DBAI TECHNICAL REPORT

INSTITUT FÜR INFORMATIONSSYSTEME

ABTEILUNG DATENBANKEN UND ARTIFICIAL INTELLIGENCE

# Applying Tabu Search to the Rotating Workforce Scheduling Problem

**DBAI-TR-2003-49**

**Nysret Musliu**

Institut für Informationssysteme
Abteilung Datenbanken und
Artificial Intelligence
Technische Universität Wien
Favoritenstr. 9
A-1040 Vienna, Austria
Tel: +43-1-58801-18403
Fax: +43-1-58801-18492
sekret@dbai.tuwien.ac.at
www.dbai.tuwien.ac.at

DBAI TECHNICAL REPORT
2003 (LAST MODIFIED: DECEMBER 2005)

TECHNISCHE UNIVERSITÄT WIEN

DBAI TECHNICAL REPORT

DBAI TECHNICAL REPORT DBAI-TR-2003-49, 2003 (LAST MODIFIED: DECEMBER 2005)

# Applying Tabu Search to the Rotating Workforce Scheduling Problem

**Nysret Musliu**[1]

**Abstract.** Rotating workforce scheduling appears in different forms in a broad range of workplaces, such us industrial plants, call centers, public transportation, and airline companies. It is of a high practical relevance to find workforce schedules that fulfill the ergonomic criteria for the employees, and reduce costs for the organization. In this paper we propose new heuristic methods for automatic generation of rotating workforce schedules. The following methods are proposed: (1)Tabu search based algorithm, (2) Heuristic method based on min-conflicts heuristic, (3) A method that includes in the tabu search algorithm the min-conflicts heuristic and random walk, (4) A method that includes the tabu mechanism and random walk in the min-conflicts heuristic. The appropriate neighborhood structure, tabu mechanism, and fitness function, based on the specifics of problem are proposed. The proposed methods are implemented and experimentally evaluated on the benchmark examples given in the literature and on the real life test problems, which we collected from a broad range of organizations. Empirical results show that the combination of min-conflicts heuristic with tabu search can be used to solve very effectively this problem. The proposed methods improve the performance of the state of art commercial system for generation of rotating workforce schedules and are currently in phase of including in a commercial package for automatic generation of rotating workforce schedules.

[1]Technische Universität Wien mailto: musliu@dbai.tuwien.ac.at

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

## 1 Introduction

Workforce scheduling, in general, includes sub-problems which appear in many spheres of life as in industrial plants, hospitals, public transport, airlines companies etc. 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 employees work in three shifts: day shift(D), afternoon shift(A) and night shift(N). 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 shift (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.

Table 1: A typical week schedule for 9 employees

| Emp./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 |

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 (or fulfills all constraints) for all employees. The schedule presented in Table 1 would be a cyclic schedule, if after the first week the first employee takes the schedule of the second employee, second employee takes the schedule of third, $\ldots$, employee 9 takes the schedule of employee 1. 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 many constraints such as the minimum number of employees required for each shift have to be met. For this reason, they are generally difficult to solve, which corresponds to the extremely large search space and conflicting constraints. In this paper we will consider the problem of rotating workforce scheduling. This problem is an NP-complete problem ([18]).

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. The critical features of workforce scheduling algorithms are their computational behavior and flexibility for solving a wide range of problems that appear in practice. For solving the problem of workforce scheduling, different approaches have been used in the literature. Survey of algorithms used for different workforce scheduling problems is given in [2] and [4]. Examples for the use of exhaustive enumeration are [10] and [5]. Glover and McMillan [9] rely on integration of techniques from management sciences and artificial intelligence to solve general shift scheduling problems. Balakrishnan and Wong [3] solved a problem of rotating workforce scheduling by modeling it as a network flow problem. Laporte [15] considered developing the rotating workforce schedules by hand and showed how the constraints can be relaxed to get acceptable schedules. Several other algorithms for workforce schedules have been proposed in the literature [11, 12, 14, 21]. Recently, Laporte and Pesant [17] have proposed constraint programming algorithm for solving rotating workforce scheduling problem.

In [20] is proposed and implemented a method for the generation of rotating workforce schedules, which is based on pruning of search space, by involving the decision maker during the generation of partial solutions. The algorithms have been included in a commercial product called First Class Scheduler (FCS) (see [7]) which is part of a shift scheduling package called Shift-Plan-Assistant (SPA) of XIMES[1] Corp. The main feature of FCS is the possibility to generate high quality schedules through the interaction with the human decision maker. Apart from the fact that the generated schedules meet all hard constraints, it also allows to incorporate preferences of the human decision maker regarding soft constraints that are more difficult to assess and to model otherwise. Although this package has been shown to work well in practice for solving of a broad range of real life problems, for very large instances solutions cannot always be guaranteed, because of the size of the search space.

In this paper we propose methods with aim of more effective generation of solution for large instances of rotating workforce schedules. Of particular interests was to develop techniques for this problem which can solve problem instances which can not be solved by FCS in a reasonable amount of time. The main contributions of this paper are:

- We propose a tabu search based method for solving of rotating workforce scheduling problem.
- New method based on the min-conflicts heuristic for solving of this problem is proposed.
- We propose combination of the min-conflicts based heuristic and tabu search. A new method which includes so called tabu mechanism in the min- conflicts based heuristic for solving rotating workforce scheduling problem is proposed, and using of the min-conflicts heuristic in the tabu search is considered. Additionally, we consider introducing of random noise and random walk in the adopted random restart min-conflicts heuristic and tabu search method.

[1]http://www.ximes.com/

- We introduces the set of 17 real life problems, which are collected from different areas in industry. The proposed methods has been implemented and empirically evaluated in problems which appeared in literature and the set of introduced real life problems. Additionally, comparison of methods proposed in this paper with the commercial system for generation of rotating workforce schedules is done.

We proceed in this paper as follows: In the next section we give a precise definition of the rotating workforce scheduling problem. In Section 3 the tabu search algorithm is proposed. Section 4 describes new proposed method based on min-conflicts heuristic. In Section 5 the combination of tabu search, min-conflicts heuristic and random walk is proposed. Test problems from literature and the 17 problems collected from real life situations are presented in Section 6. Computational results for real life workforce scheduling problems and the comparison of these results with other approaches in literature is given in Section 7. In the last section conclusions remarks are given.

## 2 The rotating workforce scheduling problem

In this section, we describe the problem of assignment of shifts and days-off to employees in the case of rotating workforce schedules. This is a specific problem of a general workforce-scheduling problem. The definition is given below ([20]):
**Instance:**

- Number of employees: $n$.
- Set $A$ of $m$ shifts (activities) : $a_1, a_2, \dots, a_m$, 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: The requirement matrix R $((m-1) \times w)$, where each element $r_{i,j}$ of the requirement 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 [20], 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. As the fulfilling of all given constraints for this problem in practice is usually sufficient, we consider in this paper the generation of only one schedule, which satisfies all hard constraints given in problem definition.

