ij} \geq 15 \wedge end_i - start'_s \leq 360) \tag{10}$$

$$break30_j = break30_i \vee diff'_{ij} \geq 30 \tag{11}$$

$$unpaid_j = unpaid_i + diff''_{ij} \tag{12}$$

$$center30_j = center30_i \vee \text{isCentered30}(diff_{ij}) \tag{13}$$

Equation (8) sums up the work time, currently still including unpaid rest breaks. Qualified rest breaks, whether unpaid or not, are summed up in equation (9). Equations (10) and (11) are required to monitor split rest breaks, ensuring that the first part occurs no later than 6 hours into the shift (using $start'_s$ as shorthand for the start of the shift) and that there is a break of at least 30 minutes.

Equation (12) sums up potential unpaid rest time. The maximum amount might depend on breaks later in the schedule, therefore the final calculation is done at the end of the shift. Equation (13) keeps track whether there is a centered break of at least 30 minutes (according to the problem definition).

The values from the last bus leg ℓ of shift s are then used as follows:

$$upmax_s = \begin{cases} 90 & \text{if } first15_\ell \wedge break30_\ell \wedge center30_\ell \\ 60 & \text{if } first15_\ell \wedge break30_\ell \wedge \neg center30_\ell \\ 0 & \text{else} \end{cases} \tag{14}$$

$$W_s = wt_\ell + startWork'_s + endWork'_s - \min\{unpaid_\ell; upmax_s\} \tag{15}$$

Equation (14) calculates the maximum rest time that can be unpaid based on having a valid rest time at all and the location of the 30 minute part. Equation (15) calculates the total paid working time. Keep in mind that for the objective W_{min} still needs to be taken into account.

$$\begin{cases} W_s < 360 & \text{if } \neg first15_\ell \vee \neg break30_\ell \\ W_s \leq 540 & \text{if } first15_\ell \wedge break30_\ell \wedge rest_i < 45 \\ W_s \leq 600 & \text{else} \end{cases} \tag{16}$$

Equation (16) bounds the maximum working time based on the rest breaks that are present.

Further we need to check the maximum total time that can be calculated without an accumulator and maintain the following accumulators for the remaining objectives:

$$ride_j = ride_i + r_{ij} \tag{17}$$

$$change_j = change_i + \begin{cases} 1 & \text{if } tour_i \neq tour_j \\ 0 & \text{else} \end{cases} \tag{18}$$

$$split_j = split_i + \begin{cases} 1 & \text{if } diff_{ij} - r_{ij} \geq 180 \\ 0 & \text{else} \end{cases} \tag{19}$$

Construction Heuristic

The first part of the solution mechanism is to construct an initial solution by using a greedy assignment heuristic. While this heuristic is not meant to produce solutions of top quality, it is an important base for the use of Simulated Annealing in the later part of the algorithm.

Algorithm 1: Construction heuristic.

```
for leg in L_unassigned do
    e ← bestEmployee(leg);
    assignLegToEmployee(leg, e);
    while leg ← nextTourLeg(leg) do
        if ¬evaluateAssign(leg, e) then
            break;
        end
        assignLegToEmployee(leg, e);
    end
end
for e in Employees do
    for f in startLater(Employees, e) do
        if reassignLast(e, f) then
            break;
        end
    end
end
```

Listing 1 shows the high-level view of the algorithm. The first loop goes through all bus legs and assigns them to the best employee, identified as the employee where the assignment results in the lowest objective after assignment. Then as many legs from the same tour as possible without violations are assigned to the same employee. These legs are skipped in the outer loop. This saves much effort compared to building combinations for the same tour in later stages of the algorithm.

Table 2: Results for the benchmark dataset

	Construction Heuristic			Simulated Annealing				Hill-climbing			
Instances	Time	Empl.	Value	Time	Empl.	Best	Mean	Time	Empl.	Best	Mean
10	0.2	12.2	15747.2	22.8	11.6	**14717.4**	14740	7.8	12	14904.4	14988
20	0.54	24.2	32627.8	62.2	22.6	**30860.6**	30971	28	22.8	30931.4	31276
30	1.68	41	54141.6	108.8	38.4	**50947.4**	51258	99.4	38.8	51544.2	51917
40	3.1	55	73417	267	52.2	**69119.8**	69380	151.2	52	69533.6	71338
50	6.26	68.2	91372.8	329	66	87013.2	87557	295.4	65.8	**86718.6**	87263
60	10.62	80.8	109293.8	543.6	78.8	103967.6	104333	432.8	79	**103780**	104296
70	18	92.8	130024.2	751.4	90.4	**122753.6**	123226	718.6	90.4	122912.8	123304
80	26.54	107	148889	1140.2	104.6	140482.4	140914	959.4	104	**139765.2**	140508
90	37.62	120.2	165171.6	1453	118	156385	157426	1516.6	117.2	**156239.4**	156863
100	48	130.4	183456.4	1449.4	128.2	173524	174502	1483.2	127.4	**172327.8**	172909

