To appear in Journal of Heuristics, 2009, doi: 10.1007/s10732-008-9099-6

A Shift Sequence Based Approach for Nurse Scheduling and a New Benchmark Dataset

Peter Brucker[a], Edmund Burke[b], Tim Curtois[b], Rong Qu[b*],
and Greet Vanden Berghe[c]

[a] Fachbereich Mathematik/Informatik, University of Osnabruck
Albrechtstr. 28, 49069 Osnabruck, Germany
[b] Automated Scheduling, Optimisation and Planning (ASAP) Group
School of CSiT, University of Nottingham, Nottingham, NG8 1BB, UK
[c] Information Technology, Engineering Department
KaHo St.-Lieven, 9000 Gent, Belgium

Abstract. This paper investigates an adaptive constructive method for solving nurse rostering problems. The constraints considered in the problems are categorised into three classes: those that are *sequence* related, those that are nurse *schedule* related and those that are *roster* related. We propose a decomposition approach to construct solutions that consists of two stages: 1) to construct high quality sequences for nurses by only considering the sequence constraints, and 2) to iteratively construct schedules for nurses and the overall rosters, based on the sequences built and considering the schedule and roster constraints. In the second stage of the schedule construction, nurses are ordered and selected adaptively according to the quality of the schedules they were assigned to in the last iteration. Greedy local search is carried out during and after the roster construction, in order to improve the (partial) rosters built. We show that the local search heuristic during the roster construction can further improve the constructed solutions for the benchmark problems tested.
In addition, we introduce new benchmark nurse rostering datasets which are based upon real world data. The data sets represent a variety of real world constraints. The publication of this problem data to the research community is aimed at closing the gap between theory and practice in nurse scheduling research. One of the main objectives is to encourage more research on these data sets.

Keywords: adaptive heuristic, benchmark, nurse rostering, shift sequence

1. Introduction

Nurse rostering problems consist of allocating the required workload to nurses subject to a number of constraints (see [14,20,33]). The common objective of nurse rostering is to efficiently utilise resources and thus to produce rosters with a balanced workload as well as to satisfy individual preferences as much as possible.

Constraints are usually categorised into two groups: hard and soft constraints, which vary significantly with respect to legal regulations and individual preferences, depending on individual institutions and countries. *Hard constraints* must be satisfied to obtain *feasible* solutions. *Soft constraints* are desirable but not obligatory, and thus can be violated. In real nurse rostering settings, we noticed that the problems are

* Corresponding author: Rong Qu, email: rxq@cs.nott.ac.uk, tel: .+44 115 8466503

nearly always over-constrained. It is therefore quite common to express the quality of solutions in terms of soft constraint violations.

Early research in nurse rostering was overviewed in [30,35]. These early approaches were effective in solving small scale problems but were not flexible and were too computationally expensive to deal with large problems with many constraints.

Meta-heuristics have been widely studied in nurse rostering over the past two decades or so and have had significant success for a range of problems (see [14]). Relevant approaches include Tabu Search [12,19,34], Simulated Annealing [8,23], Evolutionary Algorithms [2,3,10,22,31] and Variable Neighbourhood Search [9,11,15]. Hybrid techniques, particularly between meta-heuristics and mathematical programming methods have formed an important part of the nurse rostering literature (see [5,6]). Artificial intelligence approaches such as Case-Based Reasoning [7,29,32], constraint based techniques [1,24,26,27] and Expert Systems [18] have also been investigated with some success.

Most nurse rostering models apply hard and soft constraints. However, in [21], constraints were categorised into *shift constraints* (which concerned the number of staff and the skill category required for each shift), and *nurse constraints* (which considered the workload for each nurse including nurse preferences, consecutive shifts and intervals between shifts). The nurse constraints were used to produce all feasible shift patterns of the whole scheduling period for each nurse, independently from shift constraints. The best combinations of these shift patterns were found using mathematical programming and metaheuristics [21].

