Ann Oper Res (2016) 239:207–223
DOI 10.1007/s10479-014-1674-7

An assessment of a days off decomposition approach to personnel shift scheduling

Sophie van Veldhoven · Gerhard Post · Egbert van der Veen · Tim Curtois

Published online: 25 July 2014

Abstract This paper studies a two-phase decomposition approach to solving the personnel scheduling problem. The first phase creates a days-off-schedule, indicating working days and days off for each employee. The second phase assigns shifts to the working days in the days-off-schedule. This decomposition is motivated by the fact that personnel scheduling constraints are often divided into two categories: one specifies constraints on working days and days off, while the other specifies constraints on shift assignments. To assess the consequences of the decomposition approach, we apply it to public benchmark instances, and compare this to solving the personnel scheduling problem directly. In all steps we use mathematical programming. We also study the extension that includes night shifts in the first phase of the decomposition. We present a detailed results analysis, and analyze the effect of various instance parameters on the decompositions' results. In general, we observe that the decompositions significantly reduce the computation time, but the quality, though often good, depends strongly on the instance at hand. Our analysis identifies which aspects in the instance can jeopardize the quality.

S. van Veldhoven
Aviv, Langestraat 11, 7511, HA Enschede, The Netherlands
e-mail: sophievveldhoven@gmail.com

G. Post (✉) · E. van der Veen
Department of Applied Mathematics, University of Twente, P.O. Box 217, Enschede, The Netherlands
e-mail: g.f.post@math.utwente.nl; g.f.post@utwente.nl

E. van der Veen
e-mail: Egbert.vanderVeen@ortec.com

G. Post · E. van der Veen
ORTEC, Houtsingel 5, 2719, EA Enschede, The Netherlands

T. Curtois
ASAP, School of Computer Science, University of Nottingham, Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, UK
e-mail: tim.curtois@nottingham.ac.uk

Keywords Personnel scheduling · Days off scheduling · Decomposition · Mathematical programming

1 Introduction

Assigning personnel to work shifts is challenging, both in theory and practice. In theory, the personnel scheduling problem is a well known hard combinatorial optimization problem. In practice, constructing high-quality shift schedules is often difficult and time consuming, due to the large number of working time regulations and employee preferences such as shift and vacation requests.

In many practical situations, scheduling days off is a separate step in the shift assignment process. For employees, it is preferable or even requisite, see Nurmi et al. (2011), to know their working days a long time ahead, so they can plan their free time. The exact working hours are not essential to know in the long term. In addition, vacation requests can be taken into account long before the actual shift planning, which may alert the planner to possible capacity problems.

In this paper, we study a decomposition approach for the personnel scheduling problem, that first solves the days-off-scheduling problem, which assigns working days and days off to a set of employees, and secondly assigns specific shifts, e.g., early or late, to working days in the days-off-schedule. This method reduces the complexity of the personnel rostering problem by decomposing the problem into subproblems that are easier to solve. Since a decomposition approach implies a possible loss of solution quality, we want to mitigate this effect by solving the subproblems to optimality. To achieve this, we apply integer linear programming to the subproblems.

The contribution of this research is an analysis of the potential and pitfalls of days off decomposition. We compare two approaches: one with and one without night shifts included in the first phase of the decomposition. To let the decomposition be the only source of loss in solution quality, both phases of the decomposition are solved to optimality. The decomposition approaches are evaluated on 25 public nurse rostering benchmark instances, see Curtois (2007).

This paper is structured as follows. First, Sect. 2 discusses the related literature, and then Sect. 3 defines the personnel scheduling problem. In Sect. 4, we present our decomposition approach, and Sect. 5 and 6 discuss and assess the computational results. Section 7 presents the conclusions.

2 Literature review

There is a large amount of literature on personnel scheduling, see the recent review by Van den Bergh et al. (2013), and the reviews by Ernst et al. (2004), Burke and Causmaecker (2004), and Cheang et al. (2003). Within this literature there are several examples of decomposition algorithms. In this section, we review these approaches and indicate how the decomposition as proposed in this paper contributes to the existing literature.

Decomposition appears to be a natural approach for personnel scheduling because of its problem structure. One way to decompose the problem is into individual scheduling subproblems for each employee. These separate problems are then only linked by shift cover (demand) constraints. This is the model used in Ikegami and Niwa (2003). The authors then further decompose the subproblems by splitting the planning horizon into shorter periods.

The shorter periods are then solved using a branch and bound method and the solutions iteratively used in the complete problem within a tabu search framework.

This is the same general decomposition used by column generation and branch and price methods for staff rostering such as in Burke and Curtois (2014), Maenhout and Vanhoucke (2010), and Mason and Smith (1998). Here columns are generated by solving subproblems which correspond to creating individual schedules for each employee. The subproblems are often modeled as resource constrained shortest path problems and solved using a dynamic programming algorithm. The subproblem can also be modeled as integer programming problems and solved with a mathematical programming based solvers or even with heuristics.

Another natural decomposition, which was used in Brucker et al. (2010), involves enumerating sequences of shifts. Instead of modeling individual shift assignments as in most approaches, their approach instead involves the assignment of the pre-enumerated, short, shift sequences. This reduces the size of the problem and consequently the computation time. The sequences are then assigned using heuristics. Millar and Kiragu (1998) use a similar decomposition on smaller, cyclic and non-cyclic nurse rostering problems but instead assign the sequences using a mathematical programming solver.

