manuscript No.
(will be inserted by the editor)

An assessment of a days off decomposition approach to personnel scheduling

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

Received: date / Accepted: date

Abstract This paper studies a two-phase decomposition approach to solve 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 in 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, and that they produce good solutions for most instances.

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

Gerhard Post (corresponding author)
Department of Applied Mathematics, University of Twente, P.O. Box 217, Enschede, The Netherlands, and ORTEC, Groningenweg 6k, 2803 PV Gouda, The Netherlands,
E-mail: g.f.post@utwente.nl

Egbert van der Veen
Department of Applied Mathematics, University of Twente, P.O. Box 217, Enschede, The Netherlands, and ORTEC, Groningenweg 6k, 2803 PV Gouda, The Netherlands,
E-mail: Egbert.vanderVeen@ortec.com

Tim 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 for 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 (ILP) to the subproblems.

The contribution of this paper is that we investigate the days off decomposition on 25 nurse rostering benchmark instances found at Curtois (2007). In addition we study the effect of specifying the night shift during decomposition. Our generic methods make clear what the potential and pitfalls of these decompositions are and make it possible to give a detailed analysis of the issues that are important in this decomposition.

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

2 Literature review

There is a vast amount of literature on personnel scheduling, see the recent review by den Bergh et al (2013), and the reviews by Ernst et al (2004), Burke et al (2004) and Cheang et al (2003). Within personnel scheduling literature days off scheduling has received considerable attention. In this section we review days off scheduling literature and indicate how the decomposition approach as proposed in this paper contributes to existing literature.

In literature there has been a particular focus on *cyclic* days off scheduling. Alfares (1998, 2001); Baker (1974a); Bartholdi et al (1980); Bartholdi (1981); Bechtold (1981); Rothstein (1973); Tibrewala et al (1972) propose algorithms to determine weekly repeating days off schedules for which employees have in each week 5 consecutive working days and 2 consecutive days off. The objective is to determine the minimum size workforce or the minimum cost workforce. These days off scheduling problems are polynomially solvable, since they can be modeled as network flow problems, see Bartholdi et al (1980). Baker (1974b) extends on this by including part-time workers, which may work less than 5 consecutive shifts and Alfares (2002) determines the minimum workforce in the situation where in each 3-week period an employee has 14 consecutive working days and 7 consecutive days off.

Baker and Magazine (1977); Brownell and Lowerre (1976); Hung (1991); Lowerre (1977) determine the minimum workforce size for cyclic days off scheduling with various days off policies and workforce demand being N on weekdays and n during weekends. Emmons (1985) extends on this by implying that an employee has at least A out of B weekends off, and Emmons and Fuh (1997) include part-time workers who work less shifts than full-time workers. Baker et al (1979) also include constraints on the number of working weekends, but assume staff requirements are the same each day, whereas Burns and Carter (1985); Burns et al (1998); Pedrosa and Constantino (2001) allow staff requirements to be different *each* day. Maier-Rothe and Wolfe (1973); Rosenbloom and Goertzen (1987) first determine a set of schedules, given a set of constraints, which are afterwards assigned to employees. Emmons and Burns (1991); Hung (1994); Narasimhan (1997) determine the minimum cost workforce for hierarchical multi-skilled workforces. With hierarchical skills there is an ordering of skills such that a higher skilled more expensive worker can always do the work of a lower-skilled less-expensive worker.

Acyclic days off scheduling has also received considerable attention. Azmat and Widmer (2004); Azmat et al (2004); Hung (1999) consider a scheduling horizon of one year, with constraints on the number of shifts per week and during the scheduling horizon. Morris and Showalter (1983); Alfares (2003) consider constraints on the number of (consecutive) working days and days off under the objective to minimize the workforce size, whereas Elshafei and Alfares (2008) minimize the workforce cost, and Beaumont (1997); Miller et al (1976) minimize the penalty implied by violations of scheduling criteria. Costa et al (2006) study a feasibility problem under numerous days off scheduling constraints.