In [32], a decision support system framework was presented, in which a pattern database, a schedule database and a nurse database were employed address solving nurse rostering problems. Shift patterns were produced for one week and a screening process was used to eliminate undesirable ones. Low penalty patterns were combined to generate cyclic work schedules using a local search method. Only cyclic schedules of complete weeks were addressed in the decision support system.

In [2], all the feasible weekly shift patterns were pre-defined and associated with costs concerning preferences, requests, the number of successive days, etc. These (in total 411) shift patterns were then used to construct nurse rosters by employing different heuristic decoders within a genetic algorithm to schedule both shifts and patterns for the best permutations of nurses. The idea of permuting the nurses to be scheduled is similar to the method presented in this paper. Two types of shifts (Night and Day) were addressed and the pre-construction of patterns required a large amount of time. In [4], exactly the same problems were solved by using a Bayesian network, which explicitly learned the rules on selecting the patterns to be assigned to the nurses. The results, compared with those obtained in [2] indicated the strength of the approach.

Valouxis and Housos (2000) [34] explored workstretch in an Integer Linear Programming model. Extra constraints on the minimum and maximum lengths of the workstretch were introduced to simplify the model so that only feasible workstretch was tackled.

In [25], Mason and Smith introduced workstretch cost and workstretch transition in an Integer Programming model to define the cost of the *day-on* within and between the workstretch. Column generation was employed to decide on the content of the workstretch and to link them in constructing the schedules concerning other costs related to shifts.

The approach in this paper draws upon that of [21], where a concept called *stint* is introduced to define a feasible sequence of shifts on consecutive days. Schedules for nurses can then be constructed by using a series of *stints*. Millar and Kiragu [28] used the term *stint* to denote patterns, which were defined by a start

date, a length, a cost and the shifts. Network programming was used where each node is a *stint* to construct either cyclic or non-cyclic rosters.

In our previous work [9], high quality pre-defined schedules have been employed to construct cyclic schedules for a group of nurses with the same requirements. Based on these partial cyclic schedules, the rest of the shifts were assigned to the rest of the nurses with different requirements. The problems can thus be seen as being decomposed into cyclic and non-cyclic parts. A Variable Neighbourhood Search developed in [15] was used to further improve the generated rosters. Preliminary results indicated promising directions on high quality pre-defined schedules.

In this work, we further explore this idea by considering patterns, which are sequences of shifts in constructing nurse schedules and rosters. The generated rosters contain non-cyclic schedules, which are applicable to more general real-world problems [14]. We consider patterns rather than individual shifts (as in many other nurse rostering models) for constructing the rosters. While nurses' schedules can be seen as shift sequences within the scheduling period, the issues of shift patterns and workstretch have received far less attention in nurse rostering (see above).

In Section 2, we present the problem formulation. The two-stage approach and the benchmark problems are described in Section 3 and Section 4, respectively. Section 5 reports on experimental results and addresses important issues concerning adaptation. Finally, we present conclusions and future work in Section 6.

2. Problem Formulation

A solution of the nurse rostering problem consists of a collection of personal schedules for each of the nurses. A schedule for a nurse usually consists of shift sequences that possibly have different lengths and different types (i.e. night shifts, early shifts). The shifts in a sequence must be performed on consecutive days, one shift per day. Between the sequences in a schedule (which are called *day-on* periods) there are *day-off* periods, i.e. days without shifts.

Accordingly we define the following terms that are frequently used in the rest of the paper.

- By *sequence* we mean a series of shifts for nurses. A sequence penalty indicates the quality of the sequence. A set of sequences is associated with each nurse.
- A *schedule* is an ordered list of sequences and day-off periods for a single nurse of a certain skill. Category. A schedule penalty indicates each schedule's quality. The length of a schedule is equal to the total number of days-off and days-on.
- A *roster* describes the overall solution. It consists of schedules of the same length, one for each nurse. The quality of the roster is determined by the violations of the constraints.