An even simpler decomposition, used in Azaiez and Al Sharif (2005), is to heuristically split the employees into groups which contain a balance of skills. The problem formed from each group is solved using a mathematical programming solver and the individual solutions are combined to produce the overall solution.

A decomposition more similar to ours is that used in Valouxis et al. (2012). They also first solve the days-off-scheduling problem before solving the shift types problem. However, before solving the days-off-scheduling problem they further decompose the planning horizon into seven day stretches as in Ikegami and Niwa (2003). After solving the shift assignment problem they then apply various neighborhood searches and search re-start strategies.

First solving the days-off-scheduling problem and then assigning shifts to employees on their working days is also done in Baxter and Mosby (1988), Day and Ryan (1997), and Gärtner et al. (2001). The research in Baxter and Mosby (1988) uses heuristics to solve the subproblems, whereas we use mathematical programming to get optimal solutions for the subproblems. In Day and Ryan (1997) and Gärtner et al. (2001), the subproblems are solved by enumerating days off patterns and solving partitioning problems. In Day and Ryan (1997), the days-off-scheduling problem is solved to optimality, however the shift assignment phase is not necessarily solved to optimality. Due to the heuristic nature of their methods, Baxter and Mosby (1988) and also Day and Ryan (1997) include a rescheduling method in their approaches.

In Abdennadher and Schlenker (1999), the shift rostering problem is decomposed into three phases: days off assignment, night shifts assignment, and morning and evening shift assignment. Each phase is solved to optimality. Our approach integrates this first and second phase. Unfortunately, Abdennadher and Schlenker (1999) do not report any computational results. The research in Parr and Thompson (2007) also decomposes the shift rostering problem in three phases. First, employees that are going to work night shifts are determined, second, the days-off-schedule is generated, and third, shifts are assigned to employees on working days. In Parr and Thompson (2007), two alternative local search methods are compared to solve these phases.

Finally, another recently introduced decomposition, involves decomposing the planning horizon while keeping the rest of the problem intact. This is the approach adopted in Santos et al. (2012). They iteratively create gradually increasing sized windows over the planning horizon and solve just the subproblems within the windows. This can also be regarded as a

very large-scale neighborhood search and similar to the neighborhoods used in Valouxis et al. (2012).

3 Problem description

This section describes the personnel scheduling problem (see Sect. 3.1) and the set of benchmark instances to which we apply our approach (see Sect. 3.2).

3.1 The personnel scheduling problem

The basic data in the personnel scheduling problem is given by:

- A *time period* consisting of a number of consecutive days, usually one or multiple weeks.
- A set of *employees* with certain *skills*.
- A set of *shift types*: A shift type is a time interval which starts at a fixed time during the day and in which a series of tasks is performed.
- A set of *shifts*: A shift is of one of the shift types with a given start day.

The objective of the personnel scheduling problem is to assign the set of shifts to the set of employees, while respecting a number of constraints:

- **Single shift per day**: Each employee works at most one shift per day.
- **Coverage requirements**: The problem instance describes *per day* a minimum, maximum, or preferred number of shifts on a day.
- **Employee constraints**: Each employee's shift assignment must satisfy a specified set of labor rules. In addition, employee specific work agreements and requests must be taken into account as well.

We formulate the personnel scheduling problem as a mathematical program. Since the main decision is to assign on each day a shift type to an employee, it is natural to introduce the binary variables x_{eds} representing this decision:

$$x_{eds} = \begin{cases} 1 \text{ if shift type } s \text{ on day } d \text{ is assigned to employee } e \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

The constraint that an employee is assigned to at most one shift on each day translates into:

$$\sum_{s} x_{eds} \leq 1 \qquad (\forall e, d) \tag{2}$$

Formulating all constraints appearing in the benchmark instances as linear constraints is tedious but most of the time straightforward. The details of these formulations are not described in this paper, but can be found in Van Veldhoven (2011). To provide the reader insight in the models, Sect. 4.2 discusses several constraints for the days-off-scheduling problem.

3.2 The benchmark instances

The Employee Scheduling Benchmark instances, see Curtois (2007), were collected over a period of several years by a number of researchers investigating the personnel scheduling problem. The instances were derived from challenging real-world instances. Because they were drawn from many sources they vary in dimensions such as the number of staff, shifts

Table 1 The instances of the employee scheduling benchmark data sets used for testing

Instance	Employees	Shift types	Days	Best
Azaiez	13	2	28	0
BCDT-sep	20	4	30	100
BCV-3.46.2	46	3	26	894
BCV-4.13.1	13	4	29	10
CHILD	41	5	42	149
ERMGH	41	4	48	779
ERRVH	51	8	48	2,001
GPost	8	2	28	5
GPost-B	8	2	28	3
Ikegami-2shift-1	28	3	30	0
Ikegami-3shift-1	25	4	30	2
Ikegami-3shift-1.1	25	4	30	3
Ikegami-3shift-1.2	25	4	30	3
LLR	27	3	7	301
MER	54	12	48	7,081
Millar-2shift-1	8	2	14	0
Millar-2shift-1.1	8	2	14	0
ORTEC01	16	4	31	270
ORTEC02	16	5	31	270
Ozkarahan	14	2	7	0
QMC-1	19	9	28	13
QMC-2	19	3	28	29
SINTEF	24	5	21	0
WHPP	30	3	14	5

types, skills and length of the planning period. More challengingly from a modeling and computational aspect however, they also vary in the number and types of constraints present in each instance. This is due to their real-world nature and the fact that each employer has different requirements often affected by national legislation, union or industry specific working regulations. Each instance is often further complicated by the presence of employee specific contracts such as part-time or night shift workers. For detailed information we refer the reader to Curtois (2007) where full details are available on each specific instance.

