Applying Tabu Search to the Rotating Workforce Scheduling Problem

Nysret Musliu*

*Technische Universität Wien.
Favoritenstrasse 9-11, 1040 Wien, Austria
musliu@dbai.tuwien.ac.at

In this paper we propose a tabu search approach for solving the rotating workforce scheduling problem. Workforce schedules affect both the health and the social life of the employees, and can also increase the risk of work-related accidents. Therefore, it is of a high practical relevance to find workforce schedules that, on the one hand, fulfill the ergonomic criteria for the employees, and, on the other, reduce costs for the organization. To solve the rotating workforce scheduling problem with tabu search we propose repair steps (moves) to explore the neighborhood of solutions for this problem. In order to accept the solution for the next iteration, basic principles of the tabu search technique are used, where we experimented with two variants of making solutions tabu. Additionally, we propose the fitness function for this problem. Computational results in the benchmark examples from the literature show that the tabu search approach gives good results for this problem.

1 Introduction

A workforce schedule represents the assignments of the employees to the defined shifts for a period of time. In Table 1 a typical representation of workforce schedules is presented. This schedule describes explicitly the working schedule of 9 employees during one week. The first employee works from Monday until Friday in a day shift (D) and during Saturday and Sunday has days-off. The second employee has a day-off on Monday and works in a day shift during the rest of the week. Further, the last employee works from Monday until Wednesday in night shifts (N), on Thursday and Friday has days-off, and on Saturday and Sunday works in the day shift. Each row of this table represents the weekly schedule of one employee.

There are two main variants of workforce schedules: rotating (or cyclic) workforce schedules and non-cyclic workforce schedules. In a rotating workforce schedule all employees have the same basic schedule but start with different offsets. Therefore, while the individual preferences of the employees cannot be taken into account, the aim is to find a schedule that is optimal for all employees. In non-cyclic workforce schedules the individual preferences of the employees can be taken into consideration and the aim is to achieve schedules that fulfill the preferences of most employees. In both variations of workforce schedules other constraints such as the

Table 1: One typical week schedule for 9 employees

Employee/day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	D	D	D	D	D	-	-
2	-	D	D	D	D	D	D
3	D	-	-	N	N	N	N
4	-	-	-	-	A	A	A
5	A	A	A	A	-	-	-
6	N	N	N	N	N	-	-
7	-	-	A	A	A	A	A
8	A	A	-	-	-	N	N
9	N	N	N	-	-	D	D

minimum number of employees required for each shift have to be met. In this paper we will consider the problem of rotating workforce scheduling. This problem is NP-complete problem.

Workforce schedules have a impact on the health and satisfaction of employees as well as on their performance at work. Therefore, computerized workforce scheduling has interested researchers for over 30 years. For solving of problem of workforce scheduling, different approaches were used in the literature. Survey of algorithms used for different workforce scheduling problems is given by Tien and Kamiyama [13]. Examples for the use of exhaustive enumeration for generation of rotating workforce schedules are [5] and [2]. Balakrishnan and Wong [1] solved a problem of rotating workforce scheduling by modeling it as a network flow problem. Several other algorithms for rotating workforce schedules have been proposed in the literature [6, 7, 10]. Laporte [8] considered developing rotating workforce schedules by hand and showed how the constraints can be relaxed to get acceptable schedules. Musliu et al [12] recently proposed and implemented a new framework for computerized rotating workforce scheduling.

In this paper, we investigate the use of tabu search technique for solving the rotating workforce scheduling problem. To our best knowledge the tabu search was never used directly to solve this problem. In the next section we give a precise definition of the problem for which the tabu search is applied. In Section 2 the tabu search approach used for solving this problem is described. In Section 3 the computational results are given and finally in the last section concluding remarks are presented.

1.1 Rotating workforce scheduling

In this section, we describe the problem of assignment of shifts and days-off to employees in case of rotating workforce schedules. This is a specific problem of a general workforce-scheduling problem. The definition is given below ([12]):

Instance:

- Number of employees: n.

- Set A of m shifts (activities) : $a_1, a_2, \ldots, a_m$, where a_m represents the special day-off "shift".
- w: length of schedule. The total length of a planning period is $n \times w$ because of the cyclic characteristics of the schedules.
- A cyclic schedule is represented by an $n \times w$ matrix $S \in A^{nw}$. Each element $s_{i,j}$ of matrix S corresponds to one shift. Element $s_{i,j}$ shows which shift employee i works during day j or whether the employee has time off. In a cyclic schedule, the schedule for one employee consists of a sequence of all rows of the matrix S. The last element of a row is adjacent to the first element of the next row, and the last element of the matrix is adjacent to its first element.
- Temporal requirements: $(m-1) \times w$ matrix R, where each element $r_{i,j}$ of matrix R shows the required number of employees for shift i during day j.
- Constraints:
  - Sequences of shifts permitted to be assigned to employees (the complement of inadmissible sequences): Shift change $m \times m \times m$ matrix $C \in A^{(m^3)}$. If element $c_{i,j,k}$ of matrix C is 1, the sequence of shifts (a_i, a_j, a_k) is permitted, otherwise it is not.
  - Maximum and minimum length of periods of consecutive shifts: Vectors $MAXS_m$, $MINS_m$, where each element shows the maximum respectively minimum permitted length of periods of consecutive shifts.
  - Maximum and minimum length of blocks of workdays: $MAXW$, $MINW$.