We categorise the constraints as *sequence*, *schedule* and *roster* constraints, which are listed below:

- *Sequence* constraints are applied within shift sequences (i.e. to construct shift sequences and to evaluate their quality) for each nurse with certain skills (i.e. each nurse is associated with only those valid sequences satisfying constraint No. 1 in Table 1).
- *Schedule* constraints are applied to combine shift sequences (i.e. to construct schedules and evaluate their quality) for each nurse.
- *Roster* constraints are applied when combining nurse schedules (i.e. to construct an overall solution) while satisfying the coverage requirements on each day.

Furthermore, hard constraints and soft constraints are presented as follows.

- The hard constraints considered in this research are listed in Table 1. The last column in this and the next table indicates the constraint type used in our approach (i.e. sequence, schedule or roster constraints).
- The soft constraints are listed in Table 2. They represent different problem requirements and nurse preferences. Note that this is a complete constraint list representing all the possible constraints considered in the benchmark problems. Different problem instances presented in Section 4 may have different subsets of these constraints.

Hard constraint No. 1 in Table 1 is considered when constructing shift sequences for each nurse of different skills. For example, 'DH', which denotes the day shift for a head nurse, is only applied when constructing shift sequences for a head nurse with a certain skill level. Hard constraint No. 2 states that the total shifts on certain days must satisfy the coverage requirements (i.e. we require enough shifts of a certain type every day). Violations of soft constraints in the solutions are penalised by certain values and are used for evaluating the quality of the rosters produced. The objective is to find feasible solutions that satisfy the soft constraints as much as possible (i.e. minimising the overall penalty of nurses' schedules in the roster).

Table 1. Hard Constraints [13] Categorised into Sequence and Roster Constraints

	Hard Constraint	**Type**
1	Shifts which require certain skills can only be taken by (or assigned to) nurses who have those skills	sequence
2	The shift coverage requirements must be fulfilled	roster

Table 2. Soft Constraints [13] Categorised into Sequence, Schedule and Roster Constraints

	Soft Constraint	**Type**
1	Minimum rest time between shifts	sequence
2	Alternative skill (if a nurse is able to cover a shift but prefers not to as it does not require his/her primary skill)	sequence
3	Maximum number of shift assignments	schedule
4	Maximum number of consecutive working days	sequence
5	Minimum number of consecutive working days	sequence
6	Maximum number of consecutive non-working days	schedule
7	Minimum number of consecutive non-working days	schedule
8	Maximum number of hours worked	schedule
9	Minimum number of hours worked	schedule
10	Maximum total number of assignments for all Mondays, Tuesdays, Wednesdays, etc	schedule
11	Maximum number of a certain shift type worked (e.g. maximum seven night shifts for the planning period)	schedule
12	Maximum number of a certain shift type worked per week (same as above but for each individual week)	schedule
13	Valid number of consecutive shifts of the same type	sequence
14	Free days after night shifts	schedule
15	Complete weekends (i.e. shifts on both Saturday and Sunday, or no shift over the weekend)	schedule
16	No night shifts before free weekends	schedule
17	Identical shift types during the weekend	schedule
18	Maximum number of consecutive working weekends	schedule
19	Maximum number of working weekends in four weeks	schedule
20	Maximum number of working bank holidays	schedule
21	Shift type successions (e.g. Is shift type A allowed to follow B the next day, etc)	sequence
22	Requested days on or off	schedule
23	Requested shifts on or off	schedule
24	Tutorship (employee X present when employee Y is working)	roster
25	Working separately (employee X not present when employee Y is working)	roster

We present, in Figure 1, a few possible schedules with their associated penalties for an example employee for a four-week period in December. Table 3 contains some of the constraints with their penalties for this employee. Table 4 provides details of the shifts referred to in these constraints in Table 3. For example, the valid length of consecutive late shift 'L' and night shift 'N' (see Table 4 for the definition of 'L' and 'N') is either 3 or 4, respectively (constraint No. 8 in Table 3).