## 3 Tabu search for rotating workforce scheduling

Tabu search technique [8] is a powerful modern heuristic technique, which has been used successfully for many practical problems. This technique belongs to a class of local search techniques. Local search strategies ([1], [22]) have been successful applied to constraint satisfaction problems and other hard combinatorial optimization problems. The basic idea of local search techniques is to improve initial solutions iteratively during the search. One of basic local search techniques is a hill-climbing technique, which starts with a randomly generated solution and moves during each iteration to the solution in neighborhood with the best evaluation. This technique can easily get stuck in local optimum. Different approaches have been used in literature to escape from a local minimum during the search. Such examples includes simulated annealing ([13]) and tabu search ([8]).

The basic idea of tabu search technique is to avoid the cycles (visiting the same solution) during the search by using the tabu list. In the tabu list specified information for search history for a fixed specific number of past iteration are stored. The acceptance of the solution for the next iterations in this technique depends not only on its quality, but also from the information about the history of the search. In the tabu list one stores for instance the moves (specific changes of solution, see Section 3.1) or inverse moves that have been used during specific number of past iterations. The stored moves are made tabu for several iterations. A solution is classified as a tabu solution if it is generated from a move that is present in the tabu list. In this technique during each iteration the complete neighborhood (with defined moves) of the current solution is generated. In the basic variant of TS the best solution (not tabu) from the neighborhood is accepted for the next generation. However, it is also possible to accept the tabu solution if it fulfills some conditions, which are determined with the so called aspiration criteria (i.e., the solution is tabu but has the best evaluation so far).

Although tabu search was used to solve similar problems with rotating workforce scheduling (e.g. nurse scheduling, see [4]), to our best knowledge tabu search was not used to solve directly the problem we consider in this paper. The rotating workforce scheduling involves typically different constraints compared to nurse scheduling, and what also makes the rotating workforce scheduling different from nurse scheduling is that the schedule is cyclic. During the generation of solution for rotating workforce scheduling, one should also consider fulfilling of constraints which appear because of connection of schedule of each employee with the schedule of next employee.

The appropriate neighborhood structure for tabu search is very important to reach ‘good’ solutions. In this section, we first describe the neighborhood structure for the problem we solve in this paper. We propose such neighborhood structure based on specifics of a problem, so that the search can be more effective. Afterwards, we propose the structure of memory (tabu list) during the search, acceptance of the descendant solutions and aspiration criteria. Furthermore, the appropriate fitness function which guides the search towards the ’good’ solutions for this problem is presented.

## 3.1 Generation of neighborhood

As described in 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 move, which swaps the contents of two elements in the matrix $S$. However, to reduce the neighborhood of the current solution, we only allow those 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. Additionally, the above described move was extended to swap the blocks of more neighborhood cells (last cell in the schedule of employee is a neighbor to the first cell of the schedule for next employee). This move is denoted $swapBlock(j, i, k, length)$, where $j, i, k$ are the same parameters as for the move $swap(j, i, k)$, whereas $length$ represents the length of a block of cells which have to be swapped.

## 3.2 Tabu mechanisms and selection criteria

In order to avoid the cycles during the search, the search history is used. Specific information for the search history is stored in tabu list. In our case in the tabu list are stored the moves which should not be applied for several iterations during the search. For example if the solution accepted for the next iteration is obtained with swapping of contents of cells 4 and 5 in the first column of schedule ($swap(1, 4, 5)$), in the tabu list is stored the moves: $swap(1, 4, 5)$. This move is made tabu for several iterations depending on the length of the tabu list. This should avoid the cycles during the search.

For finding the most appropriate tabu length for tabu search approach we experimented with different length of tabu list. In the first variant we experimented with same length of tabu list for all instances, without taking in consideration the size of problems. In the second variant which showed to be better the length of tabu list is dependent from the size of problem (number of employees in the schedule). We experimented with these lengths of tabu list: 2*NumberOfEmployees, 4*NumberOfEmployees, ...,24*NumberOfEmployees. For each tabu length we experimented with two types of moves we defined earlier. In the first variant only $swap(i, j, k)$ move was applied, whereas in the second variant also the $swapBlock(i, j, k, length)$ move was applied to become the neighborhood solution during each iteration.

## 3.3 Selection Criteria

Considering acceptance of solution for the next iteration the following criteria is applied. The best solution from the neighborhood, if it is not tabu, becomes the current solution in the next iteration. If the best solution from neighborhood is tabu the aspiration criteria is applied. For the aspiration criterion, we use a standard version [8] according to which the tabu status of a move is ignored if the move has a cost better than the current best solution.

## 3.4 Fitness Function

During the generation of neighborhood, the evaluation of solutions is very 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. Since the problem we want to solve only has hard constraints, the solution will be found when the fitness of the solution reaches the value $0$. The fitness is calculated as follows:

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

$NW$, $ND$, 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 follows: $P1 = P2 = P3 = P4 = 1$. 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.

## 3.5 Generation of initial solution

The initial solution is generated randomly considering the positions of shifts inside of one column. However, the initial solution fulfills all workforce requirements. In each column of schedule a determined number of shifts (determined by workforce requirements) are distributed randomly. The neighborhood relation applied in the current solution generates a solution which does not violate the workforce requirements either.

## 3.6 The overall tabu search algorithm

The pseudo code of the tabu search algorithm is presented below:

**Tabu search algorithm**

```
Generate random initial solution
Initialize tabu list

DO WHILE (termination-condition)
    Generate whole neighborhood of current solution as defined in Section 3.1

    Evaluate neighborhood solutions based on fitness function defined in Section 3.4
    Select the solution for the next iteration based on selection criteria defined in Section 3.3

    Update tabu list
LOOP
```

Termination condition is fulfilled if the solution which fulfills all constraints is found, or if the determined number of evaluations of solution is reached. The limit for the number of evaluations for each method proposed in this paper is set to be 10 million (see Section 7).

# 4 Min-conflicts based heuristic for rotating workforce scheduling

The min conflicts heuristic ([19]) has been used successfully for solving constraint satisfaction problems. In this technique during each iteration randomly a conflicted variable is selected. For the selected variable the new value is picked such that the number of conflicts is minimized. Pure min conflicts technique can also get stuck in a local optimum. To escape the local optimum the algorithm can be restarted or noise strategies such as random walk ([23]) can be introduced.

