Network Flow Models for Intraday Personnel Scheduling Problems

Peter Brucker[1] and Rong Qu[2]

[1]Universität Osnabrück, Albrechtstr. 28a, 49069 Osnabrück, Germany, e-mail: pbrucker@uni-osnabrueck.de
[2]Automated Scheduling, Optimization and Planning (ASAP) Group, School of Computer Science, University of Nottingham, NG8 1BB, UK

ABSTRACT: Personnel scheduling problems can be decomposed into two stages. In the first stage for each employee the working days have to be fixed. In the second stage for each day of the planning period an intraday scheduling problem has to be solved. It consists of the assignment of shifts to the employees who have to work on the day and for each working period of an employee a task assignment such that the demand of all tasks for personnel is covered. Robinson et al. [3] formulated the intraday problem as a maximum flow problem under the following assumptions: employees are qualified for all tasks, their shifts are given, and they are allowed to change tasks during the day.

We show that the network flow model can be extended to cover the case in which employees are not qualified to perform all tasks. Further extensions allow to calculate shifts of employees for the given day under the assumption that an earliest starting time and a latest finishing time as well as a minimal working time are given. Also labour cost can be taken into account by solving a minimum cost network flow problem.

KEYWORDS: personnel scheduling, assignment problem, network flows

1 Introduction

A general personnel scheduling problem can be formulated as follows.

There is a planning horizon consisting of a number of consecutive days. Associated with each day is a set of periods in which certain tasks have to be performed. For each period of a day and task which has to be performed in this period employees are needed.

The planning horizon has to be divided into working days and rest days for each employee. A shift has to be assigned to each working day of an employee. *Shifts* consist of a set of *working periods* possibly interrupted by breaks and idle times which are part of the shift.

For each employee there is a set of tasks he can be assigned to.

A *working pattern* is defined by the set of working days and for each working day a shift. A working pattern is feasible for an employee if it satisfies a number of constraints.

One has to assign

- to each employee a feasible working pattern, and
- to each working period of this pattern a task to be performed by the employee.

This has to be done in such a way that

- all tasks can be performed (i.e. the demand of tasks for employees is satisfied), and
- corresponding costs are minimized.

The model has two levels which we denote by *days scheduling* and *intraday scheduling* level. At the days level one has to assign working days to employees while at the intraday level for each employee working on the day one has to assign a shift and to each working period of this shift a task for which the employee is qualified.

One can differentiate between preemptive and non-preemptive problems. In a preemptive problem employees may change the working place during a shift. This is not allowed in non-preemptive versions.

Robinson et al. ([3]) considered the personnel scheduling problem under the assumption that

- preemption is allowed, and
- each employee can perform each task.

They applied tabu search to find good working patterns for the employees, and given the working patterns they solved the problem of assigning tasks to the active periods of each employee by maximum flow algorithms.

A network flow model for a special non-preemptive personnel scheduling problem is discussed in [4].

This paper is organized as follows. The maximum flow model of Robinson et al. ([3]) is presented in Section 2, followed by the extended network flow model in Section 3. In Section 4 we present further extensions concerning demand and supply sides of the network model we build in Section 3. The last section contains concluding marks.

2 The maximum flow formulation of Robinson et al.

The intraday personnel scheduling problem of Robinson et al. ([3]) can be described as follows.

On each day a subset of employees is available. Each employee e working on a fixed day is available during some time window $[S_e, F_e[$. A shift of employee e is a time interval $[V_e, W_e[$ with $S_e \leq V_e \leq W_e \leq F_e$ and $W_e - V_e \geq m_e$ where m_e is a given minimal shift length. During each period within a shift the employee performs a task, or has a (long or short) break, or is idle. There are maximal or minimal time distances between V_e, W_e ,the starting times, or finishing times of breaks. Breaks are non-preemptive.

There are n tasks $j = 1, \cdots, n$. Each task j has a duration p_j and must be processed by exactly one employee within a time window $[R_j, D_j[$ with $D_j - R_j \geq p_j$. Preemption is allowed, i.e. different employees may perform a task and an employee may perform different tasks on a day. Also interruption and later consideration of a task is possible. However, the total processing of task j must be equal to p_j.

Each employee can be assigned to any task.

One has to assign feasible shifts to the employees and for each shift to assign tasks to its active periods such that

- *the duration of each task is covered within its time window, and*
- *the total labor costs are minimal.*

The labor costs are defined as follows: meal breaks are unpaid. Short rest breaks are compensated. An overtime rate is paid for the time of a shift exceeding a given limit M. If an employee is not given at least two days off for a week then there is an additional pay.