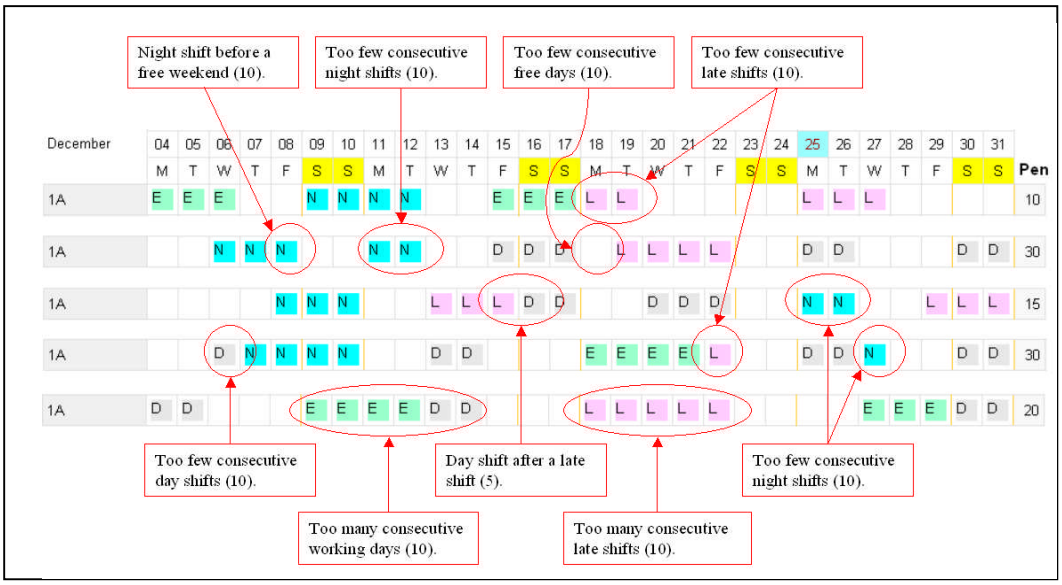

Figure 1. Example Schedules for Employee 1A.

Table 3. Constraints for an Example Employee 1A.

	Constraints	Penalty	Details
1	Max. number of assignments	5	Max. 19 shifts.
2	Max. consecutive working days	10	Max. 5 consecutive working days.
3	Min. consecutive working days	10	Min. 2 consecutive working days.
4	Min. consecutive free days	10	Min. 2 consecutive free days.
5	Max. consecutive working weekends	40	Max. 2 consecutive working weekends.
6	Complete weekends	20	Weekends should either be free or work on all weekend days
7	Identical shift types during weekend	5	
8	Number of consecutive shift types	10	Valid lengths of sequences of consecutive shift types: 'L': 3, 4; 'N': 3, 4; 'D': 2, 3, 4, 5; 'E': 3, 4. (see definition in Table 4)
9	Max. shift types	10	Max. 8 'L' shifts; Max. 5 'N' shifts; Max. 19 'D' shifts; Max. 8 'E' shifts.
10	Max hours worked	1	Max 152 hours.
11	Shift type successions	5	If shift type 'A' is allowed to follow 'B' the next day. Allowed shift combinations: "E,E", "E,D", "E,L", "E,N", "D,D", "D,L", "D,N", "L,L", "N,N"
12	No night shift before a free weekend	10	A shift should not overlap with a free weekend

Table 4. Shift Types Referred to in Employee 1A's constraints

Shift Label	Shift Type	Time Period of the Shift
D	Day shift	08:00-16:30
E	Early shift	06:30-14:30
L	Late shift	14:00-22:00
N	Night shift	21:30-07:00

3. A Two-Stage Adaptive Approach

In the nurse rostering literature, problems are usually dealt with by constructing the schedules for each individual nurse, or by generating assignments of nurses to shifts for each day of the scheduling period. These approaches can be seen as decomposing the problems into sub-problems whose solutions are afterwards combined by considering the coverage constraints in Table 1 and constraints Nos. 24 and 25 in Table 2, or the other constraints in Tables 1 and 2, respectively.

Our approach decomposes problems in a different way. It consists of two stages. They are listed below and explained in more detail in Sections 3.1 and 3.2.

- A shift sequence's construction with respect to the sequence constraints in Tables 1 and 2; and
- A construction of schedules for the nurses which are combined into a roster with respect to the schedule and roster constraints in Tables 1 and 2.

