Driver Rostering for Bus Transit Companies

K. Nurmi, J. Kyngäs and G. Post

Abstract— **Staff scheduling has become increasingly important for both the public sector and private companies. Good rosters have many benefits for an organization, such as lower costs, more effective utilization of resources and fairer workloads and distribution of shifts. The process of constructing optimized work timetables for the personnel is an extremely demanding task. Driver rostering, preceded by vehicle scheduling and driver scheduling, is the last phase in the bus transit scheduling process. This paper presents a successful way to schedule days-off on a yearly basis and shifts on a monthly basis in one of the Finnish bus transportation companies. The days-off and shifts are scheduled using an algorithm that includes features from population-based methods, simulated annealing, tabu search and ejection-chains. The generated software has been integrated into a third-party vendor product.**

Index Terms—**Real-World Scheduling, Staff Scheduling, Driver Rostering, Bus Transit Scheduling.**

I. Introduction

Staff scheduling is a difficult and time consuming problem that every company or institution that has employees working on shifts or on irregular working days must solve. The staff scheduling problem has a fairly broad definition. Most of the studies focus on assigning employees to shifts, determining working days and rest days or constructing flexible shifts and their starting times. Different variations of the problem are NP-hard and NP-complete [1]-[7] and thus extremely hard to solve. The first mathematical formulation of the problem based on a generalized set covering model was proposed by Dantzig [8]. Good overviews of staff scheduling are published by Alfares [9], Ernst et al. [10] and Meisels and Schaerf [11].

Many of the staff scheduling cases concern nurse rostering, see e.g. [12]-[19]. Other successful application areas include airline crews [20], call centers [21], check-in counters [22], ground crews [23], nursing homes and airport ground services [24], postal services [25] and transport companies [26]. Recent successful algorithms for staff scheduling include ant colony optimization [27], dynamic programming [28], constraint programming [29], genetic algorithms [30], scatter search [31], hyperheuristics [32], integer programming [33], metaheuristics [34], simulated annealing [35], tabu search [36] and variable neighborhood search [37].

There are basically four reasons for the increased interest in real-world staff scheduling. First, public institutions and private companies around the world have become more aware of the possibilities of decision support technologies, and they no longer want to handle the schedules manually. Second, human resources are one of the most critical and most expensive resources for these organizations. Careful planning can lead to significant improvements in productivity. Third, good schedules are very important for the welfare of the staff. Besides increasing employee satisfaction, effective labor scheduling can also improve customer satisfaction. Finally, new algorithms have been developed to tackle previously intractable problem instances, and, at the same time, computer power has increased to such a level that researchers are able to solve real-world problems. One further significant benefit of automating the scheduling process is the considerable amount of time saved by the administrative staff involved.

The purpose of this paper is to sequentially solve the days-off scheduling problem and the shift scheduling problem as it occurs in one of the Finnish bus transit companies. In Section II we define the staff scheduling problem and present the necessary terminology. Section III gives an outline of the overall scheduling process in bus transit companies and details the requirements and preferences of the staff scheduling problem. Our solution method is discussed in Section IV. Although there is a clear tendency to use integer and constrained programming models, our algorithm uses a mixture of evolutionary and local search methods. The algorithm is a variation of cooperative local search. In Section V we present and solve a scheduling problem in one of the Finnish bus transportation companies. It will be seen that our approach produces excellent results.

II. Staff Scheduling

Staff scheduling consists of assigning *employees* to tasks and shifts over a period of time according to a given timetable. The *planning horizon* is the time interval over which the employees have to be scheduled. Each employee has a total working time that he/she has to work during the planning horizon. Furthermore, each employee has competences (qualifications and skills) that enable him/her to carry out certain *tasks*. Days are divided into working days (*days-on*) and *rest days* (days-off). A sequence of working days with one shift each day is called a *work stretch*. Each day is divided into periods or *timeslots*. A timeslot is the smallest unit of time and the length of a timeslot determines the granularity of the schedule. A *shift* is a contiguous set of working hours and is defined by a day

K. Nurmi is with the Satakunta University of Applied Sciences, Pori, Finland (phone: +358 44 710 3371; fax: +358 2 620 3030; e-mail: cimmo.nurmi@samk.fi).

J. Kyngäs is with the Satakunta University of Applied Sciences, Pori, Finland (e-mail: jari.kyngas@samk.fi).

G. Post is with the University of Twente, Department of Applied Mathematics, Faculty of EEMCS, Twente, The Netherlands (e-mail: g.f.post@ewi.utwente.nl).

and a starting period on that day along with a *shift length* (the number of occupied periods). Shifts are usually grouped in *shift types*, for example morning, day and night shifts. Each shift is composed of a number of tasks. A shift or a task may require the employee assigned to it to possess one or more *competences*. A specific sequence of shifts for an employee is called a *stint*. A work schedule for an employee over the planning horizon is called a *roster*. A roster is a combination of shifts and days-off assignments that covers a fixed period of time.

