

# A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem

Edmund Burke<sup>1</sup>, Patrick De Causmaecker and Greet Vanden Berghe<sup>2</sup>

<sup>1</sup> Department of Computer Science, University of Nottingham, University Park, Nottingham, NG7 2RD, UK, Tel: (0044)(115)9514234, Fax: (0044)(115)9514254, e-mail: ekb@cs.nott.ac.uk

<sup>2</sup> KaHo St.-Lieven, Procestechnieken en Bedrijfsbeleid, Gebr. Desmetstraat 1, 9000 Gent, Belgium, Tel: (0032)(9)2658610, Fax: (0032)(9)2256269, e-mail: patdc,greetvb@kahosl.be

**Abstract.** This paper deals with the problem of nurse rostering in Belgian hospitals. This is a highly constrained real world problem that was (until the results of this research were applied) tackled manually. The problem basically concerns the assignment of duties to a set of people with different qualifications, work regulations and preferences.

Constraint programming and linear programming techniques can produce feasible solutions for this problem. However, the reality in Belgian hospitals forced us to use heuristics to deal with the over constrained schedules. An important reason for this decision is the calculation time, which the users prefer to reduce. The algorithms presented in this paper are a commercial nurse rostering product developed for the Belgian hospital market, entitled Plane.

**Keywords** Nurse rostering, personnel scheduling, tabu search

## 1 Introduction

In this paper we will discuss the algorithms that have been developed for the commercial nurse rostering system (Plane). The development of Plane was based on an extensive market research in 1993. One of the conclusions was that the requirements of Belgian hospitals cannot be met with a cyclic 'three shift' schedule. Recent research done by the Stichting Technologie Vlaanderen [7] also showed that, instead of a cyclic schedule, the nurses prefer an 'ad-hoc schedule' in which they can express their personal wishes and priorities. Because of the size of the solution space (the scheduling period is usually one month and the number of possible duties per day varies from 6 to 15), the nurse rostering problem tackled by Plane differs a lot from other rostering problems described in the literature. The planning period in [2, 3] is restricted to 1 week and in [2, 5, 6] there are only three different duties to be planned.

Plane can decide (per nurse) which duties can or cannot be performed (according to that nurse's qualification category) when there is not enough personnel available.

Another goal of Plane is the freedom for the user to define a personal cost function modifying predefined constraints, modifying weight parameters, . . . The solution method has to be robust enough to cope with widely varying cost functions.

In [3] a constraint programming solution for the nurse rostering problem is presented. Preliminary experiments with Oz showed that it is very hard to calculate monthly schedules that take into account the high number of 'consecutiveness' constraints that Belgian hospitals have to deal with. Also in the mathematical approaches of [4, 8], the number of different constraints is much lower than in our problem.

A heuristic method, combining tabu search and algorithms based on manual scheduling techniques proved to be very appropriate for this combinatorial problem in which the calculation speed is as important as the attempt to find a solution that is close to optimal.

## 2 Plane, nurse rostering software for Belgian hospitals

Plane is a scheduling system developed by Impakt<sup>3</sup> and GET<sup>4</sup> to assist the scheduling of personnel in hospitals for which the demands for every qualification can be determined over a fixed period in time and which have to fulfil a number of constraints, limiting their assignments.

A description of Plane, its problem domain, its system specific and functional requirements can be found in [1]. The first version of Plane was first implemented in a hospital in 1995 but the system is still evolving to cope with the new and more complicated real world problems that keep appearing. So far, several hospitals in Belgium have replaced the very time consuming manual scheduling by this system.

The cost function used in the algorithms is modular and can deal with all constraints matching the types described section 3.2.

## 3 Problem description

In general, a ward consists of about 20 people, having different qualifications and responsibilities. These people are placed into categories based upon their qualifications and job description such as head nurse, regular nurse, nurse aid, student, . . . Some of the nurses can replace people from another category (depending upon their qualifications). Each replacement by a person from another category will raise the evaluation function by an amount the user can set.

---

<sup>3</sup> Impakt N.V., Ham 64, B-9000 Gent

<sup>4</sup> GET, General Engineering & Technologie, Antwerpse Steenweg 107, B-2390 Oostmalle

### 3.1 Hard constraints

The personnel requirements are expressed in terms of a required number of nurses of every category for every duty during the planning period, which is often one month. These requirements are the only hard constraints in the problem. Optionally, the user can choose to plan the minimum number of required personnel or the preferable number of personnel. A third option is to plan at least the minimal required number of nurses and to add nurses whenever it doesn't increase the evaluation function ('add duties towards preferred personnel requirements' Fig. 1).