The overall idea of our approach is to reduce the problem complexity by grouping the constraints into the two categories, each of which is considered separately in each stage of the solution approach. By constructing all the valid high quality shift sequences, and by constructing the roster based on these sequences only, the problems can be seen as being decomposed into two parts. Each part of the approach deals with a sub-problem with a much smaller search space than that of the original problem.

3.1 The Construction of Shift Sequences

In this stage of the approach, the best shift sequences are constructed for each nurse of different skills in each problem. They are collected and ranked by their penalties for easy retrieval in stage 2 of the approach (see Section 3.2). By systematically generating a large collection of all the possible permutations of shift types over relevant periods, all the possible shift sequences are enumerated for each nurse in each problem. The ones satisfying sequence constraint No. 1 in Table 1 (skill requirement) and with the lowest total penalty are collected. This set of shift sequences will not be changed in the later stage of the approach. Table 5 presents examples of some shift sequences generated for employee 1A, whose soft constraints are given in Table 3.

Table 5. Example of some shift sequences for the example problem in Table 4

Shift Sequence	Penalty	Comment
D, D	0	
E, E, E	0	
D, D, E, E, E	5	E not allowed to follow D.
L, L, L, D, D	5	D not allowed to follow L.
N, N	10	Two N's not allowed.
E, D, D	10	One E not allowed.
D, D, E	15	E not allowed to follow D. One E not allowed.
L, D, D	15	D not allowed to follow L. One L not allowed.
L, N	25	N not allowed to follow L. One N not allowed. One L not allowed.
D, E, E	25	E not allowed to follow D. One D not allowed. Two E's not allowed
D, E, D, E	50	E not allowed to follow D. One E not allowed. One D not allowed.

To decrease the complexity, we limit the number of possible valid shift sequences by either considering only sequences with a penalty below a certain threshold (≤ 5 or 10), or by selecting the best 50 sequences for each nurse in the second stage of the approach. The numbers are set by initial

experiments and found to be applicable for all the benchmark problems presented in Section 4. For example, in Table 5, the sequence 'E E E' is generated for employee 1A with no penalty. The sequence 'D D E' in Table 5 has a penalty of 15 as E is not allowed to follow D (penalty 5) and a sequence of consecutive days of E shifts must be longer than two (penalty 10). For this employee, there are actually thirteen sequences (of maximum length five) with penalty zero; two with penalty five; thirty nine with penalty ten, twenty eight with penalty fifteen and so on.

Also, only sequences of length up to 5 are considered at this stage. Sequences could be longer, depending on the problems in hand. To construct sequences of length greater than 5, we combine these sequences of length up to 5 to produce new sequences. This is performed in the schedule and roster construction stage.

3.2 The Construction of Schedules and a Roster Based on Shift Sequences

In the second stage of the approach, schedules for each nurse are constructed iteratively, using the shift sequences produced in stage 1, as described in Section 3.1, by considering the *Schedule* constraints for the particular nurse. Based on the schedules of nurses, the overall roster will be constructed by considering the *Roster* constraints in Section 3.1.

Algorithm 1 presents the pseudo-code for the overall two-stage approach for constructing the nurse schedules and overall roster. It is an adaptive iterative method where nurses that were difficult to schedule in previous iterations are scheduled first at the current iteration. That is, the nurses who received the highest schedule penalties in the last iteration are scheduled first.

```
Algorithm 1: ConstructRoster()
// stage 1
1. construct and rank the shift sequences for each nurse
// stage 2
2. iteration = 0
3. set MaxNoIterations
4. randomly order the nurses
5. while iteration < MaxNoIterations
    6. for each nurse ∈ the ordered list of nurses
        7. ConstructSchedule(nurse, partial_roster)
        8. greedy local search to improve the partial_roster
    9. store the best roster constructed so far
    10. calculate the penalty for the schedule of each nurse
    11. sort the nurses by their schedule's penalty in a non increasing order
    12. increase the counter iteration
```