Cyclic schedules are such that all employees have the same basic schedule but start with a different day. In cyclic scheduling the goal is to find a schedule that is optimal for all employees. *Non-cyclic* schedules are individual schedules. In non-cyclic scheduling the goal is to find rosters that fulfill the most requests of the employees. *Continuous* schedules arise in organizations that operate 24 hours a day and seven days a week, otherwise a schedule is called *discontinuous*.

Table 1 shows a solution for a simple one-week staff scheduling problem with seven employees, two shifts (early and late) in a working day and one of three tasks to be completed within a shift. Moreover, task 1 and 2 cannot be carried out by Fay and Gaby, a late shift cannot be followed by an early shift on the next day, and each employee should have one day-off.

Table I
An example of a staff scheduling solution.

		Andy	Fay	Bill	Carl	Gaby	Dan	Erik
Mon	Early	1	3		2			
	Late			3			2	1
Tue	Early	1			2	3		
	Late		3	1				2
Wed	Early	1				3	2	
	Late		3		1			2
Thu	Early	1		2		3		
	Late		3				2	1
Fri	Early			2	1	3		
	Late	1					3	2
Sat	Early		3	1	2			
	Late					3	2	1
Sun	Early	1	3	2				
	Late				2	3	1	

III. Process and Model

We classify the scheduling process in bus transit companies in six phases, as given in Figure 1. In real-world scheduling scenarios, vehicle scheduling, driver scheduling, days-off scheduling and shift scheduling are all extremely hard combinatorial problems of their own.

Bus routing or *Line planning* is a preliminary phase in the development of bus service operations. In public transport the bus routes and their frequencies are defined by the city and the bus companies usually have little opportunity to influence them. Private transport operators create the bus routes based on the business opportunities. In both the public and private sectors, it is completely up to the companies to schedule their fleet of buses, roster the drivers and decide the days-off and working shifts of their drivers. An early reference on bus routing is [38].

Vehicle scheduling consists of scheduling a fleet of vehicles to cover the set of bus routes at minimum cost. The problem is solved for each day of the given time horizon separately, and the solution is a set of vehicle schedules. The vehicles are grouped into depots according to their location. A depot is a parking garage with limited space for buses to stay overnight. The vehicles are grouped also by vehicle types, such as standard, low-floor, kneel bus etc.

The basic multiple depot vehicle scheduling problem (MDVSP) is to construct vehicle schedules using a minimum number of vehicles in such a way, that each trip is assigned to exactly one vehicle schedule. A trip is represented by its departure time and arrival time, stations, and its length. A vehicle schedule is a chain of trips served by the vehicle. Moreover, the vehicle starts and ends in one of the given depots. Each vehicle schedule represents a daily work of a vehicle. A secondary objective of MDVSP is to minimize the total length of the vehicle travels.

The vehicle scheduling problem was initially introduced by Dantzig and Ramser [39] as the truck dispatching problem. The problem has been proven to be NP-hard [38]. Good overviews of vehicle scheduling can be found in [40] and [41].

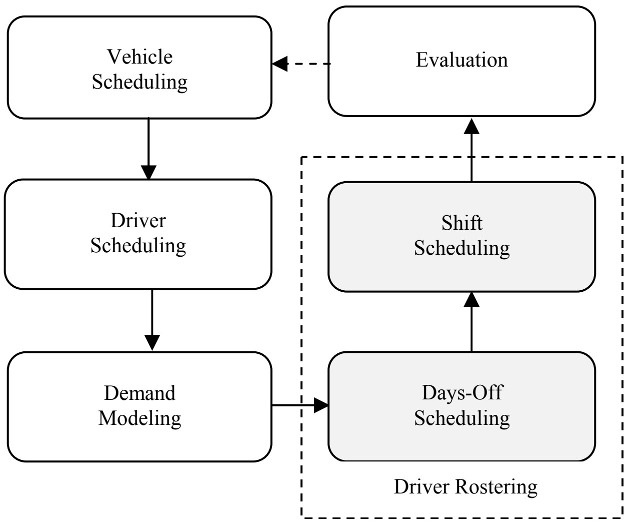

Fig. 1. The scheduling process in bus transit companies.

The goal in *driver scheduling* is to partition the vehicle schedules into operational tasks and to define the sequences of these tasks as shifts. Every task must be assigned to a shift while minimizing the cost in such a way that the daily rules are respected. A task is defined as a sequence of trips on one vehicle without a break that can be performed by a single driver without interruption. The construction of shifts is limited by a maximum total driving time, a maximum number of working hours, a maximum time period spent driving without a break, the number and length of lunch and short breaks in a scheduled time-window, etc. The measure of efficiency may be the total number of shifts used or the total cost in paid hours or a combination of both.

Driver scheduling can be modeled as a set covering problem, which is NP-hard [1]. Among the few papers on driver scheduling we mention [42] and [43].