In this paper we solve the personnel scheduling problem by first solving the days off scheduling problem and then assigning shifts to employees on their working days. This is also done by Baxter and Mosby (1988); Day and Ryan (1997); Gärtner et al (2001). Baxter and Mosby (1988) use heuristics to solve the subproblems whereas we use ILP to get optimal solutions for the subproblems. Day and Ryan (1997); Gärtner et al (2001) solve the subproblems by enumerating day off patterns and solving partitioning problems. Day and Ryan (1997) solve the day off scheduling problem 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); Day and Ryan (1997) include a re-scheduling method in their approaches. Abdennadher and Schlenker (1999) decompose the personnel scheduling problem to three phases: day 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. Parr and Thompson (2007) also decompose the personnel scheduling 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. Parr and Thompson (2007) compare two alternative local search methods to solve these phases.

Bailey (1985); Bechtold and Brusco (1994) combine days off scheduling with shift scheduling. Shift scheduling determines the set of shifts that should be assigned to employees. Bailey (1985); Bechtold and Brusco (1994) start with shift scheduling, then generate a days off schedule, and finally assign shifts to employees on their working days.

3 Problem description

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

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.
- **Cover 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 an ILP. 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 straightforward. The details of these formulations are not described in this paper. However, to provide the reader insight in these formulations, Section 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. They represent a diverse collection of challenging, real-world instances. Because they were drawn from many sources they vary in dimensions such as the number of staff, shifts 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 effected 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 at 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 7079 was established

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	2001
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	7081
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
Valouxis-1	16	3	28	20
WHPP	30	3	14	5

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, Section 4.1.1 outlines how we formulate and solve the days off scheduling problem for the benchmark instances described in Section 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 Section 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 equation (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 2 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). For this, 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 2 shift types (On, Off) and 3 shift types (Day, Night, Off), respectively. Hence, to solve the first and second phase of the decomposition we can use similar ILP formulations, where the employee scheduling problem of the second phase should obey the decisions of the first phase. Although the ILP models 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.

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. For the implementation it is important to incorporate pre-assigned shifts and requests in the possible matches of the pattern constraints, see Section 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 Cover requirements

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

In both cases the information can be more detailed expressing that the employees counting for the requirement should have certain skills. Employees having certain skills can count for more than one cover requirement at the same time. Therefore, we formulate a small ILP model to determine the minimum and maximum number of available employees that satisfy the cover requirements. These bounds are used in constraints in the days off scheduling problem that express per day the bounds on the number of employees with certain skills.

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. We illustrate this by an example that is taken from the XML representation of the Azaiez instance:

```
<Match>
 <Max>
   <Count>4</Count>
   <Weight>10000</Weight>
 </Max>
 <Pattern>
    <StartDay>Saturday</StartDay>
    <Shift>$</Shift>
  </Pattern>
  <Pattern>
    <StartDay>Sunday</StartDay>
    <Shift>$</Shift>
  </Pattern>
<Match>
```

The first pattern is for Saturdays; each day that an employee has any shift ('$') on a Saturday we have a match. Similarly, the second pattern is for Sundays. Hence, the match counts the number of Saturdays and Sundays on which a shift starts. In Azaiez we see that an employee should do at most 4 (Count) weekend shifts; violation of this constraint results in a penalty of 10000 (Weight) per extra shift. 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, or the minimum number of consecutive shifts.

If a pattern concerns all shift types, we can use it in the days off schedule. We implemented combining different pattern constraints to one days off pattern constraint, but only for patterns of length 1. 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 a stint, 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 ILP. For example if employee e has a maximum workload of 120 hours, and shortest shift contains 7 hours of work, then we add the constraint

$$7 \cdot \sum_{d} y_{ed} \leq 120,$$

implying that the employee will have at most 17 working days.

In the personnel scheduling phase we have full information, so that we can use the correct workload requirements.

5 Results

To evaluate our decomposition approach, we solve the ILP 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 loosing the optimality guarantee. The ILP formulations of the two phases of the decomposition are also solved using CPLEX, with the time limit set to 30 minutes for each phase.