### 3.2 Soft constraints

It is highly exceptional in real world problems to find a schedule that satisfies all the soft constraints, but the aim of the algorithms is to minimise the violations of these constraints. The constraints are all to be specified by the users of the system. Certain general constraints are recommended by hospital regulations (but in certain situations, may need to be ignored). There are other soft constraints that are normally created by an agreement between the head nurse (or personnel manager) and the individual nurses. At this moment there are about 30 (modifiable) constraints. It is usually the case that not all constraints can be satisfied at the same time. When a contradiction between constraints occurs, the personal preferences of staff (such as requests for holidays, requests to work a certain duty on a certain day) are stronger than any other constraint. A detailed list of the constraints in Plane can be found at <http://www.impakt.be/plane/indexf.htm>.

## 4 Tabu search algorithm and variants

The entire flow diagram of the hybrid tabu search algorithms described in this section can be seen in Fig. 1.

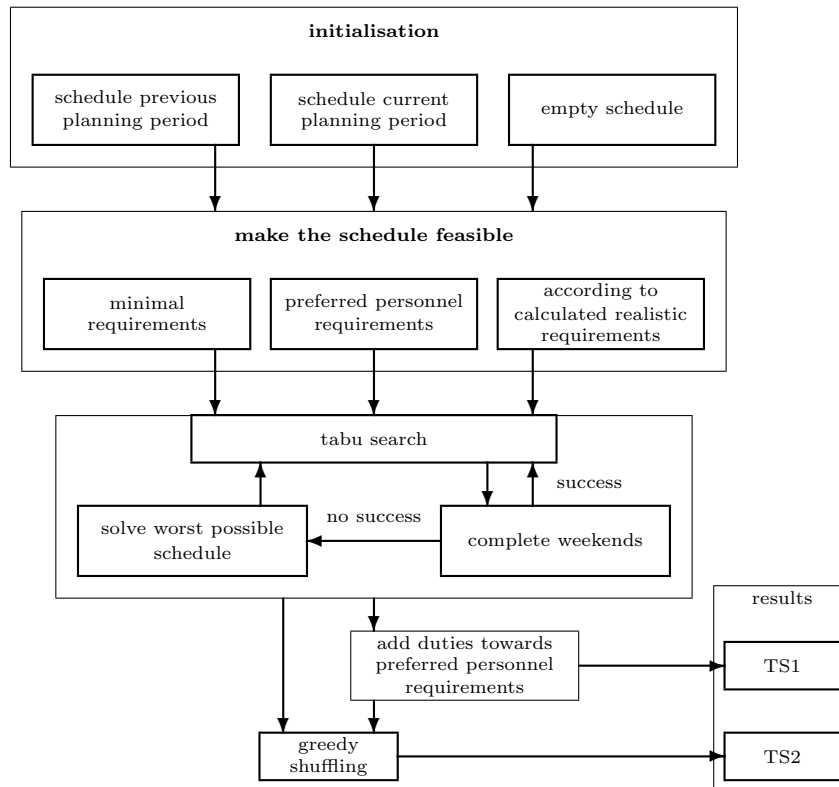
### 4.1 Feasible initial schedule

The first part of the scheduling algorithm is the construction of a feasible initial solution. For practical planning problems three possible strategies are used:

*Current schedule:* This is especially useful when urgent changes in the schedule are required. In real life this may happen when a scheduled nurse is suddenly ill and has to be replaced and, of course, we do not want to drastically change the schedule for the other people.

*Schedule of the previous planning period:* this option is useful when the schedule in the previous planning period was of very high quality and the constraints on the current and the previous planning period are similar.

*Random initialisation:* This is the simplest initialisation, it starts from an empty schedule.



**Fig. 1.** Diagram of the hybrid tabu search algorithms for the nurse rostering problem

After this initialisation, the schedule has to be made feasible. This is carried out by randomly adding and/or removing duties for every category until the requirements are met.

Although the two first initial schedule constructors may seem very attractive, our experiments show that it is not too difficult for the tabu search algorithm to produce schedules of comparable quality starting from a random initial schedule. Indeed it is often the case that with the two latter initialisations, the algorithm is in a local minimum already and has problems escaping from it.

#### 4.2 Original tabu search algorithm

In the simplest tabu search algorithm, the only move we consider is a move of a duty from one person to another on the same day. The move is not allowed if the goal person is not of the right category or is already assigned to that duty. This will not affect the hard constraints.