Under the assumption that for each employee a shift has been fixed the problem can be formulated as a maximum flow problem with the following data.

Let T be the set of all R_j- and D_j- values, and all block starting and finishing times for all employees working on the day (blocks are maximal sets of consecutive working periods of a shift). Denote by $t_1 < t_2 < ... < t_s$ the ordered sequence of all elements in T.

The network (V, A) can be constructed as follows. The set V of nodes consists of

- task nodes $j = 1, \cdots n$,
- interval nodes $[t_i, t_{i+1}[$ $(i = 1, \cdots, s-1)$, and
- a source s and a sink t.

There are three different types of directed arcs:

- arcs (s, j) with upper capacity p_j,
- arcs $([t_i, t_{i+1}[, t)$ with upper capacity $(t_{i+1} - t_i)N_i$ where N_i is the number of employees available in time period $[t_i, t_{i+1}[$,
- there is an arc between a task node j and an interval node $[t_i, t_{i+1}[$ if and only if $[t_i, t_{i+1}[\subseteq [R_j, D_j[$. The upper capacity of this arc is $t_{i+1} - t_i$.

$$
\begin{array}{ccccccc}
 & & \text{tasks} & & \text{intervals} & & \\
 & \le p_j & \vdots & \le t_{i+1}-t_i & \vdots & \le (t_{i+1}-t_i)N_i & \\
s & \xrightarrow{\qquad} & j & \xrightarrow{\qquad\qquad} & [t_i,t_{i+1}[& \xrightarrow{\qquad\qquad} & t \\
 & & \vdots & \text{iff } [t_i,t_{i+1}[\subseteq [R_j,D_j[& \vdots & &
\end{array}
$$

Figure 1: Network for the assignment of tasks to employees

The network is shown in Figure 1.

A flow in an arc $(j,[t_i,t_{i+1}[)$ may be interpreted as working time assigned to task j in the interval $[t_i,t_{i+1}[$. There exists a feasible task assignment if and only if the value of a maximal flow is equal to $\sum_{j=1}^{n} p_j$.

If there is a maximal flow with this property then in each task node j the processing time p_j is distributed to the time intervals $[t_i,t_{i+1}[$ in which j can be processed and the time j is processed in $[t_i,t_{i+1}[$ cannot exceed $t_{i+1}-t_i$. Furthermore, due to the flow-balance constraints in the interval nodes $[t_i,t_{i+1}[$ the sum of these processing times cannot exceed $(t_{i+1}-t_i)N_i$. It is well known (see e.g. [1] P. 108) that under these conditions it is possible to process the parts of tasks assigned to $[t_i,t_{i+1}[$ by N_i employees if preemption is allowed.

Robinson et al. describe a tabu search heuristic for calculating shifts for the employees for a time horizon of several days and corresponding assignments to tasks. The tabu search can be described as follows.

A working pattern of an employee consists of all shifts assigned to the employee within the time horizon. A solution consists of the working pattern of all employees. A solution is feasible if it allows to cover the demand of all tasks on every day. Feasibility can be checked and task assignments can be calculated by solving a maximum flow problem for each day. The search is performed within the set of all feasible solutions.

The assumption that each employee can be assigned to any task is not always realistic. Therefore the model will be extended in the next section.

3 An extended network flow model

In this and later sections the assumption that employee e can perform only tasks $j \in Q_e \subseteq \{1,\cdots,n\}$ is added. A network which takes care of these additional constraints can be described as follows.

Again $t_1 < t_2 < ... < t_s$ are the time instances where the data are changing. The set of nodes of the network consists of

- task nodes $j = 1,\cdots,n$,
- interval-task nodes $[t_i,t_{i+1}[_j$ for all intervals $[t_i,t_{i+1}[$ with $[t_i,t_{i+1}[\subseteq [R_j,D_j[$,
- interval-employee nodes $[t_i,t_{i+1}[_e$ for all working intervals $[t_i,t_{i+1}[$ of employee e, and
- a source s and a sink t.