Problem: Find a cyclic schedule (assignment of shifts to employees) that satisfies the requirement matrix, and all other constraints.

Note that in [12], finding as many non-isomorphic cyclic schedules as possible that satisfy all constraints, and are optimal in terms of weekends without scheduled work shifts (weekends off), is required. We consider in this paper the generation of only one schedule, which satisfies all constraints. Furthermore, in this paper we do not consider the optimization of weekends off.

2 Tabu search for rotating workforce scheduling

Tabu search technique ([4]) is one of the most powerful modern heuristic techniques, which has been used successfully for many practical problems. The basic idea of this technique is to avoid the cycles (visiting the same solution) during the search by using the tabu list. In the tabu list, specific information of the search history for a fixed specific number of past iterations is stored. Accepting the solution for the next iteration in this technique depends not only from its quality, but also from the tabu list.

Further, we describe the main characteristics of a tabu search technique, which is applied for solving the rotating workforce scheduling problem.

2.1 Generation of neighborhood

Like described in the Section 2, the cyclic schedule is represented by an $n \times w$ matrix S, where each element $s_{i,j}$ of matrix S corresponds to one shift or day off. To generate the neighborhood of the current solution we apply a simple move, which swaps the contents of two elements in the matrix S. However, to reduce the neighborhood of the current solution, we allow only the solutions, which fulfill workforce requirements, and thus the swapping of elements is done only inside of a particular column. The whole neighborhood of the current solution is obtained by swapping all possible elements (not identical) in all columns of a table. The applied move is denoted as $swap(j, i, k)$, where j represents the column in which the swap of elements should be done, and i, k represent elements in column j that should swap their contents. The neighborhood of the current solution represents all solutions that can be obtained by applying this move ($swap(j, i, k)$) to the current solution. The pseudo code for the generation of a neighborhood is presented below.

```
For Column = 1 to NumberOfColumns
    For FirstElement = 1 to NumberOfEmployees − 1
        For SecondElement = FirstElement+1 to NumberOfEmployees
            If S(FirstElement, Column) ≠ S(SecondElement, Column)
                swap(Column, FirstElement, SecondElement)
                Add new solution to the neighborhood
            End if
        Next
    Next
Next
```

2.1.1 Tabu mechanisms and selection criteria

In order to avoid the cycles during the search, two variants of tabu list have been used. In the first variant the applied move is given in the tabu list, and this move is made tabu for several iterations. We tested several lengths of the tabu list. In the second variant, which resulted to be the best for this problem, the whole solution is given in the tabu list. The solutions, which are in the tabu list, are not accepted for a determined number of next iterations. The computational results given in this paper are based on the second variant of the tabu list.

In the first variant of a tabu list, the tabu solution is accepted only if it has the best fitness found so far; otherwise the best non-tabu solution will be selected for the next iteration. In the second variant of the tabu list only the best non-tabu solution is accepted for the next iteration.

2.1.2 Evaluation of solutions

During the generation of neighborhood, the evaluation of solutions is most time consuming, because each solution has to be checked for many constraints. To calculate the fitness of the solution, for each violation of constraint a determined number of points (penalty) is given, based on the constraint and the degree of the constraint violation. The fitness represents the sum of all penalties caused from the violation of constraints. As the problem we want to solve has all hard constraints, the solution will be found when the fitness of the solution reaches the value 0. The fitness is calculated as follows:

$$Fitness = \sum_{i=1}^{NW} P1 \times Distance(WB_i, WorkBRange) + \sum_{i=1}^{ND} P2 \times Distance(DOB_i, DayOffBRange) + \sum_{j=1}^{NumOfShifts} (\sum_{i=1}^{NS_j} P3 \times Distance(SB_{ij}, ShiftSeqRange_j)) + P4 \times NumOfNotAllowedShiftSeq$$

NW, ND, represent respectively, the number of work blocks, and days off blocks, whereas NS_j, represents number of shift sequences blocks of shift j. WB_i, DOB_i, represent respectively, a work block i, and days-off block i. SB_{ij}, represents the i-th shifts block of shift j. The penalties of the violation of constraints are set as follow: $P3 = 1, P1 = P2 = P4 = 2$. The function $Distance(XBlock, range)$ returns 0 if the length of the block $XBlock$ is inside the range of two numbers ($range$), otherwise returns the distance of length of $XBlock$ from the range. For example if the legal range of work blocks is $4 - 7$ and length of work block $XBlock$ is 3 or 8 then this function will return value 1.