The second loop performs some reassignments from employees with earlier shift to those with later shifts. The reason is that the greedy assignment of legs results in late legs often forming very short, expensive shifts. Therefore, reassigning legs from the end of earlier shifts to the start of later shifts results in a more balanced schedule. Reassignments are only done if they improve the overall objective.

Simulated Annealing

The improvement stage of the algorithm is based on Simulated Annealing, as this is a flexible method that has provided very good results for other scheduling problems.

Algorithm 2: Simulated annealing implementation.

```
1  t ← t_start;
2  changeCount ← 0;
3  while changeCount < maxCount do
4      for j ← 0 to innerIterations do
5          move ← chooseMove();
6          change ← move.evaluate();
7          if acceptMove(change, t) = true then
8              move.execute(solution);
9              solution.value ←
                 solution.value + change;
10             if change < 0 then
11                 changeCount ← 0;
12             end
13         else
14             move.abort();
15         end
16     end
17     changeCount ← changeCount + 1;
18     t ← t · coolingRate;
19 end
```

Listing 2 shows the structure of the algorithm. The moves used for the improvement phase are different kinds of swaps. They apply to a pair of employees and exchange either one or more bus legs in a selected time window between those employees. The selection where to apply happens in `chooseMove`. While the base move of reassigning one bus leg could be applied repeatedly, moving multiple legs at once is critical as it avoids intermediate penalties.

As the larger instances have up to hundreds of employees, it is important to focus the application of moves towards areas that have high objective values. Therefore, with 50% probability the first employee for a move is selected among the 10 employees with the highest objective values, otherwise randomly. The second employee is always selected randomly. The split allows potential improvements to occur everywhere, while the focus allows faster convergence by higher chance of eliminating highly penalized duties.

A move is accepted in `acceptMove`, where $change$ is the change in objective, if it does not make the solution worse or otherwise with probability $\exp\left(-\frac{change}{t}\right)$. The termination criterion is $maxCount$ iterations without improvement, ensuring the algorithm settles to a stable solution.

Evaluation

This section presents the evaluation of our method. First it presents the setup of the experiments, then it provides the results for the new benchmark instances. The results are compared to using just paid work time as the objective, highlighting the importance of additional objectives. Finally, we provide a comparison with publicly available instances from Brasil, improving some best known solutions.

Experimental Setup

The experiments are run on a PC with an Intel Core i7-7500U at 2.7 GHz. The algorithm receives a maximum number of employees that it is allowed to use and typically settles on a good number due to the working time objective. However, the algorithm can often find slightly better solutions when restricting the employees even further, therefore, several exploration runs decreasing the employee limit are performed per instance.

Based on experiments we set the parameters for the algorithm as follows. Hard constraint violations are penalized by a cost of 1000 per minute of violation. We report the results from the construction heuristic and compare Simulated Annealing with randomized hill climbing (temperature set to 0). The starting temperature is set to 100, the exploration runs use a cooling rate of 0.9 and 10 as the number of itera-

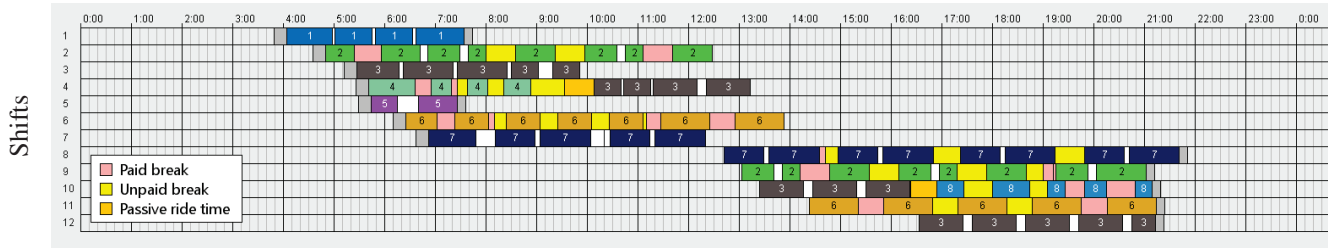

Figure 3: Best solution for instance 10_1 using objective (1)