$$\begin{array}{ccccccccc} & & \text{tasks} & & \text{intervals} & & & & \\ & \le p_j & : & \le t_{i+1}-t_i & : & & : & \le (t_{i+1}-t_i) & \\ s & \longrightarrow & j & \longrightarrow & [t_i,t_{i+1}[_j & \longrightarrow & [t_i,t_{i+1}[_e & \longrightarrow & t \\ & & : & & : & \text{iff } j\in Q_e & : & & \\ & & : & & : & & : & & \\ & & : & \multicolumn{2}{c}{\text{iff } [t_i,t_{i+1}[\subseteq [R_j,D_j[} & & \multicolumn{2}{c}{\text{iff } [t_i,t_{i+1}[\text{ is a possible}} & \\ & & : & & : & & \multicolumn{2}{c}{\text{working period for e}} & \end{array}$$

Figure 2: Extended nerwork

There are four different types of arcs:

- arcs (s, j) with upper capacity p_j,
- arcs $(j, [t_i, t_{i+1}[_j)$ with upper capacity $t_{i+1} - t_i$,
- arcs $([t_i, t_{i+1}[_j,\ [t_i, t_{i+1}[_e)$ for $j \in Q_e$, and
- arcs $([t_i, t_{i+1}[_e, t)$ with upper capacity $t_{i+1} - t_i$.

The network is shown in Figure 2. A flow in an arc $([t_i, t_{i+1}[_j\ ,\ [t_i, t_{i+1}[_e)$ may be interpreted as the number of time units employee e is assigned to task j within the time interval $[t_i, t_{i+1}[$. The flow conservation constraint for node $[t_i, t_{i+1}[_j$ distributes the time spent on task j in $[t_i, t_{i+1}[$ among employees which are qualified to do task j. The flow conservation constraint for node $[t_i, t_{i+1}[_e$ limits the workload of employee e in $[t_i, t_{i+1}[$ by $t_{i+1} - t_i$. There exists a feasible assignment of employees to tasks if and only if the maximum flow is equal to $\sum_{j=1}^{n} p_j$.

The procedure is illustrated by the following example with two employees and three tasks.

Example 1 Consider a problem with the following data. Notice that in the time interval $[2, 3[$ employee e_1 has a break.

task j	1	2	3
R_j	0	3	4
D_j	6	7	6
p_j	4	2	2

employee e_i	shift	Q_i
e_1	$[0, 2[\,, [3, 6[$	$\{1, 2\}$
e_2	$[3, 7[$	$\{2, 3\}$

The corresponding network with a solution is presented in Figure 3. Figure 4 shows the Gantt chart of the solution. The relevant t_i values are $0, 2, 3, 4, 6, 7$. Employee e_2 is idle in period $[3, 4[$.

4 Further extensions

The model introduced in the previous section can be extended at the demand side and/or the supply side. Possible extensions will be discussed in this section.

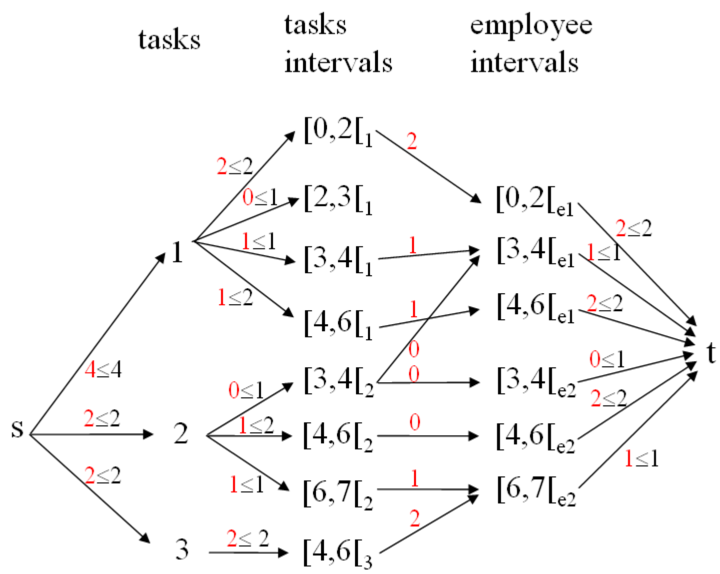

Figure 3: Network flow of Example 1

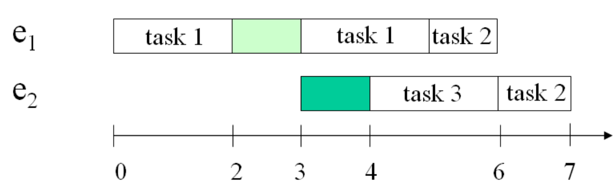

Figure 4: Gantt chart of the solution for Example 1

4.1 Extensions at the demand side

Instead of forcing the processing time of each task j to be equal to p_j by solving a corresponding maximum flow problem it is possible to enforce the constraint $LP_j \leq p_j \leq UP_j$ by the lower bound LP_j and the upper bound UP_j for the flow in the arc (s, j). In this case one has to find a feasible solution. If additionally costs are assigned to the arcs $([t_i, t_{i+1}[_e, t)$ one could minimize labour costs by solving a corresponding minimum cost network flow problem.