3 Computational results

In this section we give computational results for a benchmark example from the literature.

Problem 1 (Laporte et al. [9]): There exist three non overlapping shifts D, A, and N, 9 employees, and requirements are 2 employees in each shift and every day. A week schedule has to be constructed that fulfills these constraints: (1) Rest periods should be at least two days-off, (2) Work periods must be between 2 and 7 days long if work is done in shift D or A and between 4 and 7 if work is done in shift N, (3)Shift changes can occur only after a day-off, (4)Schedules should contain as many weekends as possible, (5)Weekends off should be distributed throughout the schedule as evenly as possible, (6) Long (short periods) should be followed by long (short) rest periods, (7)Work periods of 7 days are preferred in shift N.

Let as note that we cannot model the problem given in this section exactly in our solver. Constraints in our solver are based in a software package called First Class Scheduler (FCS) [3], except that we do not consider distribution of weekends off. This package is already internationally highly appreciated and has been used successfully since the year 2000 in several organizations for generating rotating workforce schedules. We mimic the constraints as closely as possible or to replace them by similar constraints which can be taken in consideration in our solver. In our solver constraint 1 is straightforward. Constraint 2 can be approximated if we take the minimum of work blocks to be 4. Constraint 3 can also be modeled if we take the minimum length of successive shifts to be 4. For maximum length of successive shifts we take 7 for each shift. Other constraints can not be modeled in our solver.

Table 2: Tabu search solution for the problem from [9]

Employee/day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	-	D	D	D	D	D	D
2	D	-	-	D	D	D	D
3	D	D	D	-	-	-	A
4	A	A	A	A	A	A	-
5	-	-	-	N	N	N	N
6	-	-	-	-	N	N	N
7	N	N	N	-	-	A	A
8	A	A	A	A	A	-	-
9	N	N	N	N	-	-	-

The solution for this problem found by the tabu search in 585 iterations, is presented in Table 2.

The proposed tabu search procedure could also generate a solution for the problem solved by Butler [2] and another larger benchmark problem reported in [5]. For extensive computational results see [11].

4 Conclusions

In this paper we proposed a tabu search approach for solving the rotating workforce scheduling problem. We described the neighborhood structure for this problem, tabu mechanisms, and discussed the fitness function. Computational results in the benchmark examples from the literature show that the tabu search approach we proposed in this paper gives promising results for the problem of rotating workforce scheduling. The proposed approach can be extended to solve problems with additional constraints, and to optimize the number of weekends off. Further investigation considering the neighborhood relations and fitness function will be done to improve the current implementation.

Acknowledgments: This work was supported by FWF (Austrian Science Fund) Project No. **Z29-N04**.

References

[1] Nagraj Balakrishnan and Richard T. Wong. A network model for the rotating workforce scheduling problem. *Networks*, 20:25–42, 1990.

[2] B. Butler. Computerized manpower scheduling. Master's thesis, University of Alberta, Canada, 1978.

[3] Johannes Gärtner, Nysret Musliu, and Wolfgang Slany. Rota: A research project on algorithms for workforce scheduling and shift design optimisation. *Artificial Intelligence Communications*, 14(2):83–92, 2001.

[4] Fred Glover and Manuel Laguna. *Tabu search.* Kluwer Academic Publishers, 1997.

[5] N. Heller, J. McEwen, and W. Stenzel. Computerized scheduling of police manpower. *St. Louis Police Department, St. Louis, MO*, 1973.

[6] R. Hung. A multiple-shift workforce scheduling model under 4-day workweek with weekday and weekend labour demands. *J Opl Res Soc*, (45):1088–1092, 1994.

[7] P. Knauth. Designing better shift systems. *Appl Ergonom*, (27):39–44, 1996.

[8] G. Laporte. The art and science of designing rotating schedules. *Journal of the Operational Research Society*, 50:1011–1017, 1999.

[9] G. Laporte, Y. Nobert, and J. Biron. Rotating schedules. *Eur. J. Ops. Res.*, 4:24–30, 1980.

[10] Hoong Chuin Lau. Combinatorial aproaches for hard problems in manpower scheduling. *Journal of Operations Research Society of Japan*, 39(1):88–98, 1996.

[11] Nysret Musliu. Applying tabu search to the rotating workforce scheduling problem. Technical Report DBAI-TR-2003-49, Institut für Informationssysteme der Technischen Universität Wien, 2003.

[12] Nysret Musliu, Johannes Gärtner, and Wolfgang Slany. Efficient generation of rotating workforce schedules. *Discrete Applied Mathematics*, 118(1-2):85–98, 2002.

[13] James M. Tien and Angelica Kamiyama. On manpower scheduling algorithms. *SIAM Review*, 24(3):275–287, 1982.