For each category (for each iteration) possible moves will be calculated and the

move leading to the highest benefit will be performed. If the highest benefit is negative, the move will be performed anyway, unless this move is forbidden by the tabu list. When a move is accepted, an area in the roster around the roster point where the duty comes from and where it is moved to may not be changed. For comparison purposes only, we introduced a steepest descent algorithm in which the neighbourhood of the moves is exactly the same as in the tabu search algorithm. After evaluating all the possible moves in the neighbourhood, the best one will be performed, unless this best move does not improve the schedule, in which case the algorithm stops. These algorithms turned out not to be powerful enough to produce good solutions for complex problems as is shown in the 'steepest descent' and 'tabu search' experiments in Table 1 & 2 (section 5). The tabu search algorithm performs better than the steepest descent algorithm and is therefore used as a local search heuristic in the hybrid algorithms described in section 4.4.

### 4.3 Some Heuristics for the problem

Here we describe some heuristics that can be employed (in conjunction with the tabu search algorithm) to improve the solution.

**Diversification 1: Complete weekend** Although the users of the program can assign a cost parameter to this constraint it is very hard to find satisfactory solutions. The problem is that there are so many constraints and the degree of freedom of the problem is so high that it is likely to find solutions satisfying many other constraints but not this one. In the graphical user interface, incomplete weekends really catch the eye, while other constraints such as overtime or too many morning shifts on Mondays, . . . are not immediately visible. Because it is almost impossible to guarantee good solutions with a certain setting of the parameters, we decided to solve this problem the hard way, by not caring about possible problems for other constraints.

**Diversification 2: Consider the worst personal schedule** If the complete weekend function (above) has not changed the schedule it can be beneficial to look at the people with the worst schedule (according to the evaluation function). For every person (within the category being scheduled) it is possible to calculate the value of the evaluation function after exchanging a part of the schedule of the people involved. The parts of the schedule always contain full days and the maximum length is half the planning period. After all possibilities have been calculated, which is quite time consuming, the best exchange (chosen at random from equal values) is performed. The result of this process often results in a better solution.

**Greedy shuffling: Model human scheduling techniques** There was a problem with the results of the tabu search algorithm because sometimes a human

could improve the visual result by making a small change. This process calculates all possible 'Diversification 2' (above) moves for every pair of people. After listing the gain in the cost function for every possible exchange, the shuffle leading to the best improvement will be performed. Afterwards, the next best improvement in the list is performed, provided none of the people involved were already involved in an earlier shuffle. As long there are improving exchanges in the list, they are carried out. The whole procedure starts over again until none of the possible exchanges improves the cost function.

The improvements on the schedule that can be obtained by employing this procedure and tabu search (described below) are considerable but the biggest advantage of this step is that it creates schedules for which it is almost impossible for a human to improve the schedule.

#### 4.4 Hybrid tabu search algorithms

After extensive testing of hybrid versions of the tabu search algorithm and the above heuristics 2 algorithms were developed. The first one produces schedules when a very short calculation time is required (as it often is). The second algorithm needs more calculation time but generates schedules of a considerably higher quality.

**Tabu search + diversification: TS1** The aim of this algorithm is to provide reliable solutions in a very short time. In practice this algorithm has proved to be very useful to check whether the constraints are realistic, whether during the holiday periods it will be possible to plan good schedules if every person gets their desired holiday period etc. . .

The algorithm is constructed quite simply from the original tabu search algorithm. If after a number of iterations no improvement is found, the weekend step is performed. If the weekend step does not result in a different schedule the second diversification step is performed. After this diversification step, the original tabu search algorithm is used again and so on. The calculations stop after a number of iterations without improvement.

**Tabu search + greedy shuffling: TS2** This requires more time but the results are considerably better from the human point of view. Anecdotal evidence suggests that the level of satisfaction with schedules produced by this algorithm is actually higher than the cost function indicates. The main reason for this is that after the shuffling step the users cannot easily improve the results.

It is important to do the greedy shuffling step at the end of the calculations because its real aim is to perform the exchanges a human user would perform. It is because of the exhaustive search character of the shuffling that this step takes a lot of time. It is very important to calculate this step until there are no further improvements because otherwise the goal of excluding manual improvements to the schedule might be lost (Greedy shuffling in section 4.3).