At this moment there are 27 instances available, of which 25 are used in this research, see Table 1. Table 1 provides an overview of the instances' dimensions and their best known objective function value (column 'best'). The best known solutions are known to be optimal in all cases except MER.[1] The best solutions, found by different researchers and different techniques, are also available online, see Curtois (2007). The instance Musa is not used in our research, as it contains only 1 shift type, and HED01 was not included because it uses conditional constraints, which we did not implement.

[1] For MER a lower bound of 7,079 was established.

4 Solution approach

This section outlines our solution approach to the personnel scheduling problem. We solve the personnel scheduling using a decomposition that first solves a days-off-scheduling problem. In days-off-scheduling only days off and working days are determined, so employees are not assigned to specific shifts. The idea behind the decompositions is that the position of the stints, i.e., the consecutive days with work for an employee, is dominant for the personnel scheduling problem; once the working days are known, we hope that for each working day a shift type may be chosen such that the remaining personnel scheduling constraints are met.

In the decomposition, the days-off-schedule is used as input to assign employees to specific shifts. Of course, employees may only be assigned to shifts on working days in the days-off-schedule. First, Sect. 4.1.1 outlines how we formulate and solve the days-off-scheduling problem for the benchmark instances described in Sect. 3.2. Section 4.1.2 describes an extension to our decomposition that also includes night shifts in the days-off-scheduling problem.

In order to apply these decompositions, the personnel scheduling instances have to be reduced to days-off-scheduling instances, which is discussed in Sect. 4.2.

4.1 Relations between the models

4.1.1 Days-off-scheduling

The basic decision in the days-off-scheduling problem is to decide for each day in the schedule of an employee whether it is a working day. This is represented by the variables y_{ed}:

$$y_{ed} = \begin{cases} 1 & \text{if employee } e \text{ works on day } d \\ 0 & \text{if employee } e \text{ has a day off on day } d. \end{cases} \tag{3}$$

The basic relation with the personnel scheduling problem is:

$$\sum_{s} x_{eds} = y_{ed} \qquad (\forall e, d). \tag{4}$$

The second phase in the decomposition is exactly the original personnel scheduling problem with the additional relations of Eq. (4).

4.1.2 Night shift scheduling

Constraints on night shifts can lead to very specific constraints on the position of stints, see also Glass and Knight (2010). An example is a required separation of at least two days between two stints if the first stint ends with a night shift. Hence, we also investigate an extension of the basic days-off-scheduling model in which the working day ($y_{ed} = 1$) is specialized to working a selected shift type, which we call the *night shift* (N). We stress that this need not be the night shift in the sense of working at night; it could be any of the available shift types. However, usually singling the night shift as special shift gives the best results, as the night shift usually comes with many specialized constraints. In our results we found two examples where another shift type as 'night shift' gives better results, see Table 4: shift type L in instance BCV-3.46.2 and O in instance QMC-1. Moreover the instance BCV-4.13.1 contains no night shift at all. Setting DH as special shift ('night shift') in this case gives an improvement in result.

To handle the special shift, we introduce the binary variable z_{ed}:

$$z_{ed} = \begin{cases} 1 & \text{if employee } e \text{ works shift type } N \text{ on day } d \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

In the days-off-scheduling problem, we add the following constraints:

$$z_{ed} \leq y_{ed} \qquad (\forall e, d), \tag{6}$$

and for the personnel scheduling problem the next constraints are added:

$$x_{edN} = z_{ed} \qquad (\forall e, d). \tag{7}$$

Equation (6) expresses that an employee can only work a night shift on working days. Equation (7) expresses that the choice for a night shift in the first phase should be respected in the second phase.

We refer to the decomposition approaches as:

(On, Off): Assigns working days and days off in the first phase.
(Day, Night, Off): Assigns working days, night shifts, and days off in the first phase.

4.2 Modeling the constraints

From a higher level, we can say that we reduce the original personnel scheduling problem to a personnel scheduling problem with two shift types (On, Off) and three shift types (Day, Night, Off), respectively. Hence, to solve the first and second phase of the decomposition, similar mathematical programming formulations can be used, where the work schedules created in the second phase should obey the decisions of the first phase. Although the mathematical programming formulations are similar, the reduction in the first phase is not always straightforward. Our basic principle is to reformulate the constraints of the personnel scheduling instances to necessary constraints for the days-off-scheduling problem; in this way we are able to assess the consequences of the decomposition in a uniform way. The next subsections describe how different types of constraints are handled in the days-off-scheduling problems. For full details on the mathematical programs we refer to Van Veldhoven (2011).

4.2.1 Requests and pre-assigned shifts

Employees may have pre-assigned shifts, requests for specific shift types on specific days, or requests for days off.

- Work requests or shift-on-requests result in working days in the days-off-scheduling problem.
- Shift *off* requests are ignored, since the employee might work another shift on the same day. Hence we do not know if the employee will work or have a day off.
- A days off request is copied to the days-off-scheduling problem.