Based on ideas of min-conflicts heuristic in this paper we propose new method for generation of rotating workforce schedules. Due to the specifics of problem we consider here, the proposed method has several differences compare to pure min-conflicts heuristic. Further, we present the adopted random restart min conflicts heuristic. By using random restart the min-conflicts heuristics is started for a determined number of times from different initial solution. The pseudo code of random restart min conflicts heuristics for rotating workforce scheduling is given below:

**Random restart min-conflicts heuristic**

```
DO WHILE ($NumOfRestart < MaxRestart$)
    Generate random initial solution
    DO WHILE (termination-condition)
        Select randomly a cell of schedule among the cells which appear in violated
        constraints

        Generate neighborhood of current solution by swapping of the content of
        selected cell and other cells in the same column (or by swapping of block
        of cells)

        From the generated neighborhood select for the next iteration a solution
        which minimizes number of conflicts (this corresponds to solution which
        mimimizes fitness defined in Section 3.4)
    LOOP
LOOP
```

Considering generation of neighborhood for the current solution in the simplest case the neighborhood is generated by swapping of content of cell which is randomly selected from cells which appear in some violated constraint with the content of other cells appearing in same column. The swap is limited between cells which appear in same column, because with this limitation is assured that solutions fulfill always the workforce requirements. We experimented also with swapping of block of cells. The first cell in block to be swapped is the cell which is selected randomly from the cells which appear in violated constraints. The next elements of block are the neighborhood cells to the randomly selected cell. For the cell $s(i, j)$ of the schedule the neighborhood cell is the cell $s(i, j + 1)$. In case the cell is in the last column ($j = 7$ for the week schedule) the next neighborhood cell is $s(i, 1)$ because of rotation. We experimented with these lengths of blocks: 1, 2, 3, 4. Note that the workforce requirements are also always fulfilled when blocks of cells are swapped.

The main difference of this technique compare to previous proposed tabu search algorithm is in the generation of neighborhood of current solution. In min-conflicts heuristic only part of neighborhood is generated during each iteration. The part of neighborhood to be explored is determined based on the information for the cells of schedule which appears in conflicts (not fulfilled constraints). Only the neighborhood which can be generated by swaps on a column in which one of these cells is located is considered in min-conflicts during each iteration. This makes the search much more effective, especially for larger instances, in which the whole neighborhood of current solution is very large. To have an idea for the number of solutions to be explored by tabu search and min-conflicts during each iteration, suppose that we have a problem with 9 employees and three shifts. Suppose that the requirements for each day are to have 2 employees in each shift. If only the move for swap of single cells is used the number of solutions explored by tabu search during each iteration will be 210, whereas the number of solution explored by min conflicts will be maximally 7 (swaps of one cell in one of columns with all other cells in that column minus 1, because at least one swap will be between cells with same shift). Another difference of min-conflicts heuristic with tabu search is on the acceptance of solution for the next iteration.

The adapted min conflicts based heuristic for rotating workforce scheduling has these differences compared to pure random restart min conflicts heuristic due to the specifics of this problem: In our case not only one variable (cell) in conflict change the value. Here is possible that more than one variable changes their content. In case of simple neighborhood two variables will change their contents, whereas in case of swap of block of length 4, their content will change 8 variables. In our procedure also a worse solution can be accepted for the next iteration, whereas in pure min conflicts not. Considering minimizing of conflicts the basic idea of minimizing of number of conflicts remains, but we evaluate solution not only based on number of conflicts, but also based on the degree of constraint violation (see section 3.4 ). Similar approach for calculating of fitness based on penalty functions is proposed by Galinier and Hao [6].

As for tabu search the termination condition is fulfilled if the solution which fulfills all constraints is found or if the determined number of evaluations of solution is reached. Note that in computational experiments for the min-conflicts heuristic the number of restarts is set to be 10. The maximal number of evaluations per random restart is 1 mil. evaluations.

## 5 Combining of min-conflicts heuristic, tabu search and random walk

In this section we propose combination of techniques proposed in previous section and consider also including of random walk strategy in the proposed methods. We propose combination or tabu search with min conflicts heuristics and random walk, using of random walk in min conflicts heuristics and introducing of tabu mechanism in min-conflicts heuristic. Furthermore, we consider combining of min-conflicts, tabu search and random walk strategy.

By combining of the tabu search with the min-conflicts heuristic and random walk, the basic idea is to not explore the whole neighborhood during each iteration as in classical tabu search, but with some probability to apply either random walk or min conflicts heuristic in each iteration. This makes possible exploring of different regions of search space or diversification of search and also reducing of neighborhood to be explored during each iteration.

Note that random walk has been modified according the specifics of the problem we consider here. The random walk based strategy in our case picks with probability $p$, first randomly one cell which appears in some violated constraint (conflict). The content of this cell is swapped with the content of other cell which is selected randomly in the same column. The reason why we allow the swap of contents of cells only inside of one column as described earlier is to assure that the solutions during each iteration fulfill always the workforce requirements. We have also experimented with the variant in which the two cells are picked randomly, without the restriction that the first cell appears in some conflict (random noise strategy). The combination of tabu search with a random walk is described below:

**Tabu search and random walk**

```
For Each Iteration
    With probability $p$ pick randomly the cell which is in conflict. Pick in the same
    column another cell randomly and swap the contents of two selected cells

    With probability $1 - p$, follow the tabu search algorithm
```

Combination of tabu search with the min conflicts strategy described in Section 4 is given with the pseudo code below:

**Tabu search and min-conflicts heuristic**

```
For Each Iteration
    With probability $p$, apply standard min conflicts strategy

    With probability $1 - p$, follow the tabu search algorithm
```

As we can see from the procedures during each iteration the standard tabu search approach proposed in section 3 will be applied with some probability $1 - p$, whereas with probability $p$ the random walk or min-conflicts strategy is applied. The best values for the parameter $p$ is determined experimentally. Considering probability for random walk we experimented with different probability for random walk: $0.05, 0.1, \ldots, 0.4$. Considering the probability for min-conflicts we experimented with these values for $p$: $0.05, 0.1, \ldots, 0.7$.

Further we considered combination of min conflicts with random walk and random noise. The combination of min conflicts with random walk is described with the pseudo code below:

**Min-conflicts heuristic and random walk**

```
DO WHILE ($NumOfRestart < MaxRestart$)
```

```
    Generate random initial solution
    DO WHILE (termination-condition)
        With probability $p$ pick randomly the cell which is in conflict. Pick in the
        same column another cell randomly and swap the contents of two selected
        cells

        With probability $1 - p$, follow the min conflicts based procedure
    LOOP
LOOP
```

The procedure in which random noise is introduced in the min-conflicts heuristic is the same as when the random walk is introduced, except that in this case with probability $p$ both cells to be swapped are selected randomly, i.e. the column and the rows of cells to be swapped are selected randomly. We experimented with different values for the probability $p$ : $0.02, 0.05, 0.1, 0.15$.