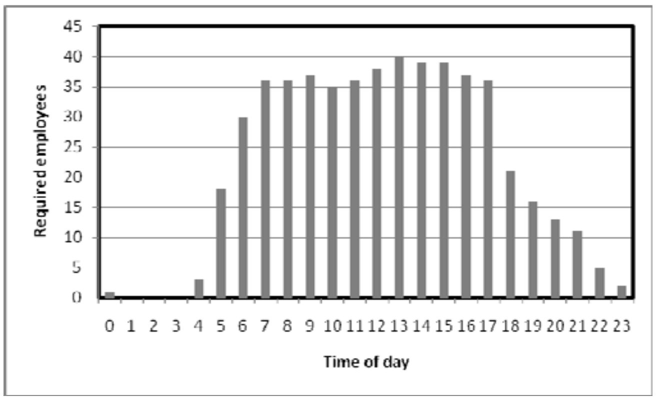

Fig. 2. An example of staffing levels in a bus transit company.

Demand modeling is the process of determining the staffing levels, that is, how many drivers are needed at different times over some planning horizon. Demand modeling includes determination of planning horizons, shift structure, competence structures, regulatory requirements and other constraints. Demand modeling requires labor forecasting as well as hiring and budgeting decisions. The output of the demand modeling is a mathematical model of the staff scheduling problem at hand. In bus transit scheduling, the driver scheduling phase determines the staffing levels. Figure 2 shows an example of staffing levels in a bus transit company.

The staff scheduling phase of the transit scheduling process is called *driver rostering.* The goal is to assign drivers to the constructed shifts over a planning horizon. Driver rostering consists of *days-off scheduling* and *shift scheduling*. Days-off scheduling deals with the assignment of rest days between working days to drivers over a given planning horizon. Shift scheduling deals with the assignment of drivers to shifts. It can also specify the starting time and duration of shifts for a given day. In other words, days-off scheduling deals with working days and shift scheduling deals with the working times of day. When days-off and shifts are scheduled simultaneously, the process is sometimes called *tour scheduling*. For example, scheduling days-off every tenth week and shifts every second week, enables the staff to plan their free time more conveniently. However, scheduling both at the same time every second week increases the probability of meeting the staffing levels without the need to hire part-time drivers.

Finally, participation in *evaluation* ranges from the individual driver through personnel managers to executives. A reporting tool should provide performance measures in such a way that the personnel managers can easily evaluate both the realized staffing levels and the staff satisfaction. When necessary, the vehicle scheduling, driver scheduling and demand modeling can be reprocessed and focused, and the driver rostering process restarted.

It is apparent that a profound understanding of the relevant requests and requirements presented by customers is a prerequisite for implementing a real-world driver rostering problem. The implementation should present a wide variety of real-world constraints and be tractable enough to enable addition of new constraints. We are aware that it is difficult to incorporate the experience and expertise of the personnel managers into a driver rostering system. Personnel managers often have extremely valuable knowledge, experience, and detailed understanding of their specific staffing problem, which will vary from company to company. To formalize this knowledge into constraints is not an easy task. Still, we believe that the model given in this section is applicable in bus transit scheduling scenarios. The model is based on the framework for implementation-oriented staff scheduling given in [44].

The schedules in the model are non-cyclic and can be either continuous or discontinuous. Drivers' total working time may vary depending on the driver. Each shift has a fixed start and end time (for example 07:15 – 16:30). Shifts may vary in length and they overlap. Shifts can be classified by shift types, which can be used to balance the working times at different times of the day between the drivers. Each driver has competences that enable him/her to carry out certain shifts.

In most cases the most important goal is to minimize understaffing and overstaffing. Low-quality rosters can lead either to an undersupply of drivers with a need to hire part-time drivers, or an oversupply of drivers with too much idle time, both implicating a loss of business. The overall objective is to meet daily staffing requirements at minimum penalty without violating work contracts and government regulations.

We next give an outline of the typical constraints of the driver rostering problem. The hard and soft constraints of the problem vary somewhat depending on the problem instance at hand. However, in most cases the hard constraints consist of coverage, regulatory and operational requirements and the soft constraints consist of operational and personal preferences. The coverage requirements ensure that there are a sufficient number of drivers on duty at all times. The regulatory requirements ensure that the driver's work contract and government regulations are respected. The personnel's requests are very important and should be met as far as possible; this leads to greater staff satisfaction and commitment, and reduces staff turnover. A bus transit company can use a mixture of the following requirements and preferences as a framework for its driver rostering generation:

Coverage requirements

(C1) A driver cannot be assigned to overlapping shifts.
(C2) A minimum number of drivers of particular competences must be guaranteed for each shift or each timeslot
(C3) A balanced number of surplus drivers must be guaranteed in each working day

Regulatory requirements

(R1) The required number of working days, working hours and days-off within a timeframe must be respected
(R2) The required number of holidays within a timeframe must be respected
(R3) The required number of free weekends (both Saturday and Sunday free) within a timeframe must be respected
(R4) The required number of special shifts (such as union steward duties) for particular drivers within a timeframe must be respected

(R5) Drivers cannot work consecutively for more than k days (the maximum length of a work stretch)
(R6) Some drivers cannot work on weekends or during specific hours of the day

Operational requirements

(O1) A driver can only be assigned to a shift he/she has competence for
(O2) At least k working days must be assigned between two separate days-off
(O3) A driver cannot be assigned to more than k weekend days within a timeframe
(O4) A driver assigned to a shift type t_1 must not be assigned to a shift type t_2 on the following day (certain stints are not allowed)
(O5) An employee must be assigned to a particular shift or off-duty on a particular day or during a particular timeslot