CPLEX finishes with 4 different statuses: status 101 and 102, indicating that the optimal solution was found, status 107, indicating that the time limit was reached without (guaranteed) optimal solution and status 109 indicating that CPLEX ran out of memory[2].

Table 2 and Table 3 present results for instances with 2 shift types and instances with more than 2 shift types, respectively. To the instances with 2 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 third column.

First, we analyze the results of the (On, Off) decomposition approache to the instances with 2 shift types, and after that we analyze the results of our decomposition approaches to the instances with more than 2 shift types.

Instances with 2 shift types

The fourth column in Table 2 gives the solution value 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 seconds (3.6 % of 117 seconds).

[2] see http://www-01.ibm.com/support/docview.wss?uid=swg21399943

Table 2 Results on instances with 2 shift types

	Direct		(On, Off)	
instance	cost	time (sec)	cost	% time saving
Azaiez	0*	7.4	6	76.7
GPost	5	117	4002	96.4
GPost-B	3*	428	2002	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

We observe that the decomposition is solved faster than the direct approach. However, except for the Millar instances the results 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 cover 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, see 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.

Instances with more than 2 shift types

For the instances with more than 2 shift types, the results of both decomposition approaches are presented in Table 3. Again results are compared with the direct approach. The first 5 columns in Table 3 are the same as the first five columns in Table 2. Column 6 indicates the shift that is chosen as "night shift". For two instances (BCV-3.46.2 and QMC-1) choosing N as the night shift did not give the best result. For these instances, we added extra rows for the "night shift" choice 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 "night 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. Note that in the cases where the optimal solution is not in the direct approach, and the time listed is less than 3600 seconds, the process ran out of memory. The decomposition yields better solutions in 5 instances, and in 2 instances the solution quality is

Table 3 Results on instances with more than 2 shift types

	Direct		(On, Off)		(Day, Night, Off)		
instance	cost	time (sec)	cost	% time saving	night shift	cost	% time saving
BCDT-Sep	104539	3602	104229	99.9	N	2350	85.2
BCV-3.46.2	894*	535	1215	99.7	N	3430	82.8
BCV-3.46.2					L	2374	99.8
BCV-4.13.1	10*	1.9	17	65.9	DH	10*	54.2
CHILD	149*	47	5066	86.9	N	3847	89.9
ERMGH	779*	1.7	929	27.9	N	1116	-119.0
ERRVH	2204	385	12832	97.9	N	22796	91.3
Ikegami-2Sh-1	0*	27	4316	97.7	N	0*	23.1
Ikegami-3Sh-1	110	3601	807	90.8	N	3884	99.8
Ikegami-3Sh-1.1	6	3601	994	94.8	N	2728	90.8
Ikegami-3Sh-1.2	23	3602	994	90.8	N	1960	85.4
LLR	301*	1.2	312	80.2	N	308	92.6
MER	56561	628	122022	-187.0	N	142133	-16.8
ORTEC01	3527	1365	2270	94.1	N	280	92.4
ORTEC02	2836	2385	1275	96.5	N	275	96.5
QMC-1	13*	3.2	29046	77.7	N	21070	38.8
QMC-1					O	177	86.9
QMC-2	29*	0.4	1045	30.8	N	33	-92.1
SINTEF	0*	1.6	12202	77.9	N	17	77.9
Valouxis-1	140	1461	20*	64.5	N	20*	93.9
WHPP	3008	155	16000	99.5	N	3001	77.7

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 biggest instance. On this instance none of the models give a satisfactory result. The direct model ends after 628 seconds (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. In the next section we explain this behavior.

5.1 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 days off scheduling.** 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 re-

Table 4 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	×					×

formulate 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, cover requirements with skills, and strong preferences for shift sequences.

Our findings are summarized in Table 4. The contents of the columns 'aggr', 'shift', 'cover', 'night', 'seq' and 'run' are addressed in Sections 5.2.1-5.2.6, respectively.