## 5.1 Introducing tabu mechanism in min conflicts based heuristic

We propose an extension of min conflicts based heuristic described in Section 4 by introducing the tabu mechanism ([8]) in the min conflicts heuristic. The basic idea is to store information about the swap of cells during each iteration as for the tabu search technique. The information about the history of swaps is then additionally used in the selection process of solution for the next iteration. We proceed as follows. Each swap used for obtaining the solution for next iteration is stored in the tabu list. For example if the solution accepted for the next iteration is obtained with swapping of contents of cells 4 and 5 in the first column (move $swap(1, 4, 5)$) of schedule, in the tabu list is stored the moves $swap(1, 4, 5)$. The information for swaps is kept in the list only for determined number of next iterations. Note that during the swap of block of cells also the information for the length of blocks is stored in history of swaps. During the selection of solution from the neighborhood for the next iteration the solution obtained from the swaps stored in the tabu list are not taken in considerations for selection. An exception is done if the solution has best evaluation so far (aspiration criteria [8]). The pseudo code of procedure which includes the tabu mechanism in a min conflicts based heuristics is given below:

**<u>Min-conflicts heuristic with tabu list</u>**

```
DO WHILE ($NumOfRestart < MaxRestart$)
    Generate random initial solution

    DO WHILE (termination-condition)
```

```
        Select randomly a cell from the cells which appear in violated constraints

        Generate neighborhood of current solution by swapping of the content of
        selected cell and other cells in the same column (or by swapping of block
        of cells)

        Eliminate from the neighborhood the solutions which are obtained with
        swaps contained in tabu list and which do not have the best fitness so far

        From the remained solutions in neighborhood select the best solution
        which minimizes number of conflicts

        Add in the tabu list the swap with which the selected solution was obtained,
        and remove from tabu list the oldest swap

    LOOP

LOOP
```

The main difference of this algorithm compare to min-conflicts based heuristics is that in this algorithm not always the solution which is the best considering minimizing of number of conflicts is selected for the next iteration. The solution will be accepted for the next iteration only if it is obtained with moves which are not stored in the tabu list during the past determined number of iterations. This should help avoiding cycles or repetitions of same solutions during the search. The only case in which a solution which is obtained with the forbidden swaps is accepted is the case when the solution has the best fitness so far.

When including tabu mechanism it is important to have the appropriate value for the length of tabu list. Very short tabu list may not have enough effects in avoiding of repetition during the search, and to long tabu list may forbid sometimes the solutions which could lead to the better solutions in the search space. In our case we determined this parameter based on the empirical results. We use such tabu list, such that the length of tabu list is dependent from the size of problem. By size of problem we mean the number of employees (groups) for which the schedule should be generated. We experimented with these lengths of tabu list: $0.5 * n, \ldots, 10 * n$, where $n$ represents number of employees.

The described procedure includes tabu mechanism in random restart min-conflicts based heuristic. We also experimented with introducing of both the tabu mechanism and random walk in the random restart min conflicts based heuristics. The pseudo code of this method is presented below:

**<u>MC heuristic with tabu list and random walk</u>**

```
For Each Iteration
```

With probability $p$ pick randomly the cell which is in conflict. Pick in the same column another cell randomly and swap the contents of two selected cells

With probability $1 - p$, follow the min-conflicts heuristic with tabu mechanism

The only difference of this procedure with the previous procedure is that with a determined probability $p$ during each iteration a random walk strategy will be applied, instead of using min-conflicts heuristic with tabu list.

## 6 Test problems

In this paper we give results for 20 problems (Example 1-20). Example 1-3 appeared earlier in the literature and they are described later in this section. In this paper we present for the first time the collection of 17 (corresponds to Examples 4-20) real life problems which were created from shifts and temporal requirements which appeared in a broad range of organizations, like airport, factory, health care organization, etc. The collection of these problems can be downloaded from http://www.dbai.tuwien.ac.at/staff/musliu/benchmarks/. Note that from the given temporal requirements and shifts different size of problems can be created based on the average number of hours per employee per week. For most examples the number of employees and constraints are taken as they are usually proposed by consultants for the given workforce requirements and shifts. For testing of algorithms we have chosen for few instances (instances with more than 48 groups) the average number of hours such that the number of groups is very large. However, also for these instances the temporal requirements and shifts are from real life situations. These 17 problems have these features:

- The number of employees (or groups) for these problems is from 7 to 163. The collection contains small, middle-size and very large instances considering number of employees.
- Number of shifts for 15 examples is 3, whereas for 2 examples is 2. Problems include standard shifts: D (day), A (Afternoon) and N (Night) shift.
- Each example includes constraints about not allowed shift sequences, length of shift sequences, and length of work and days-off blocks.

In Table 2 the features of instances 4-20 are presented in details. Second column represents number of groups (or employees) in each problem. Number of shifts is presented in the third column. In the fourth column are presented not allowed sequences of shifts. Considering not allowed sequences there are two types of sequences. Critical sequences (named seq1): "'N D"' (employee is not allowed to work on day shift after night shift), "'N A"' and "'A D"'. Second type of not allowed sequences named seq2 are: "'N - D"' (employee is not allowed to work on night shift then have free and then work in a day shift), "'N - A"', "'A - D"', "'N - N"' and they are used only if the single day off is allowed. Column named "'Seq. length"' represents length of sequences for each shift (in order of shifts D, A and N). The column named WBL represents the possible length of work blocks and the last column the allowed length of days-off blocks. Workforce requirements for each example are not presented here and the reader is refereed to http://www.dbai.tuwien.ac.at/staff/musliu/benchmarks/ for the specific temporal requirements for each problem.