For the implementation it is important to incorporate pre-assigned shifts and requests in the possible matches of the pattern constraints, see Sect. 4.2.3. For example, if there is a pre-assigned shift we can evaluate whether this shift matches a certain pattern or not, even in the days-off-scheduling phase.

4.2.2 Coverage requirements

Coverage requirements express the minimum and maximum number of employees that are required to be available on a day either per shift type or per time interval. In the latter case, shift types assigned to the employees should be such that these limits per time interval are respected.

In both cases, the definition of a coverage requirement can indicate that only employees having certain skills contribute to the coverage requirement. In the instances of the Employee Scheduling Benchmark Data Sets, employees can contribute to more than one coverage requirement at the same time. For example, if there are two coverage constraints for shift type A, one with skill s_1 and one with skill s_2, then an employee working shift A, having both skills s_1 and s_2 will count for both coverage requirements.

In the first phase of our decomposition, we determine employees that work on a certain day. The employees should be chosen such that they can satisfy the coverage requirements. However, to determine whether the chosen employees are able to cover all coverage requirements is non-trivial, since we do not assign shifts in this phase. Still, to make 'reasonably sure' that the chosen employees are able to cover all coverage requirements, we formulate and solve a small linear program.

Suppose the skills are represented by the set Q, and that shift type s requires that at least m_{dsq} employees having skill $q \in Q$ are assigned shift type s on day d. Let E_q denote the subset of employees having skill q. Each (shift type, skill) coverage requirement on day d will induce a linear constraint in the days-off-scheduling problem of the form:

$$\sum_{e \in E_q} y_{ed} \geq m_{dq}.$$

This constraint expresses that on day d at least m_{dq} employees with skill q should be present. Here, m_{dq} is solved by a mathematical program, which is solved as preprocessing to the days-off-scheduling phase of the decomposition. Let Z_{eds} denote binary variables such that:

$$Z_{eds} = \begin{cases} 1 & \text{if employee } e \text{ works on day } d \text{ in shift type } s \\ 0 & \text{otherwise.} \end{cases}$$

Then m_{dq} is the result of the minimization problem:

$$m_{dq} = \min \sum_{e \in E_q} \sum_{s} Z_{eds}$$

for a given (d, q)-combination. The constraints are:

$$\sum_{s} Z_{eds} \leq 1 \qquad (\forall e) \tag{8}$$

and

$$\sum_{e \in E_q} Z_{eds} \geq m_{dsq} \qquad (\forall s). \tag{9}$$

Here, equation (8) expresses that an employee can be assigned to at most one shift on day d. Equation (9) expresses that there should be assigned at least m_{dsq} employees with skill q to shift type s on day d.

For each day d, this model will enforce that there are enough skilled employees around. The problem can be that there is a unique set of employees for which the minimum m_{dq} is attained, potentially leading to infeasibilities in the second phase.

4.2.3 Pattern constraints

Pattern constraints express a wide variety of constraints for individual schedules. A pattern consists of a description of sequences of shifts that form a *match* for the pattern. Moreover, the pattern contains information for which period of the schedule the pattern needs to be considered, e.g., for the full scheduling period or only the periods starting on Saturday. Pattern constraints seem complicated at first, but they are able to model a wide variety of constraints that appear in practice. They can model constraints on, e.g., the maximum number of shifts per week, shift sequences, and the minimum number of consecutive shifts. Pattern constraints specified for a specific shift type cannot be incorporated in the construction of a days-off-schedule. In the case of (Day, Night, Off), we can also incorporate pattern constraints for the night shift as well. For example, it is common that there are constraints on the total number of night shifts and the length of night shift sequences. Moreover, night shifts are usually at the end of stints, which can pose some restrictions on its position, for example that such stint should not end on Friday.

4.2.4 Workload requirements

Workload requirements describe the number of hours an employee should work in the planning period. Clearly, these are difficult to use in the days-off-schedule if different shift types have different lengths. Since we use only necessary conditions, the best we can do is to calculate upper and lower bounds on the working days, and add those conditions as constraints to the mathematical program. For example if employee e has a maximum workload of 120 hours in the planning period, and the shortest shift contains 7 hours of work in the scheduling period, then we add the constraint:

$$7 \cdot \sum_{d} y_{ed} \leq 120 + \epsilon, \qquad \epsilon \geq 0.$$

The objective function will contain the non-negative slack variable ϵ. In this way, the constraint will penalize solutions that assign more than 17 working days to employee e.

In the second phase of the decomposition, we have full information, so that we can use the correct workload requirements.

5 Results

To evaluate our decomposition approach, we solve the mathematical programming formulations of the personnel scheduling instances in Table 1 using CPLEX 12.2 with the time limit set to one hour on a Dell Optiplex 990 (64-bit, 3.4 GHz, 4 GB RAM, 8 cores). As the objective value is integer, we set the absolute gap to 0.999 without losing the optimality guarantee. The two phases of the decomposition are also solved using CPLEX, with the time limit set to 30 minutes for each phase.

Table 2 and Table 4 present results for instances with two shift types and instances with more than two shift types, respectively. To the instances with two shifts types only the (On, Off)-approach is applied, since the (Day, Night, Off)-approach would be the original personnel scheduling problem. For 15 instances the direct approach finds the optimal solution within one hour; these are the solution values that are marked with * in the second column. Moreover, 12 instances are solved within one minute, see the runtime in the third column.