Operational preferences

(E1) Single days-off should be avoided
(E2) Single working days should be avoided
(E3) The maximum length of consecutive days-off is k
(E4) A balanced assignment of single days-off and single working days must be guaranteed between the drivers
(E5) A balanced assignment of different shift types must be guaranteed between the drivers
(E6) A balanced assignment of weekdays must be guaranteed between drivers

Personal preferences

(P1) Assign or avoid assigning given drivers to the same shifts
(P2) Assign a requested day-on or avoid a requested day-off
(P3) Assign or avoid a given shift type before or after a free period (days-off, vacation).

Generally, a driver cannot be assigned to more than one shift per day. The definition of constraint C1 allows two or more shifts to be assigned provided they do not overlap. Instead of using shifts in C2, the minimum number of on-duty staff can be assigned to timeslots. Quite often a company has more drivers working than is needed to cover the minimum number of drivers each working day. The surplus drivers are used to cover the expected sick leaves and other no-shows (see constraint C3). Constraints R1-R6 can be different from one driver to another, based on the employee's contract.

IV. The Solution Method

Our driver rostering algorithm is a population-based local search method. As we know, the main difficulty for a local search is

1) to explore promising areas in the search space to a sufficient extent, while at the same time,
2) to avoid staying stuck in these areas too long,
3) to escape from these local optima in a systematic way.

The heart of the algorithm is based on ideas similar to the Lin-Kernighan procedures [45] and ejection chains [46]. The basic hill-climbing step is extended to generate a sequence of moves in one step, leading from one solution candidate to another. Our main heuristic operator is the greedy hill-climbing mutation (GHCM). A recent description of GHCM can be found in [47] and a very detailed description in [48].

The GHCM operator moves an object, o_1, from its old position, p_1, to a new position, p_2, and then moves another object, o_2, from position p_2 to a new position, p_3, and so on, ending up with a sequence of moves. In shift scheduling, each position corresponds to a day, and an object is a shift.

The initial solution is created by setting the shifts to random days. The starting shift (shift 1) for the GHCM operator is selected randomly. The new day to which shift 1 is moved is selected considering all possible days on the time horizon and selecting the day that causes the least increase in the objective function when considering just the relocation cost. Then, shift 2 in the new day is selected such that the removal cost of shift 2 (after adding shift 1) causes the highest decrease in the objective function. Next, a new day is selected for shift 2, and so on. The sequence of moves stops if the last move causes an increase in the objective function value and the value is larger than that of the previous non-improving move, or if the maximum length (set to 10) is reached. In those cases a new starting shift for the GHCM operator is selected.

The population-based method uses a population of solutions in each iteration. Population-based methods are good to escape from local optima. Our algorithm is similar to the cooperative local search introduced by Preux and Talbi [49]. In the cooperative local search scheme, each individual carries out its own local search, in our case the GHCM operator. When the operator gets stuck it asks for the cooperation of the population in order to find a direction to move in and continues the search from another point in the solution space. The results in each individual may be different at different times and this encourages diversity within the population. We use a population size of 20.

Our algorithm introduces several mechanisms to avoid staying stuck in a not-so-promising search area for too long. We have implemented tabu search and simulated annealing as part of the GHCM operator. Even though they work well, they are not sufficient to escape from local optima. After being stuck for a given number of iterations, we shuffle the current solution, that is, we allow worse solutions to replace better ones in the current population. We use the five simple shuffling operators described in [50].

The reproduction operation of the algorithm is, to a certain extent, based on the steady-state reproduction: the new schedule replaces the old one if it has a better or equal objective function value. Furthermore, the least fit is replaced with the best one when n better schedules have been found, where n is the size of the population. The pseudo code of the algorithm is given in Figure 3.

```
Set the time limit t, no_change limit m and the population size n
Generate initial population of schedules by randomly
assigning shifts to days
Set no_change = 0 and better_found = 0
WHILE elapsed-time < t
    REPEAT n times
        Select a schedule S by using a marriage selection
        Apply GHCM to S to get a new schedule S'
        Calculate the change Δ in fitness value
        IF Δ < = 0 THEN
            Replace S with S'
            IF Δ < 0 THEN
                better_found = better_found + 1
                no_change = 0
            ENDIF
        ELSE
            no_change = no_change + 1
        ENDIF
    ENDREPEAT
    IF better_found > n THEN
        Replace the worst schedule with the best schedule
        Set better_found = 0
    ENDIF
    IF no_change > m THEN
        Apply shuffling operators
        Set no_change = 0
    ENDIF
    Update the dynamic weights of the hard constraints (ADAGEN)
ENDWHILE
Choose the best schedule from the population
```

Fig. 3. The pseudo code of the driver rostering algorithm.

The traditional penalty method assigns positive weights (penalties) to the soft constraints and sums the violation scores to the hard constraint values to get a single value to be optimized. In our research we use the ADAGEN method [48] which assigns dynamic weights to the hard constraints.