Table 2: Features of instances 4-16

| Ex | Gr | Shif. | Illegal seq. | Seq length | WBL | DOBL |
|---|---|---|---|---|---|---|
| 4 | 13 | 3 | seq1, seq2 | 2-6, 2-6, 2-4 | 3-7 | 1-4 |
| 5 | 11 | 3 | seq1, seq2 | 2-6, 2-5, 2-4 | 4-7 | 1-4 |
| 6 | 7 | 3 | seq1, seq2 | 2-6, 2-6, 2-6 | 4-7 | 1-4 |
| 7 | 29 | 3 | seq1 | 2-7, 2-6, 2-5 | 4-7 | 2-4 |
| 8 | 16 | 3 | seq1 | 2-7, 2-6, 2-5 | 3-7 | 2-4 |
| 9 | 47 | 3 | seq1 | 2-7, 2-7, 2-6 | 2-7 | 2-4 |
| 10 | 27 | 3 | seq1 | 2-7, 2-6, 2-5 | 4-7 | 2-4 |
| 11 | 30 | 3 | seq1 | 2-6, 2-5, 2-4 | 3-7 | 2-4 |
| 12 | 20 | 2 | seq1 | 2-6, 2-5 | 4-7 | 2-4 |
| 13 | 24 | 3 | seq1 | 2-6, 2-5, 1-4 | 3-7 | 2-4 |
| 14 | 13 | 3 | seq1, seq2 | 2-6, 2-5, 2-4 | 4-7 | 1-4 |
| 15 | 64 | 3 | seq1, seq2 | 2-6, 2-6, 2-5 | 3-6 | 1-4 |
| 16 | 29 | 3 | seq1 | 2-6, 2-5, 2-4 | 4-7 | 2-4 |
| 17 | 33 | 2 | seq1 | 2-6, 2-5 | 3-7 | 2-4 |
| 18 | 53 | 3 | seq1 | 2-7, 2-6, 2-5 | 4-7 | 2-4 |
| 19 | 120 | 3 | seq1 | 2-6, 2-5, 2-4 | 3-7 | 2-4 |
| 20 | 163 | 3 | seq1, seq2 | 2-6, 2-6, 2-5 | 3-6 | 1-4 |

Considering Examples 1-3, they appeared previously in literature and are included in a collection of problems for which we give the computational results here. These three problems are described in details below.

**Example 1**: The first problem from literature is a small problem solved by Butler [5] for the Edmonton police department in Alberta, Canada. Properties of this problem are:
Number of employees: 9
Shifts: 1 (Day), 2 (Evening), 3 (Night)

Temporal requirements:

$$R_{3,7} = \begin{pmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 3 & 3 & 3 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{pmatrix}$$

Constraints: (1)Length of work periods should be between 4 and 7 days; (2) Only shift 1 can precede the rest period preceding a shift 3 work period; (3)Before and after weekends off, only shift 3 or shift 2 work periods are allowed; (4)At least two consecutive days must be assigned to the same shift; (5)No more than two 7-day work periods are allowed and these work periods should not be consecutive.

Let us note here that in all three examples given, we cannot model the problem exactly (the same was true for [3] and [20] approach to the original problems), which is to a high degree due to the different legal requirements found in diffrent states, but we tried to mimic the constraints as closely as possible or to replace them by similar constraints that appeared more meaningful in the European context. In this paper we consider the same constraints considered in [20].

Considering this problem constraints two and three cannot be represented in our framework. The other constraints can be applied in our model and are left like in the original problem. As mentioned, we include additional constraints about maximum length of successive shifts and minimum and maximum length of days-off blocks. In summary, additional constraint used for our solver and in [20] are: Not allowed shift changes: (N D), (N A), (A D); Length of days-off periods should be between 2 and 4; Vector $MAXS_3 = (7, 6, 4)$.

**Example 2** (Laporte et al. [16]): 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.

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.

**Example 3**: This problem is a larger problem first reported in [10]. Characteristics of this problem are:
Number of employees is 17 (length of planning period is 17 weeks).
Three nonoverlapping shifts.
Temporal requirements are:

$$R_{3,7} = \begin{pmatrix} 5 & 4 & 4 & 4 & 4 & 4 & 3 \\ 5 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 3 & 3 & 3 & 4 & 4 & 4 \end{pmatrix}$$

Constraints: Rest-period lengths must be between 2 and 7 days; Work-periods lengths must be between 3 and 8 days; A shift cannot be assigned to more than 4 consecutive weeks in a row; Shift changes are allowed only after a rest period that includes a Sunday or a Monday or both; The only allowable shift changes are 1 to 3, 2 to 1, and 3 to 2.

We cannot model constraints 3, 4, and 5 in their original form. We allow changes in the same work block and for these reason we have other shift change constraints. In our case the following shift changes are not allowed: 2 to 1, 3 to 1, and 3 to 2. Additionally, we limit the rest period length from 2 to 4 and work periods length from 4 to 7. Maximum and minimum length of blocks of successive shifts are given with vectors $MAXS_3 = (7, 6, 5)$ and $MINS_3 = (2, 2, 2)$. Same constraints are used for this problem in [20].

## 7 Computational results

In this section we report on computational results obtained with the current implementation of methods described in this paper. The results for 20 problems described in previous section are given. For comparison are taken into consideration the number of evaluations needed to find the solution and the time for which the solution is found. The experiments were performed in a machine Pentium 4, 1,8GHZ, 512 MB RAM. For each problem 10 independent runs with each algorithm are executed. The maximal number of evaluations for each run is 10 million evaluations.

### 7.1 Results

In the Table 3 a summary of results for the tabu search strategy for 20 problems is given. This table represents the best results obtained with tabu search algorithm, and they are obtained with tabu length $10 * NumberOFEmployees$. Table 4 presents results for tabu search and random walk. These are the best results, which are obtained with probability $p = 0.2$ for random walk. Further, in the Table 5 the results for tabu search and min conflicts are given. The best probability for min-conflicts was shown to be $0.5$. For each problem, the average number of evaluations and the average time needed (in seconds) for generation of legal solution (solution which fulfills all constraints), are presented. Note that the time for which the solutions are found is usually very important, and especially in cases when consultants have to propose to the decision makers in a short time different solutions which need to be discussed. For calculation of averages only the successful runs are considered. The last column named 'solutions' represents number of legal solutions found in 10 runs.

From the tables we can conclude that tabu search approach finds solutions in at least three runs (out of 10 runs) for each instance except for problems 15, 19, and 20 which are the largest problem in collection of problems. As described in Section 3 the tabu search strategy explores whole neighborhood during each iteration, and thus for this strategy is hard to find solutions (in 10 mil. evaluations) for very large instances as neighborhood during each iteration is very large. Tabu search approach is improved in combination with random walk considering both the average number of evaluations and also the number of runs in which solutions are found. However, also

Table 3: Tabu search algorithm (10 runs)

| Example | Number of Evaluat. | Time(sec) | Solutions |
|---|---|---|---|
| 1 | 66554.3 | 1.0 | 10 |
| 2 | 62631.3 | 1.1 | 10 |
| 3 | 454894 | 11.1 | 10 |
| 4 | 55568.3 | 1.3 | 10 |
| 5 | 150836 | 3.2 | 10 |
| 6 | 19953.9 | 0.3 | 10 |
| 7 | 6.12E+06 | 239.2 | 3 |
| 8 | 231230 | 5.4 | 10 |
| 9 | 4.24E+06 | 255.7 | 9 |
| 10 | 1.43E+06 | 52.8 | 10 |
| 11 | 4.82E+06 | 185.5 | 6 |
| 12 | 2.44E+06 | 61.4 | 8 |
| 13 | 435139 | 14.9 | 10 |
| 14 | 354999 | 8.1 | 10 |
| 15 | - | - | - |
| 16 | 1.08E+06 | 40.9 | 10 |
| 17 | 1.22E+06 | 48.5 | 10 |
| 18 | 5.88E+06 | 411.2 | 6 |
| 19 | - | - | - |
| 20 | - | - | - |

including of random walk does not help to solve problems 15,19 and 20. Combination of tabu search with min-conflicts heuristic outperforms both tabu search and tabu search with random walk considering the time and number of evaluations for which the solution are generated and the number of successful runs. The only problem which can not be solved by this strategy is problem 20. The results in general show that the combination of tabu search with random walk and min-conflicts heuristic improves this technique in case of rotating workforce scheduling problem.