Another option is to replace

$$s \xrightarrow{\leq p_j} j \xrightarrow{\leq t_{i+1} - t_i} [t_i, t_{i+1}[_j$$

by

$$s \xrightarrow{\leq p_{ij}\,(t_{i+1} - t_i)} [t_i, t_{i+1}[_j$$

where p_{ij} is the number of employees needed for task j in the time interval $[t_i, t_{i+1}[$. Again one has to solve a maximum flow problem to cover the demand. Also by lower and upper bounds on the arcs $(s, [t_i, t_{i+1}[_j)$ the constraints $LD_{ij}\,(t_{i+1} - t_i) \leq p_{ij}\,(t_{i+1} - t_i) \leq UD_{ij}\,(t_{i+1} - t_i)$ can be enforced.

4.2 Extensions at the supply side

Instead of fixing the shift of employee e in advance one could fix only the availability interval $[S_e, F_e[$ and a minimal working time m_e for employee e. Then shifts for the employees which cover the demand of tasks can be calculated. To achieve this one has to replace

$$[t_i, t_{i+1}[_e \xrightarrow{\leq t_{i+1} - t_i} t$$

by

$$[t_i, t_{i+1}[_e \xrightarrow{\leq t_{i+1} - t_i} e \xrightarrow{\geq m_e} t$$

Due to node e and arc (e, t) the total working time of employee e cannot be smaller than m_e.

$$s \xrightarrow[]{\substack{\geq LD_{ij}\Delta_i \\ \leq UD_{ij}\Delta_i}} [t_i, t_{i+1}[_j \xrightarrow[\text{iff } j \in Q_e]{\leq \Delta_i} [t_i, t_{i+1}[_e \xrightarrow{\leq \Delta_i} e \xrightarrow{\geq m_e} t$$

Figure 5: Combined extensions

4.3 Combined extensions

The extensions at the demand and supply side can be combined. A possible combination is shown in Figure 5 where $\Delta_i := t_{i+1} - t_i$. A feasible network flow solution corresponds to a feasible shift and task assignment. Also overtime costs can be taken into account by assigning these overtime costs to the arcs (e, t), zero costs to all other arcs, and by solving the corresponding minimum cost network flow problem.

5 Concluding remarks

In this note we have shown that the problem of assigning shifts to employees and employees to tasks to cover the demand can be efficiently solved by network flow algorithms if preemption is allowed, even if employees are not qualified for all tasks. This can be exploited in heuristics for personnel scheduling problems for a time horizon of several days.

However, a side effect is that employees have to switch between tasks (working places) during their shifts. These switches depend on the constraints under which shifts are calculated and may be unavoidable. In connection with this the following working place change minimization (WPCM-) problem is of interest: *Assume that shifts have been assigned to all employees working on a given day. Then we call a task assignment for these employees feasible if the demand of all tasks for employees is covered. Find a feasible assignment which minimizes the number of working place changes.*

In [2] it has been shown that the WPCM-problem is NP-hard if possible shifts for e have the form $[t, t + p_e[$ $(t = 0, \cdots, P - p_e)$ where P is the number of working periods of the day. The complexity of the WPCM-problem for other ways of shift assignments is unknown.

Based on the present on the network flow models, extended investigations will be carried out in our future work to develop heuristic algorithms which assign feasible shifts to employees and construct (directly) preemptive schedules taking care of working place changes (e.g. by constructing good shifts). Numerical results will be reported and analysed on solving real world problems.

References

[1] Brucker, P. (2007), Scheduling Algorithms, Springer, Berlin.

[2] Brucker, P., Qu, R., Burke, E. (2008). Personnel scheduling: models and complexity, Working Paper, Automated Scheduling, Optimization and Planning (ASAP) Group, School of Computer Science, University of Nottingham.

[3] Robinson, R., Sorli, R., Zinder, Y. (2005), Personnel scheduling with time windows and preemptive tasks, In: E. K. Burke and M. Trick (editors), Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling, 18th August - 20th August 2004, Pittsburgh, PA USA: 561-566

[4] Segal, M. (1974), The operator scheduling problem: a network flow approach, Operations Research 22: 808-823.