Table 2 Results on instances with two shift types

Instance	Direct		(On, Off)	
	Cost	Time (sec)	Cost	% Time saving
Azaiez	0*	7.4	6	76.7
GPost	5	117	4,002	96.4
GPost-B	3*	428	2,002	92.5
Millar-2shift-1	0*	0.2	0*	66.2
Millar-2shift-1.1	0*	0.2	0*	61.6
Ozkarahan	0*	0.2	800	90.4
Ozkarahan (skills)	0*	0.1	0*	87.2

* optimal value

First, we analyze the results of the (On, Off)-decomposition approach to the instances with two shift types, and after that we analyze the results of our decomposition approaches to the instances with more than two shift types.

5.1 Instances with two shift types

The fourth column in Table 2 gives the solution values of the (On, Off)-decomposition, while column 5 states the time saving, i.e., 100 % minus the percentage of the time needed to solve the decomposition divided by the time needed to solve the direct approach. For example, the time saving of 96.4 % for the GPost instance indicates that the decomposition is solved in 4.2 s (3.6 % of 117 s).

We observe that the decomposition is solved faster than the direct approach. However, except for the Millar and the Ozkarahan (skills) instances, the results (column 4) are not competitive. The reasons are slightly different for each instance. The Azaiez instance requires separate stints for day shifts and night shifts, information that can not be incorporated in the (On, Off)-decomposition. The GPost instances require that some specific stints end at specific days, for example (sub)stints of night shifts should end on specific days, see Glass and Knight (2010). The Ozkarahan instance contains skills in the coverage requirements, which is modeled by certain forbidden patterns for employees. If we remodel this using skills for the employees then the decomposition finds the optimal solution. This is instance Ozkarahan (skills), the final row in Table 2

These two aspects, the special role of night shifts and the complication with skills, reappear in the instances with more than two shift types. The special role of night shifts motivated us to extend the (On, Off)-approach to the (Day, Night, Off)-approach.

There is another aspect that deserves attention, namely the fact that the days-off-scheduling instances usually have many optimal solutions, which can lead to solutions of different quality in the shift scheduling phase. To investigate this, we have performed 10 additional runs on the instances in Table 2 and recorded the costs of the best solution, median cost, the average cost, and the cost of the worst solution. To generate these alternative optimal solutions we have added a small random cost to the decision variables in the days-off-scheduling phase. The result are given in Table 3.

We see that in three of the four cases that the optimum was not reached in Table 2 the result of the best of 10 runs is better. Moreover, we see that the average of the medians of the 11 runs is exactly the average of the values in column 4 of Table 2. This supports our opinion that though on individual cases the non-random run may be slightly 'off'-shape, on

Table 3 Results on instances with two shift types based on 11 runs

Instance	Best	Median	Average	Worst
Azaiez	0	6	6	18
GPost	2,002	3,002	3,502	5,002
GPost-B	1,002	3,002	2,602	3,002
Millar-2shift-1	0	0	0	0
Millar-2shift-1.1	0	0	0	0
Ozkarahan	800	800	1,120	1,600
Ozkarahan (skills)	0	0	0	0

Table 4 Results on instances with more than two shift types

Instance	Direct		(On, Off)		(Day, Night, Off)		
	Cost	Time (sec)	Cost	% Time saving	Night shift	Cost	% Time saving
BCDT-Sep	104,539	3,602	104,229	99.9	N	2,350	85.2
BCV-3.46.2	894*	535	1,215	99.7	N	3,430	82.8
BCV-3.46.2					L	2,374	99.8
BCV-4.13.1	10*	2	17	65.9	DH	10*	54.2
CHILD	149*	47	5,066	86.9	N	3,847	89.9
ERMGH	779*	2	929	27.9	N	1,116	−119.0
ERRVH	2,204	385	12,832	97.9	N	22,796	91.3
Ikegami-2sh-1	0*	27	4,316	97.7	N	0*	23.1
Ikegami-3sh-1	110	3,601	807	90.8	N	3,884	99.8
Ikegami-3sh-1.1	6	3,601	994	94.8	N	2,728	90.8
Ikegami-3sh-1.2	23	3,602	994	90.8	N	1,960	85.4
LLR	301*	1	312	80.2	N	308	92.6
MER	56,561	628	122,022	−187.0	N	142,133	−16.8
ORTEC01	3,527	1,365	2,270	94.1	N	280	92.4
ORTEC02	2,836	2,385	1,275	96.5	N	275	96.5
QMC-1	13*	3	29,046	77.7	N	21,070	38.8
QMC-1					O	177	86.9
QMC-2	29*	0	1,045	30.8	N	33	−92.1
SINTEF	0*	2	12,202	77.9	N	17	77.9
Valouxis-1	140	1,461	20*	64.5	N	20*	93.9
WHPP	3,008	155	16,000	99.5	N	3,001	77.7

* optimal value

average it is representative. For this reason, we have not included here the extensive tables for the runs for more than two shift types.