In the Table 6 a summary of results for the random restart min-conflicts based strategy is presented. Considering random restart, for each run 10 random restarts are performed. Maximal number of evaluations per random restart is 1 mil. evaluations (for each run total 10 mil. evalua-

Table 4: Tabu search and random walk (10 runs)

| Example | Number of Evaluat. | Time(sec) | Solutions |
|---|---|---|---|
| 1 | 64320.7 | 1.0 | 10 |
| 2 | 55209 | 0.9 | 10 |
| 3 | 296947 | 7.5 | 10 |
| 4 | 50817.7 | 1.2 | 10 |
| 5 | 216117 | 4.6 | 10 |
| 6 | 20553.8 | 0.3 | 10 |
| 7 | 1.74E+06 | 69.0 | 3 |
| 8 | 192726 | 4.5 | 10 |
| 9 | 2.22E+06 | 139.1 | 10 |
| 10 | 566757 | 22.1 | 10 |
| 11 | 2.66E+06 | 103.7 | 10 |
| 12 | 960118 | 24.7 | 10 |
| 13 | 223626 | 7.7 | 10 |
| 14 | 144895 | 3.3 | 10 |
| 15 | - | - | - |
| 16 | 793351 | 30.8 | 10 |
| 17 | 689217 | 28.6 | 10 |
| 18 | 4.49E+06 | 322.7 | 10 |
| 19 | - | - | - |
| 20 | - | - | - |

tions). Further, in the Table 7 the best results for min-conflicts based strategy and random walk are presented (MC-RW). These results are obtained with probability 0.05 for random walk. Results for min-conflicts and random noise (MC-RN) are similar to these results and are not presented here. MC-RN considering number of solutions found in 10 runs, gives same results as MC-RW and considering time for finding of solutions this technique needs in average slightly more time than MC-RW. Best results for min-conflicts based strategy in which tabu mechanism is included are given (MC-T) in the Table 8. These results are obtained with length of tabu list which is equal to number of employees for each example.

From the Tables 6, 7, 8 we can conclude that the ingredients given to random restart min-

Table 5: Tabu search and min-conflicts heuristic (10 runs)

| Example | Number of Evaluat. | Time(sec) | Solutions |
|---|---|---|---|
| 1 | 17846.1 | 0.3 | 10 |
| 2 | 16536.9 | 0.3 | 10 |
| 3 | 134973 | 3.4 | 10 |
| 4 | 21949 | 0.5 | 10 |
| 5 | 77054.9 | 1.6 | 10 |
| 6 | 18945.6 | 0.3 | 10 |
| 7 | 5.08E+06 | 196.1 | 9 |
| 8 | 98563.5 | 2.3 | 10 |
| 9 | 1.21E+06 | 79.5 | 10 |
| 10 | 300060 | 12.2 | 10 |
| 11 | 3.48E+06 | 134.2 | 10 |
| 12 | 721982 | 18.3 | 10 |
| 13 | 188700 | 6.4 | 10 |
| 14 | 98887.1 | 2.3 | 10 |
| 15 | 7.27E+06 | 738.0 | 5 |
| 16 | 262337 | 10.8 | 10 |
| 17 | 256918 | 10.9 | 10 |
| 18 | 2.29E+06 | 169.2 | 10 |
| 19 | 9.4E+06 | 1550.99 | 3 |
| 20 | - | - | - |

conflicts improves this technique for the rotating workforce scheduling problem. Introduction of random walk (and random noise) makes possible to solve all problems in each run. Finally results show that introducing of tabu mechanism in min-conflicts heuristics makes this technique more powerful for the problem we consider here. The results obtained by MC-T are slightly better considering the average time for solving of problems compare to introducing random walk strategy.

We also experimented to additionally introduce random noise and random walk in MC-T. However, combination of min-conflicts heuristic with tabu mechanism and random walk gives not better results than MC-T alone and for this reason these results are not presented here.

In summary based on the experimental results we can conclude that from the 6 methods pre-

Table 6: Random restart min-conflicts based strategy (10 runs)

| Example | Number of Evaluat. | Time(sec) | Solutions |
|---|---|---|---|
| 1 | 306517 | 4.77 | 10 |
| 2 | 104486 | 1.48 | 10 |
| 3 | 2.82E+06 | 69.36 | 10 |
| 4 | 5303.3 | 0.12 | 10 |
| 5 | 820761 | 15.78 | 10 |
| 6 | 207406 | 2.89 | 10 |
| 7 | 1.72E+06 | 62.51 | 10 |
| 8 | 1.52E+06 | 32.52 | 10 |
| 9 | 1.52E+06 | 84.17 | 5 |
| 10 | 320288 | 11.40 | 10 |
| 11 | 7.05E+06 | 254.82 | 1 |
| 12 | 3.12E+06 | 74.26 | 10 |
| 13 | 2.23E+06 | 68.32 | 10 |
| 14 | 417234 | 8.77 | 10 |
| 15 | 3.65E+06 | 331.11 | 7 |
| 16 | 415611 | 14.48 | 10 |
| 17 | 1.51E+06 | 54.79 | 10 |
| 18 | 967663 | 60.58 | 10 |
| 19 | 4.28E+06 | 577.96 | 7 |
| 20 | 779768 | 183.82 | 10 |

sented in this paper the best results are obtained by min-conflicts heuristic which includes tabu mechanism or random walk during the search.

In Tables 9 and 10 are presented solutions of min-conflicts heuristic with tabu mechanism for problem 2 and problem 4 (one of solution from 10 runs). We do not present all solutions here and the reader is refereed to http://www.dbai.tuwien.ac.at/staff/musliu/benchmarks/ for the solutions obtained with MC-T strategy for all problems presented in this paper.