A very detailed description of the algorithm is given in [48]. The parameters of the algorithm (maximum length of the mutation sequence and population size) are those that were found to work best in [51].

Many researchers have argued that the quality of the solution (strongly) depends on the initial solution. However, the initial solution does not have to be good. On the contrary, it has been argued that very good initial solutions are hard to improve on with the method at hand. We note here, that our test runs have shown that our algorithm works best with a random initial population.

V. The Problem in a Finnish Transit Company

In our previous studies we have successfully scheduled the Finnish major ice hockey league [52] and the Finnish 1st division ice hockey league [50]. When the CEO of Turku Transport Services Ltd. heard that we had scheduled these leagues, he contacted us. Turku Transport Services Ltd. is a bus transportation company in the City of Turku. They currently have 58 full-time, eight part-time and four retired-but-still-active bus drivers. Two part-time drivers count as one full-time driver.

Prior to the year 2010, rosters for bus drivers were produced by a cyclic shift scheduling software that was quite out-of-date. The current number of drivers and the current way of doing business had outgrown the limits of the current system. The old system had led to an oversupply of bus drivers with too much idle time. Furthermore, the old system also required far too much manual work. For example, the days-off scheduling was done completely manually. The CEO was interested in a more effective and sophisticated way of constructing days-off and shifts and avoiding overstaffing.

The driver rostering problem in the company is non-cyclic and discontinuous and must be divided into two separate phases: days-off scheduling and shift scheduling. The drivers must know their days-off a year beforehand while the working times in a day must be known only four weeks beforehand.

The days-off scheduling problem

The company has *n* (62) full-time-equivalent drivers. Over the planning horizon of one year (13 timeframes with a timeframe of four weeks totaling 364 days), we must find *n* such sequences of days-on and days-off that satisfy the following hard constraints (see Section IV):

– A minimum number of drivers (see Table II) must be guaranteed for each working day (C2).
– A balanced number of surplus drivers must be guaranteed in each working day (C3).
– Each driver should have nine days-off in every four-week timeframe (R1).
– Drivers cannot work consecutively for more than six days (R5).
– Six drivers cannot work on weekends (R6).
– At least two working days must be assigned between two separate days-off (O2).
– The number of days-off per weekday between drivers should not differ by more than 10% (E6).
– The same sequence within each of the three driver groups with three drivers must be guaranteed (P1).

Moreover, the following soft constraints are considered:

– Single days-off should be minimized (two violations for each single day-off) (E1).
– Single working days should be avoided (one violation for each single working day) (E2).
– The maximum length of consecutive days-off is three (ten violations for each day more than three) (E3).
– The number of single days-off and single working days between drivers should not differ by more than 25% (five violations for each unit of percentage over 25) (E4).

Table II
The minimum number of drivers needed per day of week.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Drivers	47	47	47	47	50	27	10

The company has more drivers working than is needed to cover the minimum number of drivers each working day. The surplus drivers are used to cover the expected sick days.

In addition, retired-but-still-active drivers can be used if necessary. The average number of surplus drivers is calculated as follows. Per week we need 275 drivers (see Table II), hence the number of man-days needed over the planning horizon is $52 \times 275 = 14{,}300$. The drivers work $28 - 9 = 19$ days in four weeks, hence the man-days available is $62 \times 19 \times 13 = 15{,}314$. Hence we have an absolute surplus of 1014, and a surplus of 7.1% per man-day needed.

The average surplus of drivers on the different days of the week is calculated proportionally, giving 3.3, 3.3, 3.3, 3.3, 3.6, 1.9 and 0.7. The second hard constraint (C3) can now be rephrased as "On Mondays, Tuesdays, Wednesdays, Thursdays and Fridays either three or four, on Saturdays either one or two, and on Sundays either zero or one surplus drivers must be guaranteed".

Table III
The constant weights for the soft constraints and the maximum values for the hard constraints (minimum is 1).

E1	E2	E3	E4	C3	R1	R5	R6	O2	E6	P1
2	1	10	5	50	50	50	50	20	30	70

The objective is to find a solution that has no hard constraint violations and minimizes the weighted sum of the soft constraint violations. We use the adaptive penalty method for multi-objective optimization (see Section IV). The importance of the soft constraints is handled by giving them different constant weights. Hard constraint weights are dynamically calculated according to the ADAGEN method. The values of the weights, given in Table III, were decided based on the information from the company. Note that hard constraint C1 is not listed in the table because the algorithm uses the exact number of employees as given in Table II.

We generated ten days-off schedules and selected the best one. This schedule has no hard constraint violations and 443 single days-off and 513 single working days. Thus, an average driver has a single days-off in every seventh week, and a single working day in every sixth week. The number of single days-off and single working days between drivers does not differ by more than 25%. Every days-off sequence is at most three. As a result, the weighted sum of the soft constraint violations is $2 \times 443 + 1 \times 513 = 1399$. The algorithm was run on an Intel Core 2 Extreme QX9775 PC with a 3.2GHz processor and 4GB of random access memory running 64bit Windows Vista Business Edition. The best solution was found in 16 hours of computer time. The time may appear to be long. However, the point here is not to find a solution fast enough and with sufficient quality, but to find a solution of the best quality. The planning horizon is one year, so it is worth running the algorithm overnight.