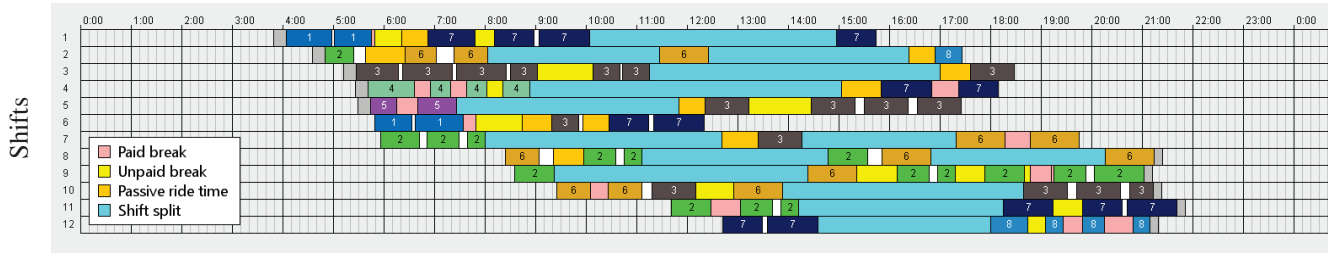

Figure 4: Best solution for instance 10_1 using only working time as objective

tions without change. More precise runs use a cooling rate of 0.99 and 100 iterations without change as they trade speed for result quality. The number of inner iterations (moves per temperature level) is set to 1000. We repeat the best configuration three times to get an understanding of the variation introduced by random decisions in the algorithm.

Results

Table 2 shows the results for the 50 benchmark instances, for brevity reasons using the average over each size category of instances. Runtimes are in seconds. The construction heuristic produces fast results of reasonable quality. In comparison, the best obtained results are typically 5 to 7 percent better at the cost of higher runtime. Greedy strategies work well as taking the best option to finish the current employee's schedule yields a range of very good schedules. However, in the end there are typically several parts that do not fit together, leading to the need for more sophisticated improvement methods.

After the construction heuristic, a few fast runs are performed to determine if better results can be obtained with lower employee limits. The results show that the best found solutions use only a few employees less than the original number. Runtime and number of employees for Simulated Annealing and hill-climbing are reported from the run producing the best result, the mean result from three runs with equal configuration per instance.

In comparison, Simulated Annealing and hill-climbing produce similar results. While differences are not significant, a tendency shows better results for Simulated Annealing on smaller instances and better performance of hill-climbing on larger instance. The most likely cause is that Simulated Annealing helps to avoid local optima in smaller instances while in larger instances the search space is so large that the effort of repairing deteriorating moves is higher than the benefit from escaping local optima. Further, hill-climbing is faster on smaller instances, but looses the advantage on larger instances.

Detailed Solution Analysis

In order to understand how the characteristic features of the instances influence the results, Figure 3 shows the best found solution for instance 10_1 using objective (1). Note that this instance is chosen to illustrate observations that are found across the whole set of instances.

The high peak in the morning typically leads to several short shifts in the morning, despite the minimum of at least 6.5 hours. Some of them might be combined with later parts of the schedule using shift splits, however, as their use is discouraged, at least some short shifts will usually remain.

The shifts use the waiting time available on the tours to fill their breaks. Therefore, most long tours only need to be handed over once. Note that tour 3 does not have enough breaks, therefore it needs to be handed over more frequently. The algorithm combines it with short tours in the morning and evening in order to utilize employees well, or generates shorter shifts.

When talking about objectives for optimization, the primary motivation for bus companies is to reduce cost. However, we are using the combined objective (1) that includes several aspects that create better schedules for employees. While on first sight this narrows the possibilities to reduce the cost as much as possible, in practice reasonable schedules for employees not only benefit them, but can also contribute to a better work atmosphere that might lead to lower personnel fluctuations and fewer sick leaves.