Algorithm 2 presents the pseudo-code for the schedule construction process. It builds a schedule for the nurse based on the partial roster built so far for other nurses and returns its penalty to *Algorithm 1*. The overall guidance of *Algorithm 2* is to generate a schedule with a low penalty value for the nurse, using low penalty shift sequences. In *Algorithm 1* and *Algorithm 2*, the schedule, sequence and roster constraints are combined (i.e. both are used) in penalty calculations and hard constraint violation checks. As *Algorithm 1* is carried out iteratively while concerning the coverage requirement, and different orders of nurses are considered during iterations, the cost of the schedules for each nurse may be different at each iteration. Different shift sequences may be selected to construct different schedules, resulting in different rosters.

Algorithm 2: ConstructSchedule(nurse, partial_roster)

1. set final_threshold
2. current_threshold = 0
3. while (current_threshold ≤ final_threshold)
   4. for each shift sequence ∈ the ranked list for the nurse do
      5. for each day from the first day in the planning period
         6. assign the sequence's corresponding shifts from this day based on the partial_roster, if it does not violate any hard constraint and the penalty occurred <= current_threshold
   7. increase the value of current_threshold by 5
8. return the schedule

The basic idea is to assign as many shifts as possible to the most constrained nurse schedule, so as to obtain the lowest possible penalty. First, the shift sequences with the lowest penalty in the ranked list are considered. If no valid assignment can be made for the current nurse, the shift sequence with the second lowest penalty is considered and so on. The sequences are assigned for the current nurse if the penalty of assigning them is under the current threshold.

During the roster construction, and after a schedule has been generated for the current nurse, an improvement method based on an efficient greedy local search is carried out on the partial roster. It simply swaps any pair of shifts between two nurses in the (partial) roster, as long as the swaps satisfy hard constraints and decrease the roster penalty. By doing this, some of the days in the roster may be assigned shifts that cannot be made in the schedule construction. The method stops when no further improvement can be made. Then, the schedule for the next nurse selected is constructed based on the partial roster built so far, and improved by the greedy local search afterwards. The aim is to reduce the overall penalty of the partial rosters generated for all the nurses scheduled so far.

After all the schedules have been constructed and a roster has been built, there may still be some shifts for which the coverage is not satisfied. To repair this, a greedy heuristic is used. Each extra shift to be assigned is added to the nurse's schedule whose penalty decreases the most (or increases the least if all worsen) on receiving this shift. After this repair step, the local search is applied once more to improve the quality of the overall roster.

In real-world nurse rostering, the number of nurses needed in hospital wards is planned before the definition of the problem, i.e. the problem requires enough nurses of certain skill level to satisfy the coverage requirement. For example, if there is a requirement for one head nurse shift on each day during the scheduling period, then at least one head nurse is included in the problem.

4. Benchmark Nurse Rostering Problems

It is well known that in nurse rostering, researchers tend to deal with problems defined by local hospitals in different countries and, often, with simplifications of those problems [14]. This makes comparisons and evaluations of different approaches very difficult, if not impossible. In [14], it is stated that "it is extremely important to start building up easily accessible benchmark problems" in nurse rostering.

In this paper, we introduce a benchmark nurse rostering problem data set that consists of 11 different problem instances. The data sets were collected from rather small departments at real hospitals, using the nurse rostering model and algorithms developed at KaHo Sint-Lieven [13]. Before having made this data

publicly available, we had to anonymize it, remove any confidential information and removing some rather country/location specific constraints in order to make the data more useable for other researchers. Obviously, this means that the data sets are slightly simpler than those presented in [13]. Many of the instances are still far from trivial, however. The complexity loss is small as these problems retain most of the original constraints.

Table 6 presents the characteristics of the 11 problem instances in the data set. They are non-cyclic problems and are significantly different from each other, not only with respect to the number of nurses (ranging from 4 to 46), the number of skill levels (1 or 2) and the number of shift types, but also the number of personal requests (constraint 22 and 23 in Table 2) and the length of the scheduling period (ranging from 26 to 31 days). There are also different numbers of shift types (ranging from 3 to 6) in different problems. Note that the shift types in different problems may define different start and end times within a day. Rather than presenting specific constraints of fixed characteristics for particular problems, this set of benchmark data represents a wide variety of nurse rostering problems with non-trivial properties which are derived from complete real world complex instances.