The shift scheduling problem

A driver roster is a combination of shifts and days-off assignments that covers a fixed period of time. In our case, the days-off are scheduled separately prior to shift scheduling.

In Section III we stated that the driver scheduling phase partitions the vehicle schedules into pieces of work and defines the sequences of these pieces of work as shifts. A piece of work was defined as a sequence of trips on one vehicle without a break that can be performed by a single driver without interruption. A shift includes several different bus routes. The shift length is determined by the time needed to complete all the routes. The length varies between 4 hours 55 minutes and 9 hours 23 minutes. The company uses six different shift types: early, late, night, school, peak and service. Mondays, Tuesdays, Wednesdays and Thursdays have an equal shift structure. Fridays, Saturdays and Sundays each have unique shift structures. The total number of shifts is 275 per week (see Table II).

The solution of the days-off scheduling problem is the input to the shift scheduling problem. Over the planning horizon of four weeks, given a days-off schedule and a set of predetermined shifts, the problem is to find a roster for each driver that satisfies the following hard constraints:

– A driver can only be assigned to a shift he/she has competence for (O1).
– A driver assigned to a late or night shift must not be assigned to an early shift on the following day (O4).
– An employee must be assigned to an off-duty shift (day-off) on a particular day (O5).

Moreover, the following soft constraints are considered:

– The number of working hours for each driver should be 153 (one violation for each partial hour below or above 153) (R1)
– The number of different shift types between drivers should not differ by more than 25% (five violations for each unit of percentage over 25) (E5)
– Assign an early shift before a day-off or a vacation and a late or night shift after a day-off or a vacation (one violation for each such assignment) (P3).

The objective again is to find a solution that has no hard constraint violations and minimizes the weighted sum of the soft constraint violations. The rosters with less than 153 hours are considered as bad as the rosters with more than 153 hours. For some companies the most important goal could be to reduce idle time for employees. The values of the three hard constraint weights are all between one and five.

We solved the problem using exactly the same algorithm and the same computer as for the days-off scheduling problem. We generated ten shift schedules for the first four-week planning horizon and selected the best one. The best solution was found in four hours of computer time. The time is in line with the fact that the planning horizon is four weeks. The best schedule has no hard constraint violations. Sixteen drivers have exactly 153 working hours, 31 drivers have a maximum of 152 working hours and 15 drivers have a maximum of 151 working hours. Thus, an average driver has about one hour idle time, less than 1%. Note that the total sum of the working hours in all the shifts was 61 less than the total number of working hours to be scheduled to the drivers in the days-off scheduling.

The number of different shift types between drivers did not differ by more than 25%. Two other than early shifts were assigned before a day-off or a vacation and no late or night shifts after a day-off or a vacation. As a result, the weighted sum of the soft constraint violations is $1 \times 31 + 2 \times 15 + 1 \times 2 = 63$.

The company is very satisfied with our results. In their opinion the days-off scheduling algorithm could run for days because it is generated only once a year. As explained before, they are interested in the best possible value of the objective function, not in how fast this is reached. The shift scheduling algorithm can be run overnight. It is perfectly reasonable to run it for up to 15 hours, again because it is generated only once per month.

The company listed a number of advantages and savings as a result of switching to the developed system: the reduced time for developing rosters, the fairer and more balanced days-off and shifts, and the reduced idle time for bus drivers. The system also produces rosters that are more stable with regard to small changes, both in the operational environment and in the employees' work contracts. One further significant benefit is that the system can be used as a planning tool for future scenarios. The company actually demonstrated the effect of using different planning horizons and different vacation patterns.

The company wanted to integrate our algorithms into their operational systems. However, our system did not include an adequate user interface nor financial management links and customer reports. For this reason, we contacted the major bus transit software vendor in Finland. This vendor had no optimization in their product and was very interested in cooperating with us. After ten months of further coding and fine-tuning, the generated driver rostering software was integrated into the third-party vendor product. The software

- Allows users to specify the importance of requests and requirements.
- Minimizes the scheduling time required by users.
- Runs on any modern desktop computer.
- Does not use third-party mathematical software packages with expensive licensing policies.
- Generates a few solutions to choose from.
- Generates clearly different solutions to choose from.

The company, the third-party vendor and we were all very pleased with how the project ended. We do not have to work on user interfaces, financial management links, customer reports, help desks etc. Instead, we can concentrate on our core competence: development of algorithms that are useful in real-world applications.

VI. Conclusions and Future Work

We scheduled the drivers in a Finnish bus transit company. Our algorithm found feasible and acceptable solutions to their days-off scheduling and shift scheduling problems. The generated software has been integrated into a third-party vendor product.

Our direction for future research will be to solve the vehicle scheduling and driver scheduling problems that precede the driver rostering problem solved in this paper.

References

[1] Garey M.R. and Johnson D.S., Computers and Intractability. A Guide to the Theory of NP-Completeness, Freeman, 1979.