In order to understand how tremendous the differences in generated schedules can be, Figure 4 shows the best found

Table 3: Objective values of the different solutions

	Working time	Span	Passive ride	Tour changes	Shift splits
Combined	4840	4611	66	2	0
Working time	4680	8470	375	19	13
Change	-3.3%	$+83.7\%$	$+468.2\%$	$+850\%$	$+\infty\%$

Table 4: Comparison on Brasil instances

	CC130	CC251	CV412	CC512	CC761	CC1000	CC1253	CC1517	CC2010	CC2313
GrBDSP	**8389.4**	17600	**29512.5**	35105	**47532.9**	**64873.6**	82842.9	99852.8	**128964.2**	**147215**
SA	8432	**17009.7**	30395.2	**33743**	48862.8	65015.6	**82332.6**	**99439**	130410	150215.5

solution for instance 10_1 when only using paid working time (W'_s) as the objective.

Both schedules are feasible according to all constraints and represent the best found solutions for their respective objective. However, it would be very hard to convince any operating company to adopt the schedule from Figure 4. Here, almost every employee has at least one shift split, as these do not add any cost to the objective and are therefore easily used to combine bus legs at different times of the day.

Table 3 compares the results for the objective values of the different solutions. The paid working time could indeed be reduced by 3.3% percent using only the working time objective. However, this is in a strong contrast to an increase in total span by 83.7% and even more extreme increases in the remaining objective values.

When comparing all 50 instances, the average reduction in work time is very considerable with 7.9%. However, the average increase is 60.9% in total span, 209% in passive ride time, 219% in tour changes and 1060% in shift splits.

This comparison clearly shows why it is very important to use a balanced set of objectives that actually produce solutions that are well suited to be used in practice.

Comparison

As the randomized search procedure can not prove optimality, we provide several arguments for the quality of the solutions. First, our aim is to compare to exact methods, however, direct CP formulations cannot even solve reduced versions of the problem. Therefore, techniques using column generation are under investigation, however, also leading to complex subproblems. So far, we were able to obtain exact results for size 10 and lower bounds from a linear relaxation for sizes 20 and 30. For size 10 our method reaches the optimum in 4 out of 5 instances, the mean is 0.2% above the optimum (HC 1.9%). For size 20 the mean of SA is 2.5% above the lower bound (HC 3.5%), for size 30 the mean of SA is 3.4% above the lower bound (HC 4.7%).

Second, we were able to successfully improve schedules for a large-scale real-life application with slightly different constraints using more than 2700 legs by about 4% compared to their previous schedule, which was incrementally optimized by a human expert over a long period of time. As personal cost is typically the highest cost factor, this directly results in savings for the company while maintaining the same or even slightly better level of service and employee satisfaction. We also successfully applied the method to several smaller problems for clusters of lines in a major city.

Comparing with other work in the area is difficult for this problem as most often each work uses a different set of constraints, and applies their method on their own set of instances that are not publicly available. However, for a problem from Brasil (Constantino et al. 2017), where instances with more than 2300 bus legs are available online, we are able to compare our approach to their work.

However, their constraints are different from the ones we use. There are no driving breaks and only at most one rest break as well as no passive ride times. There is a minimum working time like in our problem, but also overtime cost as soon as the time is above the minimum, leading to a much more pronounced balancing objective in this problem. As we developed our solution method with potential application to similar problems in mind, we only need to exchange the part for the constraint representation and can then directly apply our method. Additionally we added a term to the objective function penalizing a combination of paid time between legs and overtime for speeding up convergence

Table 4 shows the comparison of our method SA with their method GraphBDSP (Constantino et al. 2017). We can improve 4 out of 10 benchmark instances with our method, showing that it is applicable to problems with very different constraints and still able to provide good results.

Conclusion

In this paper, we presented the bus driver scheduling problem using the complex set of rules in the Austrian collective agreement. We analysed characteristic features of real-life instances, in particular the demand distribution and waiting times and discussed their implications on scheduling. We provided new publicly available benchmark instances for further research and presented a solution method based on a construction heuristic and Simulated Annealing. We successfully applied these methods to the new benchmark instances and discussed the importance of a well-designed objective function. We provided insights in the successful application in real-life scenarios and compared with a problem from Brasil, where we could improve several benchmark instances. Further work will focus on improving exact solutions and lower bounds by investigating more sophisticated methods based on column generation.

Acknowledgements

The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

References

Beer, A.; Gaertner, J.; Musliu, N.; Schafhauser, W.; and Slany, W. 2008. Scheduling Breaks in Shift Plans for Call Centers. In *Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling*, 1–17.

Beer, A.; Gärtner, J.; Musliu, N.; Schafhauser, W.; and Slany, W. 2010. An AI-Based Break-Scheduling System for Supervisory Personnel. *IEEE Intelligent Systems* 25(2):60–73.