Table 7: Min-conflicts based heuristic with random walk (10 runs)

| Example | Number of Evaluat. | Time(sec) | Solutions |
|---|---|---|---|
| 1 | 4885.6 | 0.07 | 10 |
| 2 | 7374.3 | 0.11 | 10 |
| 3 | 28205.4 | 0.68 | 10 |
| 4 | 5920.5 | 0.13 | 10 |
| 5 | 23291.4 | 0.48 | 10 |
| 6 | 4308.1 | 0.06 | 10 |
| 7 | 1.40E+06 | 51.68 | 10 |
| 8 | 28332.2 | 0.63 | 10 |
| 9 | 205379 | 11.48 | 10 |
| 10 | 25878.4 | 0.94 | 10 |
| 11 | 274413 | 10.02 | 10 |
| 12 | 89181.1 | 2.17 | 10 |
| 13 | 56653.4 | 1.74 | 10 |
| 14 | 35890.5 | 0.79 | 10 |
| 15 | 3.60E+06 | 328.52 | 10 |
| 16 | 24319.9 | 0.90 | 10 |
| 17 | 34727.7 | 1.31 | 10 |
| 18 | 108731 | 7.22 | 10 |
| 19 | 312237 | 44.18 | 10 |
| 20 | 317008 | 78.36 | 10 |

## 7.2 Comparison with other methods in literature and with a commercial workforce scheduling system

In this section the results of min-conflicts heuristic with tabu mechanism proposed in this paper are compared with other approaches proposed in literature for the three existing problems in literature. The results are compared with [3],[20], and [17]. In [3] the rotating workforce scheduling problem is solved by modeling it as a network flow problem. Their algorithm was implemented in Fortran and was tested on an IBM 3081 computer. In [17] this problem is solved by using a constraint programming algorithm which was coded in ILOG Solver 4.4. The algorithms are tested in a Sun Sparc Ultra 5 workstation (400MHz, 128 Mb RAM). Algorithms proposed in [20] are part

Table 8: Min-conflicts based heuristic with tabu mechanism (10 runs)

| Example | Number of Evaluat. | Time(sec) | Solutions |
|---|---|---|---|
| 1 | 5071.1 | 0.07 | 10 |
| 2 | 4890.5 | 0.07 | 10 |
| 3 | 16798.5 | 0.42 | 10 |
| 4 | 5183.5 | 0.11 | 10 |
| 5 | 20638.9 | 0.43 | 10 |
| 6 | 5941.1 | 0.08 | 10 |
| 7 | 1.42E+06 | 52.79 | 10 |
| 8 | 33455 | 0.74 | 10 |
| 9 | 283803 | 15.96 | 10 |
| 10 | 15815.7 | 0.60 | 10 |
| 11 | 354996 | 13.15 | 10 |
| 12 | 47553.7 | 1.17 | 10 |
| 13 | 27535 | 0.87 | 10 |
| 14 | 34284.6 | 0.76 | 10 |
| 15 | 1.72E+06 | 159.04 | 10 |
| 16 | 13728.4 | 0.54 | 10 |
| 17 | 57943.7 | 2.16 | 10 |
| 18 | 101432 | 6.83 | 10 |
| 19 | 543171 | 75.83 | 10 |
| 20 | 284938 | 71.38 | 10 |

of the commercial software First Class Scheduler (FCS). The algorithms proposed in [20] and the algorithms proposed in this paper are tested on a Pentum 4, 1,8GHZ, 512 MB RAM. In the Table 11 the results for three problems are shown. For each of approaches proposed in literature the time needed to reach first solution is shown (for MC-T this is the average time over 10 runs). Although the experiments were performed in different computers and the results are not totally comparable because of some differences in constraints, we can conclude from the table that the MC-T approach proposed in this paper gives very good results for these problems and comparable results with these given in literature.

For the next 17 Problems proposed in this paper, we can not make comparison with other meth-

Table 9: One of solutions for problem 2 with MC-T method

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

Table 10: One of solutions for problem 4 with MC-T method

| Emp./day | Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|---|
| 1 | A | A | A | A | A | A | - |
| 2 | A | A | A | - | - | - | - |
| 3 | D | D | D | A | A | A | - |
| 4 | N | N | N | - | - | - | - |
| 5 | A | A | A | N | N | - | - |
| 6 | A | A | A | A | A | A | - |
| 7 | A | A | A | A | A | A | - |
| 8 | - | - | - | D | D | D | - |
| 9 | D | D | D | D | D | D | - |
| 10 | D | D | D | A | A | A | - |
| 11 | - | - | - | D | D | D | - |
| 12 | D | D | D | D | D | D | - |
| 13 | D | D | D | D | D | D | - |

Table 11: Comparison between MC-T and other aproaches in literature

| Example | Groups | [3] | [20] | [17] | MC-T |
|---|---|---|---|---|---|
| 1 (from [5]) | 9 | 73.54 | 0.9 | 3.78 | 0.07 |
| 2 (from [16]) | 9 | 310.84 | 0.4 | 0.03 | 0.07 |
| 3 (from [10]) | 17 | 457.98 | 1.9 | 10.26 | 0.42 |

ods (except the method proposed in [20]), as we do not have access to these methods. However, for these examples we compare the best method proposed in this paper with the method included in the last version of commercial software for generation of rotating workforce schedules (First Class Scheduler (FCS) ([7])). This system has been used very successfully since year 2000 in practice for many companies in Europe. To our best knowledge FCS is the state of art commercial system for generation of rotating workforce schedules.

FCS is based on interaction with decision maker. The process of generation of schedules is divided in four steps ([20]) : (1) choosing a set of lengths of work blocks (a work block is a sequence of consecutive days of work), (2) choosing a particular sequence of blocks of work and days-off blocks amongst these that have optimal weekend characteristics, (3) enumerating possible shift sequences for the chosen work blocks subject to shift change constraints and bounds on sequences of shifts, and (4) assignment of shift sequences to work blocks while fulfilling the staffing requirements. We can not do direct comparison between the methods proposed here and FCS, as FCS is based on decision maker selection of intermediate solution in each step. To make possible this comparison we automate the process of generation of solution in FCS as follows: First the possible solutions considering work blocks are generated. For each possible solution in step 1 is proceeded with generation of sequence of work and days off and these sequences are tested for possible assignment of shifts. Search is interrupted when the first solution is found. In each of steps in FCS is possible to have huge number of intermediate solutions especially for large instances and the process of generation of intermediate solutions can take to long. In FCS decision maker can stop at anytime the process of generation of intermediate solutions. We mimic the decision maker here such that we limit the time for test for each solution produced in step 1. For each example we experimented first with limitation of 1 second, then if no solutions is found with limitation of 10 seconds, and if again the solution are not found the limit for test is increased in 100 seconds, and then in 1000 seconds. We begin with 1 second, because it is possible that there exist class solutions in step 1 for which there is not possible to find legal assignment of shifts. Such limitation makes possible to find faster solutions and not lose time in intermediate solutions which can not produce legal solutions. We proceed with other phases (10, 100, 1000 sec), because if the time limitation is to small it is possible to not find solution from the class solutions in step 1, even if for the class solution may exist legal assignments of shifts.