[2] Bartholdi, J.J., A Guaranteed-Accuracy Round-off Algorithm for Cyclic Scheduling and Set Covering, Operations Research 29, 501–510, 1981.

[3] Tien J. and Kamiyama A., On Manpower Scheduling Algorithms. In SIAM Rev. 24 (3), 275–287, 1982.

[4] Lau, H. C., On the Complexity of Manpower Shift Scheduling, Computers and Operations Research 23(1), 93-102, 1996.

[5] Kragelund L. and Kabel T., Employee Timetabling. An Empirical Study, Master's Thesis, Department of Computer Science, University of Aarhus, Denmark, 1998.

[6] Fukunaga, A., Hamilton, E., Fama, J., Andre, D., Matan, O. and Nourbakhsh, I, Staff scheduling for inbound call and customer contact centers, AI Magazine 23(4), 30-40, 2002.

[7] Marx, D., Graph coloring problems and their applications in scheduling, Periodica Polytechnica Ser. El. Eng. 48, 5–10, 2004.

[8] Dantzig, G.B., A comment on Edie's traffic delays at toll booths, Operations Research 2, 339–341, 1954.

[9] Alfares, H.K., Survey, categorization and comparison of recent tour scheduling literature, Annals of Operations Research 127, 145-175, 2004.

[10] Ernst, A. T., Jiang H., Krishnamoorthy, M., and Sier, D., Staff scheduling and rostering: A review of applications, methods and models, European Journal of Operational Research 153 (1), 3-27, 2004.

[11] Meisels, A. and Schaerf, A. 2003, Modelling and solving employee timetabling problems, Annals of Mathematics and Artificial Intelligence 39, 41-59, 2003.

[12] Bard, J. and H. Purnomo, Hospital-wide reactive scheduling of nurses with preference considerations, IIE Trans. 37(7), 589–608, 2005.

[13] Beddoe, G.R., Petrovic, S. and Li, J., A Hybrid Metaheuristic Case-based Reasoning System for Nurse Rostering, Journal of Scheduling 12, 99–119, 2009.

[14] Bilgin, B., De Causmaecker, P., Rossie, B. and Vanden Berghe G., Local Search Neighbourhoods to Deal with a Novel Nurse Rostering Model. In Proc. of the 7th Int. Conf. on the Practice and Theory of Automated Timetabling, Montréal, Canada, 2008.

[15] Burke, E., P. De Causmaecker, S. Petrovic, and G.Vanden Berghe, Metaheuristics for Handling Time Interval Coverage Constraints in Nurse Scheduling, Applied Artificial Intelligence, 743-766, 2006.

[16] Diaz, T., D. Ferber, C. deSouza, A. Moura. 2003. Constructing nurse schedules at large hospitals. Internat. Trans. Oper. Res. 10(3), 245–265.

[17] Kawanaka, H., T. Yoshikawa, T. Shinogi, S. Tsuruoka. 2003. Constraints and search efficiency in nurse scheduling problem. Proc. 2003 Internat. Sympos. Comput. Intelligence Robotics Automation, Vol. 1, 312–317.

[18] Meyer auf'm Hofe, H. 2001. Solving rostering tasks by generic methods for constraint optimization. Internat. J. Foundations Comput. Sci. 12(5) 671–693.

[19] Van Wezel, W., R. Jorna. 1996. Scheduling in a generic perspective: Knowledge-based decision support by domain analysis and cognitive task analysis. Internat. J. Expert Systems 9(3) 357–381.

[20] Dowling, D., Krishnamoorthy, M., Mackenzie, H. and Sier, D., Staff rostering at a large international airport", Annals of Operations Research 72, 125-147, 1997.

[21] Beer, A., Gaertner, J., Musliu, N., Schafhauser, W. and Slany, W., Scheduling breaks in shift plans for call centers. In Proc. of the 7th Int. Conf. on the Practice and Theory of Automated Timetabling, Montréal, Canada, 2008.

[22] Stolletz, R., Operational workforce planning for check-in counters at airports. Transportation Research Part E 46, 414-425, 2010.

[23] Lusby, R., Dohn, A., Range, T. and Larsen, J., Ground Crew Rostering with Work Patterns at a Major European Airlines. In Proc of the 8th Conference on the Practice and Theory of Automated Timetabling (PATAT), Belfast, Ireland, 2010.

[24] Ásgeirsson, E.I., Bridging the gap between self schedules and feasible schedules in staff scheduling. In Proc of the 8th Conference on the Practice and Theory of Automated Timetabling (PATAT), Belfast, Ireland, 2010.

[25] Bard, J. F., Binici, C. and Desilva, A. H., Staff Scheduling at the United States Postal Service, Computers & Operations Research 30, 745-771. 2003.