Chen, S.; Shen, Y.; Su, X.; and Chen, H. 2013. A Crew Scheduling with Chinese Meal Break Rules. *Journal of Transportation Systems Engineering and Information Technology* 13(2):90–95.

Constantino, A. A.; de Mendonça Neto, C. F. X.; de Araujo, S. A.; Landa-Silva, D.; Calvi, R.; and dos Santos, A. F. 2017. Solving a large real-world bus driver scheduling problem with a multi-assignment based heuristic algorithm. *Journal of Universal Computer Science* 23(5):479–504.

De Leone, R.; Festa, P.; and Marchitto, E. 2011. A Bus Driver Scheduling Problem: a new mathematical model and a GRASP approximate solution. *Journal of Heuristics* 17(4):441–466.

Desrochers, M., and Soumis, F. 1989. A Column Generation Approach to the Urban Transit Crew Scheduling Problem. *Transportation Science* 23(1):1–13.

Ernst, A.; Jiang, H.; Krishnamoorthy, M.; and Sier, D. 2004. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research* 153(1):3–27.

Gopalakrishnan, B., and Johnson, E. L. 2005. Airline Crew Scheduling: State-of-the-Art. *Annals of Operations Research* 140(1):305–337.

Ibarra-Rojas, O.; Delgado, F.; Giesen, R.; and Muñoz, J. 2015. Planning, operation, and control of bus transport systems: A literature review. *Transportation Research Part B: Methodological* 77:38–75.

Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P. 1983. Optimization by Simulated Annealing. *Science* 220(4598):671–680.

Li, J., and Kwan, R. S. 2003. A fuzzy genetic algorithm for driver scheduling. *European Journal of Operational Research* 147(2):334–344.

Lin, D.-Y., and Hsu, C.-L. 2016. A column generation algorithm for the bus driver scheduling problem: Bus Driver Scheduling Problem. *Journal of Advanced Transportation* 50(8):1598–1615.

Lourenço, H. R.; Paixão, J. P.; and Portugal, R. 2001. Multiobjective Metaheuristics for the Bus Driver Scheduling Problem. *Transportation Science* 35(3):331–343.

Martello, S., and Toth, P. 1986. A heuristic approach to the bus driver scheduling problem. *European Journal of Operational Research* 24(1):106–117.

Portugal, R.; Lourenço, H. R.; and Paixão, J. P. 2009. Driver scheduling problem modelling. *Public Transport* 1(2):103–120.

Respício, A.; Moz, M.; and Vaz Pato, M. 2013. Enhanced genetic algorithms for a bi-objective bus driver rostering problem: a computational study. *International Transactions in Operational Research* 20(4):443–470.

Shen, Y., and Kwan, R. S. K. 2001. Tabu Search for Driver Scheduling. In Fandel, G.; Trockel, W.; Aliprantis, C. D.; Kovenock, D.; Voß, S.; and Daduna, J. R., eds., *Computer-Aided Scheduling of Public Transport*, volume 505. Berlin, Heidelberg: Springer Berlin Heidelberg. 121–135.

Smith, B. M., and Wren, A. 1988. A bus crew scheduling system using a set covering formulation. *Transportation Research Part A: General* 22(2):97–108.

Tóth, A., and Krész, M. 2013. An efficient solution approach for real-world driver scheduling problems in urban bus transportation. *Central European Journal of Operations Research* 21(S1):75–94.

Van den Bergh, J.; Beliën, J.; De Bruecker, P.; Demeulemeester, E.; and De Boeck, L. 2013. Personnel scheduling: A literature review. *European Journal of Operational Research* 226(3):367–385.

Widl, M., and Musliu, N. 2014. The break scheduling problem: complexity results and practical algorithms. *Memetic Computing* 6(2):97–112.

WKO.at. 2019. Kollektivvertrag für Dienstnehmer in privaten Autobusbetrieben gültig ab 1.1.2019. https://www.wko.at/service/kollektivvertrag/kv-private-autobusbetriebe-2019.html. Accessed 8 Jul. 2019.

Wren, A., and Rousseau, J.-M. 1995. Bus Driver Scheduling – An Overview. In Fandel, G.; Trockel, W.; Daduna, J. R.; Branco, I.; and Paixão, J. M. P., eds., *Computer-Aided Transit Scheduling*, volume 430. Berlin, Heidelberg: Springer Berlin Heidelberg. 173–187.