5.2 Detailed analysis ot the results

5.2.1 Aggregate constraints

The '×' in column 'aggr' of Table 4 indicates that a set of constraints in the instance was taken together to an equivalent instance with one aggregated constraint. 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 following by the *shift group* consisting of the types E, D, L, etc. In our tests we used these aggregated versions which are also available at Curtois (2007).

5.2.2 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 cover requirements, but can count as work (in the Ikegami instances and ORTEC02) or not (QMC-1 and BCDT-Sep). The cover requirements on day level (see Section 4.2.2) can be adjusted for these shifts. However, there might exist 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.
- **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 don't 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 doesn't contain a night shift.

5.2.3 Cover requirements

Section 4.2.2 described how cover 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 skilled cover requirements and in combination with the strong preferences on shift sequences the results are unsatisfactory in these instances. Indeed the cover 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 introduced more skill details in the cover requirements of the instance Ozkarahan, leading to a better result on that instance, see Section 5. These covers are related to skills for disjoint groups of employees, which explains why it works very well here.

5.2.4 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 by some specific other shift (a leave shift, a late shift or another night shift), see also 5.2.2. In such cases it might be better to ignore the night shift, and use the (On, Off)-approach instead, like in BCV-3.46.2, ERRVH and the Ikegami-3Shifts cases.

5.2.5 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.

5.2.6 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 seconds, 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).

6 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 ILP, and evaluated using public personnel scheduling benchmark instances. The results of the decomposition approaches are compared against solving an ILP 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.

We 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 sec-

ond phase. This had no significant positive or negative effect on the originally obtained solutions. Hence, we conclude that the presented results are a representative sample.

In view of the observation that CPLEX solves 15 instances to optimality within one hour, it is in several cases easier to first generate the shift schedule, and use this to publish the days off schedule, as suggested in Bailey (1985); Bechtold and Brusco (1994). In general we can state that applying the days off decomposition approach should be implemented with care. With a detailed analysis of the instance (class) at hand, we can expect this approach to be successful on most instances.

References

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

Alfares HK (1998) An efficient two-phase algorithm for cyclic days-off scheduling. Computers & Operations Research 25(11):913–923

Alfares HK (2001) Efficient optimization of cyclic labor days-off scheduling. OR Spectrum 23(2):283–294

Alfares HK (2002) Optimum workforce scheduling under the (14, 21) days-off timetable. Journal of Applied Mathematics and Desicision Sciences 6(3):191–199

Alfares HK (2003) Four-day workweek scheduling with two or three consecutive days off. Journal of Mathematical Modelling and Algorithms 2(1):67–80

Azmat CS, Widmer M (2004) A case study of single shift planning and scheduling under annualized hours: A simple three-step approach. European Journal of Operational Research 153(1):148–175

Azmat CS, Hürlimann T, Widmer M (2004) Mixed integer programming to schedule a single-shift workforce under annualized hours. Annals of Operations Research 128(1):199–215

Bailey J (1985) Integrated days off and shift personnel scheduling. Computers & Industrial Engineering 9(4):395–404

Baker KR (1974a) Scheduling a full-time workforce to meet cyclic staffing requirements. Management Science 20(12):1561–1568

Baker KR (1974b) Scheduling full-time and part-time staff to meet cyclic requirements. Operational Research Quarterly (1970-1977) 25(1):65–76

Baker KR, Magazine MJ (1977) Workforce scheduling with cyclic demands and day-off constraints. Management Science 24(2):161–167

Baker KR, Burns RN, Carter M (1979) Staff scheduling with day-off and workstretch constraints. A I I E Transactions 11(4):286–292

Bartholdi JJ (1981) A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering. Operations Research 29(3):501–510

Bartholdi JJ, Orlin JB, Ratliff HD (1980) Cyclic Scheduling via Integer Programs with Circular Ones. Operations Research 28:1074–1085

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

Beaumont N (1997) Using mixed integer programming to design employee rosters. Journal of the Operational Research Society 48(6)

Bechtold SE (1981) Work force scheduling for arbitrary cyclic demands. Journal of Operations Management 1(4):205–214