5.2 Instances with more than two shift types

For the instances with more than two shift types, the results of both decomposition approaches are presented in Table 4. Again results are compared with the direct approach. The first 5

columns in Table 4 are the same as the first five columns in Table 2. Column 6 indicates the shift that is chosen as 'night shift'; remember from Sect. 4.1.2 that we use the term 'night shift' to designate a special shift that is singled out in the (Day, Night, Off)-decomposition. For two instances (BCV-3.46.2 and QMC-1) choosing the shift type N as the special shift does not give the best result. For these instances, we added an extra row in Table 4 for the special shift that gave the best solution in the decomposition approach. Instance BCV-4.13.1 does not have a night shift at all, and choosing shift DH as the special shift leads to the optimal solution. Column 7 and 8 give the result of the (Day, Night, Off)-decomposition and the time saving, respectively.

For 9 instances, the direct approach finds an optimal solution, whereas the decomposition finds optimal solutions for 3 instances. Optimal values are indicated by *. In the cases where the optimal solution is not reached in the direct approach and the time listed is less than 3,600 s, the process ran out of memory. This is the case with the instances ERRVH, MER, ORTEC01, ORTEC02, Valouxis-1, and WHPP. The decomposition yields better solutions in 5 instances, and in 2 instances the solution quality is comparable (LLR and QMC-2). For 9 out of 19 instances the decomposition approach gives satisfactory solutions. The time saving is in most cases substantial: in 11 of the instances the saving is 90 percent or more. However, for 3 instances the decomposition takes more time and yields worse solutions. Instance MER is the largest instance. On this instance none of the models give a satisfactory result. Solving the direct model is terminated after 628 s since the solver ran out of memory. The decomposition models run longer without going out of memory, explaining the fact that there is a negative time saving here.

In several instances (BCV-3.46.2, ERMGH, ERRVH, all Ikegami–3Sh instances, and MER) the (On, Off)-approach yields better solutions than the (Day, Night, Off)-decomposition. This and other aspects are analyzed in detail in the next section.

6 Assessment of the decomposition approaches

In this section we investigate the quality of solutions produced by the decompositions approaches. Basically there are two orthogonal explanations for poor quality solutions of the decomposition approach:

- **The essence of the instance could be caught by the days-off-scheduling problem**: Hence some information, hidden in a combination of factors, was not taken into account in *the implementation of* the days-off-scheduling problem. A way to improve this is to reformulate certain constraints, or reformulate the instance, such that the solutions we are interested in, still correspond to low costs. This is instance-dependent, and difficult to automate.
- **The essence of the instance is lost in the days-off-schedule**: The instance may contain aspects that can not be handled in the days-off-schedule and are of decisive importance. The night shift was such an aspect which led to the (Day, Night, Off)-approach. Other aspects are shifts with specific requirements, coverage requirements with skills, and strong preferences for shift sequences.

Our findings are summarized in Table 5. The contents of the columns 'aggr', 'shift', 'cover', 'night', 'seq' and 'run' are addressed in Sects. 6.1.1–6.1.6, respectively.

Table 5 Analysis of instances

Instance	Aggr.	Shift	Cover	Night	Seq.	Run
BCDT-Sep		V		×		×
BCV-3.46.2	×	DH			×	×
BCV-4.13.1		DH				
CHILD	×	ET			×	
ERMGH	×					×
ERRVH				×		
Ikegami-3shift-1	×	O	×		×	×
Ikegami-3shift-1.1	×	O	×		×	×
Ikegami-3shift-1.2	×	O	×		×	×
LLR						
MER						×
ORTEC01						×
ORTEC02		V				×
Ozkarahan			×			
QMC-1		O				
QMC-2					×	
SINTEF					×	
Valouxis-1	×					
WHPP	×					×

6.1 Detailed analysis of the results

6.1.1 Aggregate constraints

The '×' in column 'aggr' of Table 5 indicates that a set of constraints in the instance was merged into one aggregated constraint to form an equivalent but more compact instance. Many instances contain a number of constraints expressing that the night shift can not be followed by shift types E, or D, or L, etc. The aggregated constraint expresses that the night shift can not be followed by the *shift group* consisting of the types E, D, L, etc. In our tests we used these aggregated versions; these are available on Curtois (2007).

6.1.2 Special shifts

Several special shifts appear in the instances, which usually complicate the days-off-scheduling. There are two classes of these shifts:

– **Absence shifts**: These shifts will not contribute to the coverage requirements, but can count as work (in the Ikegami instances and ORTEC02) or not (QMC-1 and BCDT-Sep). The coverage requirements on day level (see Sect. 4.2.2) can be adjusted for these shifts. However, there might be patterns for all shifts, except the absence shifts, for example: "A night can only be followed by a night shift or a special shift". As a consequence we lose these patterns in the first phase of days-off-scheduling, which for the example will lead to scheduling night shifts in the middle of stints.
Another example is in the instance QMC-1. It defines the shift group 'ALL' that contains all shifts except shift type O (the day off shift). In addition the constraints on stints length

and work load use the shift group 'ALL'. Consequently the constraints can not be used in the (On, Off)-decomposition. In fact it can only be used in the (Day, Night, Off)-approach if O is selected as night shift. This explains why O as special shift gives the best results.
– **Skilled shifts**: The BCV instances and CHILD contain shifts that should preferably be assigned to specific employees; information that can not be included in the first phase of days-off-scheduling. Consequently we create stints for these specific employees that do not match these shifts. In BCV-4.13.1 we see that the DH shift type gives the best result in the (Day, Night, Off)-approach; remember that BCV-4.13.1 does not contain a night shift.