The comparison of methods proposed in this paper and the last version of FCS is given in Table 12. The second column in table named groups represents number of groups (or employees if the group has only one employee) for each example. Third column represents time in seconds needed to generate first solution in FCS, and the last column represents average time from 10 runs to find solutions using tabu search in combination with min-conflicts based heuristic (MC-T). The experiments for MC-T and FCS were performed in the same machine (Pentum 4, 1,8GHZ, 512 MB RAM).

Table 12: Comparison between First Class Scheduler and Min-conflicts heuristic with tabu list

| Ex. | Groups | FCS (time in sec) | MC with tabu list (sec) |
|---|---|---|---|
| 1 | 9 | 0.9 | 0.07 |
| 2 | 9 | 0,4 | 0.07 |
| 3 | 17 | 1.9 | 0.42 |
| 4 | 13 | 1.7 | 0.11 |
| 5 | 11 | 3.5 | 0.43 |
| 6 | 7 | 2 | 0.08 |
| 7 | 29 | 16.1 | 52.79 |
| 8 | 16 | 124 | 0.74 |
| 9 | 47 | >1000 (?) | 15.96 |
| 10 | 27 | 9.5 | 0.60 |
| 11 | 30 | 367 | 13.15 |
| 12 | 20 | >1000 (?) | 1.17 |
| 13 | 24 | >1000 (?) | 0.87 |
| 14 | 13 | 0.54 | 0.76 |
| 15 | 64 | >1000 (?) | 159.04 |
| 16 | 29 | 2.44 | 0.54 |
| 17 | 33 | >1000 (?) | 2.16 |
| 18 | 53 | 2.57 | 6.83 |
| 19 | 120 | >1000 (?) | 75.83 |
| 20 | 163 | >1000 (?) | 71.38 |

From Table 12 we can conclude that the methods proposed in this paper outperform FCS almost in all instances. Considering larger instances FCS solves faster instance 7 and 18. These two problem instances have same workforce requirements for all days and shifts. According to previous experiences with FCS, this system can usually solve instances with such nature, even if they are large instances. Based on empirical results for the methods proposed here such instances seems not to be easier to solve compare to the other problems of similar size. For instances in which in column FCS '>1000 (?)' is written, FCS could not find solution in 1000 seconds and it is not clear if FCS can find solutions for these problems. For small problems FCS has its advantages, because of generation of more solutions and possibility to include soft constraints in interaction with decision maker. However, methods proposed in this paper improve performance of system for solving of middle and larger instances of problems which can not be solved by FCS in a reasonable amount of time.

## 7.3 Conclusions

In this paper we proposed novel methods for solving rotating workforce scheduling problem. We proposed the tabu search based algorithm and a method which is based in ideas of min-conflicts based heuristic. Introducing of min- conflicts strategy and random walk in tabu search was further considered. The method based in min-conflicts strategy was extended by introducing of tabu mechanism in this strategy. Additionally, combination of all three methods: min -conflicts heuristic, random walk and tabu search was considered.

The proposed methods have been implemented and experimentally evaluated in examples from literature and other real life examples, which appeared in a broad range of organizations. Experimental results show that combination of tabu search and min-conflicts strategies proposed in this paper improves significantly the performances of tabu search and min-conflicts based heuristic alone. In particularly introduction of random walk and min-conflicts in tabu search improves the performance of tabu search alone. Min-conflicts strategy is improved by introducing the tabu mechanism and random walk. Overall, experimental results show that the min-conflicts based strategies are good choice for this problem to reach very good results in a short time. The best results are reached by introducing of tabu mechanism or random walk in min-conflicts based heuristic. Both techniques can find for each problem solutions in each run in a short amount of time.

Min-conflicts heuristic with tabu mechanism has been compared with three other approaches from literature for existing 3 benchmark problems proposed earlier in the literature. The results show that our approach gives very comparable results for these problems. Furthermore, comparison for 20 problems with a commercial workforce scheduling system First Class Scheduler show that proposed approach improves significantly the performance of this system, especially for large instances. The proposed methods in this paper are in phase of including on a last version of a system for automatic generation of rotating workforce schedules.

For the future work we are considering the extension of the methods proposed in this paper to solve the nurse scheduling problem. In this problem other constraints may appear and individual preferences of employees can be taken into consideration.

## References

[1] Emile Aarts and Jan Karl Lenstra, editors. *Local Search in Combinatorial Optimization*. Wiley, 1997.

[2] H. K. Alfares. Survey, categorization, and comparison of recent tour scheduling literature. *Annals of Operations Research*, 127(1-4):145–175(31), 2004.

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

[4] Edmund K. Burke, Patrick De Causmaecker, Greet Vanden Berghe, and Hendrik Van Landeghem. The state of the art of nurse rostering. *J. of Scheduling*, 7(6):441–499, 2004.

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

[6] P. Galinier and J.K. Hao. A general approach for constraint solving by local search. *Journal of Mathematical Modelling and Algorithms*, 3(1):73–88, 2004.

[7] 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.

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

[9] Fred Glover and Claude McMillan. The general employee scheduling problem: An integration of MS and AI. *Comput. Ops. Res.*, 13(5):563–573, 1986.

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

[11] R. Hung. A three-day workweek multiple-shift scheduling model. *J Opl Res Soc*, (44):141–146, 1993.

[12] 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.

[13] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

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

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

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

[17] G. Laporte and G. Pesant. A general multi-shift scheduling system. *Journal of the Operational Research Society*, 55/11:1208–1217, 2004.

[18] Hoong Chuin Lau. On the complexity of manpower shift scheduling. *Computers. Ops. Res.*, 23(1):93–102, 1996.

[19] Steven Minton, Mark D. Johnston, Andrew B. Philips, and Philip Laird. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58:161–205, 1992.

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

[21] K. Nonobe and T. Ibaraki. A tabu search approach to the constraint satisfaction problem as a general problem solver. *European Journal of Operational Research*, 106:599–623, 1998.

[22] Colin R. Reeves, editor. *Modern Heuristic Techniques for Combinatorial Problems*. Halsted Pr., 1993.

[23] Richard J Wallace and Eugene C Freuder. Heuristic methods for over-constrained constraint satisfaction problems. In *Overconstrained Systems, Springer 1106*, 1996.