[26] Nurmi K., Kyngäs J. and Post G., Staff Scheduling for Bus Transit Companies, Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2011, IMECS 2011, 16-18 March, 2011, Hong Kong
[27] Seçkiner, S.U. and Kurt, M., Ant colony optimization for the job rotation scheduling problem, Applied Mathematics and Computation 201(1-2), 149-160, 2008.
[28] Elshafei M. and Alfares H., A dynamic programming algorithm for days-off scheduling with sequence dependent labor costs, Journal of Scheduling 11(2), 85-93, 2008.
[29] Qu R. and He F., A hybrid constraint programming approach for nurse rostering problems. In Allen, Ellis, and Petridis, Editors, Applications and Innovations in Intelligent Systems XVI, Cambridge, UK, 211-224, 2008.
[30] Dean J., Staff Scheduling by a Genetic Algorithm with a Two-Dimensional Chromosome Structure. In Proc of the 7th Conference on the Practice and Theory of Automated Timetabling, Montreal, Canada, 2008.
[31] Burke, E.K., Curtois, T., Qu, R. and Vanden Berghe, G., A scatter search methodology for the nurse rostering problem, Journal of the Operational Research Society , 2010.
[32] Remde, S., Cowling, P. I., Dahal, K. P. and Colledge, N., Exact/Heuristic Hybrids Using rVNS and Hyperheuristics for Workforce Scheduling. In Proc. of the 7th Evolutionary Computation in Combinatorial Optimization, Lecture Notes in Computer Science 4446, Springer, 188-197, 2007.
[33] Brunner, J.O., Bard, J.F., and Kolisch, R., Flexible shift scheduling of physicians, Health Care Management Science. 12(3), 285-305, 2009.
[34] Burke, E., De Causmaecker P., Petrovic S., and Vanden Berghe G., Metaheuristics for Handling Time Interval Coverage Constraints in Nurse Scheduling, Applied Artificial Intelligence, 743-766, 2006.
[35] Goodale, J. and Thompson, G., A Comparison of Heuristics for Assigning Individual Employees to Labor Tour Schedules, Annals of Operations Research 128(1), 47-6, 2004.
[36] Musliu, N., Heuristic Methods for Automatic Rotating Workforce Scheduling, International Journal of Computational Intelligence Research 2(4), 309-326, 2006.
[37] Burke, E.K., Curtois, T., Post, G.F., Qu, R. and Veltman, B., A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem, European Journal of Operational Research 188(2), 330-341, 2008.
[38] Bertossi, A.A., Carraresi, P., Gallo, G., On some matching problems arising in vehicle scheduling models, Networks 17, 271–281, 1987.
[39] Dantzig, G.B., and Ramser, J.H., The truck dispatching problem, Management Science 6, 81-91, 1959.
[40] Desaulniers, G and Hickman, M., Public transit. Handbooks in Operations Research and Management Science, Transportation, Vol. 14, G. Laporte and C. Barnhart (eds), Elsevier, Amsterdam, 69-127, 2007.
[41] Bunte, S., & Kliewer, N., An overview on vehicle scheduling models. Journal of Public Transport 1(4), 299-317, 2009.
[42] Wren, A; Fores, S; Kwan, A; Kwan, R; Parker, M; Proll, L., A flexible system for scheduling drivers, Journal of Scheduling 6, 437-455, 2003.
[43] Li J. and Kwan R.S.K., A Self-Adjusting Algorithm for Driver Scheduling, Journal of Heuristics 11 (4), 351-367, 2005.
[44] Ásgeirsson, E.I., Kyngäs, J., Nurmi, K. and Stølevik, M., A Framework for Implementation-Oriented Staff Scheduling, In Proc of the 5th Multidisciplinary Int. Scheduling Conf.: Theory and Applications (MISTA), Phoenix, USA, 2011.
[45] Lin, S. and Kernighan B. W., An effective heuristic for the traveling salesman problem, Operations Research 21, 498–516, 1973.
[46] Glover, F., New ejection chain and alternating path methods for traveling salesman problems. In Computer Science and Operations Research: New Developments in Their Interfaces, edited by Sharda, Balci and Zenios, Elsevier, 449–509, 1992.
[47] Nurmi, K. and Kyngäs, J., Days-off Scheduling for a Bus Transportation Staff. In Proc of the 4th International Conference on Bioinspired Optimization Methods and their Applications, Ljubljana, Slovenia, 2010.
[48] Nurmi, K., Genetic Algorithms for Timetabling and Traveling Salesman Problems, Ph.D. dissertation, Dept. of Applied Math., University of Turku, Finland, 1998. Available: http://www.bit.spt.fi/cimmo.nurmi/
[49] Preux, P. and Talbi, E-G., Towards Hybrid Evolutionary Algorithms International Transactions in Operational Research 6, 557-570, 1999.
[50] Kyngäs, J. and Nurmi, K., Scheduling the Finnish 1st Division Ice Hockey League. In Proc. of the 22nd Florida Artificial Intelligence Research Society Conference, Florida, USA, 2009.
[51] Nurmi, K. and Kyngäs, J., A Framework for School Timetabling Problem. In Proc. of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications, Paris, France, 2007.
[52] Kyngäs, J. and Nurmi, K., Scheduling the Finnish Major Ice Hockey League. In Proc. of the IEEE Symposium on Computational Intelligence in Scheduling, Nashville, USA, 2009.