The data set is available at http://www.asap.cs.nott.ac.uk/projects/nmhpr/data/index.html, where problems are presented using XML, which is a very flexible, yet simple, format widely used in exchanging many types of data. The representation is easily extendible to add new characteristics and is flexible in presenting different problems in a variety of situations. At the same website, we also provide API functions that can be called to evaluate the quality of rosters as a standard measure to ensure the accuracy of any new result. The web site also presents additional details about the data sets.

Table 6. Characteristics of the Benchmark Nurse Rostering Problem. (Instances BCV-1.8.1, BCV-1.8.2 and BCV-1.8.3 are similar instances which differ in adjustments to some of the constraint parameters and to the personal requests.)

Problems	BCV-1.8.1	BCV-1.8.2	BCV-1.8.3	BCV-2.46.1	BCV-3.46.1	BCV-4.13.1	BCV-5.4.1	BCV-6.13.1	BCV-7.10.1	BCV-8.13.1	BCV-A.12.1
No. of Nurses	8	8	8	46	46	13	4	13	10	13	12
No. of Shift Types	4	4	4	3	3	4	4	4	6	4	4
Period (Days)	28	28	28	28	26	29	28	30	28	28	31
No. of Skill Levels	2	2	2	1	1	2	1	2	1	2	2

5. Experimental Results

We have tested the two-stage approach to solve the modified benchmark problem instances presented in Section 4. Two sets of experiments, with and without the adaptive ordering of nurses (step 11 in *Algorithm 1*), have been carried out. The results are presented in Table 7 and Table 8, respectively. In each table, the results for two variants of the approach are reported: the approach reported in *Algorithm 1* and *Algorithm 2*; and the approach without the greedy local search (step 8 in *Algorithm 1*). When applying the algorithm, we resolved ties (when ordering the nurses with respect to schedule panelties and when selecting shift sequences in order of their quality) by creating a random order. For each problem instance, 8 runs with different seeds for the random generator are carried out, of which the best, worst and average value of the results are reported. The best results are highlighted in bold. The experiments are carried out on a Pentium IV 2.4GHz machine.

We can see from both Table 7 and Table 8 that greedy local search during the roster construction greatly improves the solution quality. The adaptive ordering of nurses during the iterative process also has a positive effect on the results of the construction approach (the results in Table 8 are generally better than

those in Table 7). This can be seen in Table 9, where the average improvements using greedy local search and adaptive ordering are presented.

Table 7. Results from the Approach without Adaptive Ordering on the Benchmark Nurse Rostering Problems (GLS: Greedy Local Search; best results and best average are in bold).

Problems	**BCV-1.8.1**	**BCV-1.8.2**	**BCV-1.8.3**	**BCV-2.46.1**	**BCV-3.46.1**	**BCV-4.13.1**	**BCV-5.4.1**	**BCV-6.13.1**	**BCV-7.10.1**	**BCV-8.13.1**	**BCV-A.12.1**
with GLS best	352	947	**283**	**1594**	**3724**	**18**	**200**	**986**	472	**148**	**3335**
with GLS worst	378	1012	517	1680	3897	130	200	1528	596	160	4540
with GLS avg	**365**	**975**	347	1629	**3789**	84	**200**	**1209**	507	151	**3972**
time (s)	6	6	7	191	137	10	1	15	10	11	44
without GLS best	**323**	**930**	315	1603	3795	19	**200**	1251	**427**	**148**	3845
without GLS worst	537	1134	347	1646	3911	155	200	1519	543	153	4688
without GLS avg	414	1024	**331**	**1626**	3861	**81**	**200**	1361	**489**	**149**	4191
time (s)	3	2	2	21	21	3	0	4	4	3	13

Table 8. Results from the Approach with Adaptive Ordering on the Benchmark Nurse Rostering Problems (GLS: Greedy Local Search; best results and best average are in bold)