6.1.3 Coverage requirements

Section 4.2.2 described how coverage requirements in combination with skills or time units are handled. Unfortunately this is not always the best approach. In particular, the Ikegami instances contain complicated skill coverage requirements and in combination with the strong preferences on shift sequences the results are unsatisfactory in these instances. Indeed the coverage models we solve to calculate the number of required employees, might lead to the same group of employees for all days, which unfortunately will not be feasible to schedule. On the other hand, we have introduced more skill details into the coverage requirements of the instance Ozkarahan, leading to a better result on that instance, see Sect. 5. These covers are related to skills for disjoint groups of employees, which explains why it works very well here.

6.1.4 The night shift (reprise)

One of the main conclusions of this study is that night shifts need special attention. However, it is sometimes difficult to incorporate all effects of night shifts. In particular, several instances allow the night shift to be followed only by some specific other shift (a leave shift, a late shift or another night shift), see also 6.1.2. In such cases it might be better to ignore the night shift, and use the (On, Off)-approach instead, such as in BCV-3.46.2, ERRVH and the Ikegami-3Shifts cases. Considering the night shift might lead to scheduling it in places that are difficult to accommodate in the second phase.

6.1.5 Shift sequences

Usually there are patterns describing which shift sequences are preferable. In some cases the violation of these preferences can not be avoided in the second phase of the decomposition, for example (again) in the Ikegami-3Shifts cases.

6.1.6 Run aborted

As highlighted there are some instances where the (Day, Night, Off)-approach stopped before reaching optimality. In these cases the run was aborted, due to the time restriction or because of running out of memory. For the instance MER none of the approaches finds a reasonable solution. Due to the memory requirements the direct approach ends after 628 s, explaining why the decompositions take longer. In some other instances (ORTEC01 and ORTEC02) the solver almost reached optimality, or at least better solutions than the direct approach (BCDT-Sep and WHPP).

Table 6 Comparisons to other approaches

Instance	Burke and Curtois		Ikegami and Niwa		(Day, Night, Off)		
	Cost	Time (sec)	Cost	Time (sec)	Night shift	Cost	Time (sec)
Azaiez	0*	0	–	–	–	6	2
GPost	5*	2	–	–	–	4,002	4
GPost-B	3*	29	–	–	–	2,002	32
Millar-2shift-1	0*	0	0*	1	–	0*	0
Millar-2shift-1.1	0*	0	–	–	–	0*	0
Ozkarahan	0*	0	–	–	–	0*	0
BCDT-Sep	100*	6,240	–	–	N	2,350	533
BCV-3.46.2	894*	8	–	–	L	2,374	1
BCV-4.13.1	10*	893	–	–	DH	10*	1
CHILD	149*	20	–	–	N	3,847	5
ERMGH	779*	2	–	–	N	1,116	4
ERRVH	2,001*	976	–	–	N	22,796	34
Ikegami-2Sh-1	0*	42	0*	13	N	0*	21
Ikegami-3Sh-1	2*	598	6	346,981	N	3,884	6
Ikegami-3Sh-1.1	4	995	13	41,626	N	2,728	331
Ikegami-3Sh-1.2	5	5,412	12	54,142	N	1,960	526
LLR	301*	1	–	–	N	308	0
MER	7,081	36,003	–	–	N	142,133	734
ORTEC01	270*	69	–	–	N	280	104
ORTEC02	270*	105	–	–	N	275	83
QMC-1	13*	58	–	–	O	177	0
QMC-2	29*	2	–	–	N	33	1
SINTEF	0*	11	–	–	N	17	0
Valouxis-1	80	910	–	–	N	20*	89
WHPP	5*	18	–	–	N	3,001	34

6.1.7 Comparisons to other approaches

Finally, we have included the results of two other approaches that have previously been applied to the benchmark instances in Table 6. The algorithm in Ikegami and Niwa (2003) also decomposes the problem to day-off scheduling. However it also splits up the planning horizon into shorter periods and solves subproblems for individual employees. It uses enumeration and metaheuristics. The branch and price method of Burke and Curtois (2014) can also be classified as a decomposition method because it involves iteratively solving many subproblems equivalent to the scheduling of individual employees (like Ikegami). To date this algorithm has produced many best and equal best results on the benchmark instances.

The results of Burke and Curtois (2014) are better in quality in 19 of the 25 instances, for 5 instances both approaches found the optimal solution, and for one instance (Valouxis-1) the days-off-decomposition gives the best result. Direct time comparisons are difficult because the experiments were done on different machines. However, from Table 6 it is apparent that for some instances the days-off-decompositions finds better or comparable results with less

computational effort. This shows that days-off-decomposition has practice value, but that its contribution is dependent on the specifics of the scheduling instance.

7 Conclusions

We studied the effects of solving the personnel scheduling problem by decomposition approaches. The first approach, (On, Off), is a two-phase decomposition approach that first assigns employees to working days and days off, and secondly assigns employees to shifts on the working days. The second approach, (Day, Night, Off), additionally includes night shifts in the first phase of the decomposition. Both phases of the decomposition approaches are solved using mathematical programming, and evaluated using public personnel scheduling benchmark instances. The results of the decomposition approaches are compared against solving a mathematical programming formulation of the personnel scheduling directly.