Bechtold SE, Brusco MJ (1994) A microcomputer-based heuristic for tour scheduling of a mixed workforce. Computers & Operations Research 21(9):1001–1009

den Bergh JV, Belien J, Bruecker PD, Demeulemeester E, Boeck LD (2013) Personnel scheduling: A literature review. European Journal of Operational Research 226(3):367–385

Brownell WS, Lowerre JM (1976) Scheduling of work forces required in continuous operations under alternative labor policies. Management Science 22(5):597–605

Burke E, de Causmaecker P, vanden Berghe G, van Landeghem H (2004) The state of the art of nurse rostering. Journal of Scheduling 7(6):441–499

Burns RN, Carter MW (1985) Work force size and single shift schedules with variable demands. Management Science 31(5):599–607

Burns RN, Narasimhan R, Smith LD (1998) A set-processing algorithm for scheduling staff on 4-day or 3-day work weeks. Naval Research Logistics (NRL) 45(8):839–853

Cheang B, Li H, Lim A, Rodrigues B (2003) Nurse rostering problems–a bibliographic survey. European Journal of Operational Research 151(3):447–460

Costa MC, Jarray F, Picouleau C (2006) An acyclic days-off scheduling problem. 4OR: A Quarterly Journal of Operations Research 4(1):73–85

Curtois T (2007) Employee scheduling benchmark data sets. http://www.cs.nott.ac.uk/∼tec/NRP/

Day PR, Ryan DM (1997) Flight attendant rostering for short-haul airline operations. Operations Research 45(5):649–661

Elshafei M, Alfares H (2008) A dynamic programming algorithm for days-off scheduling with sequence dependent labor costs. Journal of Scheduling 11(2):85–93

Emmons H (1985) Work-force scheduling with cyclic requirements and constraints on days off, weekends off, and work stretch. IIE Transactions 17(1):8–16

Emmons H, Burns RN (1991) Off-day scheduling with hierarchical worker categories. Operations Research 39(3):484–495

Emmons H, Fuh DS (1997) Sizing and scheduling a full-time and part-time workforce with off-day and off-weekend constraints. Annals of Operations Research 70(0):473–492

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 CA, Knight RA (2010) The nurse rostering problem: A critical appraisal of the problem structure. European Journal of Operational Research 202(2):379 – 389
Hung R (1991) Single-shift workforce scheduling under a compressed workweek. Omega 19(5):494–497
Hung R (1994) Single-shift off-day scheduling of a hierarchical workforce with variable demands. European Journal of Operational Research 78(1):49–57
Hung R (1999) Scheduling a workforce under annualized hours. International Journal of Production Research 37(11):2419–2427
Lowerre JM (1977) Work stretch properties for the scheduling of continuous operations under alternative labor policies. Management Science 23(9):963–971
Maier-Rothe C, Wolfe HB (1973) Cyclical scheduling and allocation of nursing staff. Socio-Economic Planning Sciences 7(5):471–487
Miller HE, Pierskalla WP, Rath GJ (1976) Nurse scheduling using mathematical programming. Operations Research 24(5):857–870
Morris JG, Showalter MJ (1983) Simple approaches to shift, days-off and tour scheduling problems. Management Science 29(8):942–950
Narasimhan R (1997) An algorithm for single shift scheduling of hierarchical workforce. European Journal of Operational Research 96(1):113–121
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, 10.1007/s10479-007-0202-4
Pedrosa D, Constantino M (2001) Days-off scheduling in public transport companies. In: VoßS, Daduna JR (eds) Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems, vol 505, Springer Berlin Heidelberg, pp 215–232
Rosenbloom E, Goertzen N (1987) Cyclic nurse scheduling. European Journal of Operational Research 31(1):19–23
Rothstein M (1973) Hospital manpower shift scheduling by mathematical programming. Health Services Research 8(1):60–66
Tibrewala R, Philippe D, Browne J (1972) Optimal scheduling of two consecutive idle periods. Management Science 19(1):71–75