## 5 Test results

The tests in this paper are restricted to planning the minimal requirements (R-min), planning between the minimal and the preferred requirements (R-min-pref), and planning according to the calculated demands (R-calc) as explained in section 3.1 (Hard constraints). For the latter we decided to do the step 'add duties towards preferred personnel requirements' whenever this does not cause a violation of the soft constraints (section 3.2). In Table 1 and 2, the results of

Problem 1	R-min		R-min-pref		R-calc	
	Value	Time	Value	Time	Value	Time
<b>steepest descent</b>	2594	1'26"	2395	1'37"	2657	1'36"
<b>tabu search</b>	2435	2'05"	2214	2'06"	1928	1'59"
<b>ts stop crit. x50</b>	1915	40'58"	1675	41'21"	1534	23'58"
<b>ts1</b>	1341	6'00"	1089	5'59"	929	5'27"
<b>ts2</b>	1264	20'15"	1011	24'39"	809	28'08"

**Table 1.** Value of the evaluation function and results of the steepest descent and variants of the hybrid tabu search algorithm for Problem 1, planning order of the qualifications as chosen by the customer

Problem 2	R-min		R-min-pref		R-calc	
	Value	Time	Value	Time	Value	Time
<b>steepest descent</b>	1338	44"	1338	45"	1134	47"
<b>tabu search</b>	1189	57"	1189	58"	933	1'03"
<b>ts1</b>	843	3'18"	843	3'18"	867	2'14"
<b>ts2</b>	809	6'25"	809	6'25"	588	10'19"

**Table 2.** Value of the evaluation function and results of the steepest descent and variants of the hybrid tabu search algorithm for Problem 2, planning order of the qualifications as chosen by the customer

the variants of the tabu search algorithm are compared to the steepest descent algorithm. The test examples Problem 1 and Problem 2 are hard to solve real world problems and in both cases the personal demands make a good schedule almost impossible.

The column 'value' shows the value of the evaluation function (cost parameter per constraint times the extent the constraint is violated). The column 'Time' contains the calculation times on an IBM Power PC RS6000.

The third set of results, where the demands are adapted to the constraints as described in section 3.1 (calculating more realistic demands), are better than the results in the first column. In Problem 2, there was no difference between the minimal and the required demands.

For all the considered examples, the tabu search algorithm performs better than the steepest descent algorithm. We decided to organise the stop criterion for the tabu search algorithm so that the calculation time is of the same order of magnitude as the time required to do steepest descent. Only Table 1 contains

the results of the original tabu search algorithm for a longer calculation time. The behaviour of the hybrid algorithms is better than the behaviour of the normal tabu search algorithm (with a short calculation time). Even considering the calculation time, for use in practice it is worth using the hybrid tabu search algorithm because the degree of confidence the users have in the program is much higher.

## 6 Conclusion

By automating the nurse rostering problem, the scheduling effort and calculation time are reduced considerably. The evaluation of schedules is very quick for all possible combinations of constraints and the quality of the automatically produced schedules is much higher than the quality of the manual schedules. The users of Plane often place an emphasis on the higher quality of the solution because the system provides an objective schedule in which all nurses are treated equally and in which the number of violated constraints is very low. Combining the simple tabu search algorithm with some specific problem solving heuristics not only guarantees better quality rosters but also satisfies the users of the system to a very high extent because it is almost impossible for experienced planners to improve the results (considering the constraints) manually. For many practical scheduling problems the higher quality of the solutions produced by the hybrid algorithm compared to the simple tabu search algorithm compensates for the increase in calculation time.

## References

1. Coppens, P.: Geautomatiseerde personeelsplanning bij het A.Z. Sint-Erasmus. GET info **9** (1995) 1–3
2. Dowsland, K.: Nurse scheduling with Tabu Search and Strategic Oscillation. EJOR (to appear)
3. Meisels, A., Gudes, E., Solotorevski, G.: Employee Timetabling, Constraint Networks and Knowledge-Based Rules: A Mixed Approach. Practice and Theory of Automated Timetabling, First International Conference Edinburgh (1995) 93–105
4. Miller, H., Pierskalla, W., Rath, G.: Nurse Scheduling Using Mathematical Programming. Operations Research **24** (1976) 857–870
5. Okada, M.: Prolog-Based System for Nursing Staff Scheduling Implemented on a Personal Computer. Computers and Biomedical Research **21** (1988) 53–63
6. Okada, M.: An approach to the Generalised Nurse Scheduling Problem - Generation of a Declarative Program to represent Institution-Specific Knowledge. Computers and Biomedical Research **25** (1992) 417–434
7. Vanderhaeghe, S.: Dienstroosters in de gezondheidszorg: ongezond? Informatiedossier. Stichting Technologie Vlaanderen (1998)
8. Warner, M., Prawda, J.: A Mathematical Programming Model for Scheduling Nursing Personnel in a Hospital. Management Science **19** (1972) 411–422