We see that the decomposition has a large impact on the solving time: for most instances the solving time is reduced by more than 80 % or 90 %. However, the solutions do not give such a clear answer. One can roughly say that in 1/3 of the instances the (On, Off)-approach gives good results, in 1/3 of the instances the (Day, Night, Off)-approach gives good results, and in 1/3 of the instances this decomposition does not give competitive results. Especially for the (Day, Night, Off)-approach we have to pay attention to problem specific aspects.

An advantage of the decompositions is that they are relatively simple to implement. We also discussed several improvements for our implementations. Significant improvements could be obtained by aggregating scheduling constraints and by alternative modeling of skill-related constraints. This re-modeling enabled us to effectively consider these constraints in the first phase of the decompositions resulting in improved overall results. Moreover, in our approach we currently use necessary constraints but including additional, stronger, constraints may also improve the results.

Since the first phase usually has multiple optimal solutions, we analyzed the effect different solutions of the first phase had on the outcome of the second phase. This had no significant positive or negative effect on the originally obtained solutions, which implies that the presented results are a representative sample.

In conclusion, we conclude that applying the decomposition approaches significantly improves the required computation time, but they should be implemented with care. For example choosing the 'best' night shift, in the (Day, Night, Off)-approach requires some attention. We noticed that some instances contain special shifts that are so special that it would be better to pre-assign them, like shift type ET in the instance CHILD, that can only be assigned to two employees. By tailoring the decomposition approach to the instance (class) at hand, we would expect our decomposition approach to be successful for many personnel scheduling instances. This assessment gives clear indications of the potential strengths and weaknesses of days-off-scheduling.

References

Abdennadher, S., Schlenker, H. (1999). Nurse scheduling using constraint logic programming. In *Proceedings of the 1998 Winter Simulation Conference Proceedings* (pp. 838–843).

Azaiez, M. N., & Al Sharif, S. S. (2005). A 0–1 goal programming model for nurse scheduling. *Computers & Operations Research*, *32*(3), 491–507.

Baxter, J., & Mosby, M. (1988). Generating acceptable shift-working schedules. *The Journal of the Operational Research Society*, *39*(6), 537–542.

Brucker, P., Burke, E., Curtois, T., Qu, R., & Vanden Berghe, G. (2010). A shift sequence based approach for nurse scheduling and a new benchmark dataset. *Journal of Heuristics*, *16*(4), 559–573.
Burke, E. K., & Curtois, T. (2014). New approaches to nurse rostering benchmark instances. *European Journal of Operational Research*, *237*(1), 71–81.
Burke, E. K., de Causmaecker, P., van Landeghem, H., & vanden Berghe, G. (2004). The state of the art of nurse rostering. *Journal of Scheduling*, *7*(6), 441–499.
Cheang, B., Li, H., Lim, A., & Rodrigues, B. (2003). Nurse rostering problems-a bibliographic survey. *European Journal of Operational Research*, *151*(3), 447–460.
Curtois, T. (2007). Employee scheduling benchmark data sets. http://www.cs.nott.ac.uk/~tec/NRP/. Accessed June 2014.
Day, P. R., & Ryan, D. M. (1997). Flight attendant rostering for short-haul airline operations. *Operations Research*, *45*(5), 649–661.
Ernst, A., Jiang, H., Krishnamoorthy, M., Owens, B., & Sier, D. (2004). An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research*, *127*(1), 21–144.
Gärtner, J., Musliu, N., & Slany, W. (2001). Rota: A research project on algorithms for workforce scheduling and shift design optimization. *AI Communications*, *14*(2), 83–92.
Glass, C. A., & Knight, R. A. (2010). The nurse rostering problem: A critical appraisal of the problem structure. *European Journal of Operational Research*, *202*(2), 379–389.
Ikegami, A., & Niwa, A. (2003). A subproblem-centric model and approach to the nurse scheduling problem. *Mathematical Programming*, *97*(3), 517–541.
Maenhout, B., & Vanhoucke, M. (2010). Branching strategies in a branch-and-price approach for a multiple objective nurse scheduling problem. *Journal of Scheduling*, *13*(1), 77–93.
Mason, A., Smith, M. (1998). A nested column generator for solving rostering problems with integer programming. In *Proceedings of International Conference on Optimisation: Techniques and Applications* (pp. 827–834).
Millar, H., & Kiragu, M. (1998). Cyclic and non-cyclic scheduling of 12 h shift nurses by network programming. *European Journal of Operational Research*, *104*(3), 582–592.
Nurmi, K., Kyngäs, J., & Post, G. (2011). Driver rostering for bus transit companies. *Engineering Letters*, *19*(2), 125–132.
Parr, D., & Thompson, J. (2007). Solving the multi-objective nurse scheduling problem with a weighted cost function. *Annals of Operations Research*, *155*(1), 279–288.
Santos, H.G., Toffolo, T., Ribas, S., Gomes, R. (2012). Integer programming techniques for the nurse rostering problem. In *Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling* (pp. 257–283).
Valouxis, C., Gogos, C., Goulas, G., Alefragis, P., & Housos, E. (2012). A systematic two phase approach for the nurse rostering problem. *European Journal of Operational Research*, *219*(2), 425–433.
Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., & De Boeck, L. (2013). Personnel scheduling: A literature review. *European Journal of Operational Research*, *226*(3), 367–385.
Van Veldhoven, S. (2011). Days off personnel scheduling. Master's thesis, University of Twente, The Netherlands.