Problems	**BCV-1.8.1**	**BCV-1.8.2**	**BCV-1.8.3**	**BCV-2.46.1**	**BCV-3.46.1**	**BCV-4.13.1**	**BCV-5.4.1**	**BCV-6.13.1**	**BCV-7.10.1**	**BCV-8.13.1**	**BCV-A.12.1**
with GLS best	**362**	942	**288**	**1596**	**3601**	**21**	**200**	1037	**396**	**148**	**3570**
with GLS worst	413	1023	383	1620	3761	130	200	1544	601	153	4450
with GLS avg	**388**	974	**330**	**1609**	**3705**	**66**	**200**	**1334**	540	**150**	**3856**
time (s)	6	6	6	196	184	10	1	15	10	11	47
without GLS best	363	**929**	314	1606	3669	67	**200**	**890**	**396**	**148**	3743
without GLS worst	546	997	483	1636	3894	107	200	1584	586	160	4908
without GLS avg	431	**964**	353	1624	3804	76	**200**	1346	**467**	152	4239
time (s)	2	2	2	20	19	3	0	4	3	3	14

Table 9. Average % improvements from greedy local search or adaptive ordering over all data sets

	Average % improvement using GLS compared with no GLS	Average % improvement using adaptive ordering compared with no adaptive ordering
Best case	3.8	1.7
Worst case	3.4	0.3
Average	3.7	0.5

Note that, without the greedy local search heuristic, the approach is purely constructive. It obtains the result fairly quickly (within 21 seconds for the largest instances). Moreover, it does not require much fine-tuning of the parameters when dealing with specific problems. The only parameter that can be changed is the increment value of current_threshold. All the problem instances (having very different characteristics) have been solved using the same constructive method.

6. Conclusions and Future Work

In this paper, we introduced a set of benchmark nurse rostering problems and reported the first results on this data. To encourage scientific comparisons we have set up a web site at http://www.asap.cs.nott.ac.uk/projects/nmhpr/data/index.html, where XML is used to present different problems. The representation can be easily extended to include new constraints. So either similar or very different instances can be easily introduced. We also provide a standard API evaluation function and updated solutions for the problems whenever newly published results become available. The aim is to bring up easy scientific comparisons on the complex nurse rostering problems with a variety of constraints. We welcome more nurse rostering research upon this data set.

We investigated a constructive approach where constraints in nurse rostering problems are grouped into *sequence*, *schedule* and *roster* constraints. Each of the groups is dealt with respectively in constructing shift sequences, constructing schedules for nurses based on the sequences obtained, and constructing the overall roster. The problem complexity can thus be considerably reduced as problems can be seen as decomposed into two parts, each of which is only concerned with the related constraints and thus is much smaller.

When constructing the schedules, the algorithm adaptively orders and selects nurses according to the difficulty of constructing their schedules in the previous iteration. Nurses that received high penalty schedules in previous iterations of the schedule construction will be considered early in the current iteration, which reduces the chance of having trouble with generating their individual schedule later on in the process. The approach is simple and efficient. It obtains reasonably good results quickly for a set of real world, newly presented benchmark nurse rostering problems. It is purely constructive and does not involve much fine-tuning effort on parameters for different problems. The adaptive selection is a robust strategy that does not rely on any domain knowledge.

It is also observed that the greedy local search carried out between the schedule constructions for each nurse greatly improved the roster constructed. This indicates that the approach can be easily adapted for hybridisation with other techniques (i.e. exact methods and meta-heuristics). It will be particularly interesting to perform a study using exact methods to obtain the best combinations for high quality schedules or rosters. Operating on sequences, rather than on individual shifts by meta-heuristics, is also an interesting direction for future work. More strategies of adaptive selection of nurses and sequences will be explored, employing new general methodologies such as hyper-heuristics (e.g. [16,17]).

Acknowledgements

This work was supported by EPSRC grants GR/S31150/01 and GR/T23374/01. We would like to thank for the anonymous referees for their constructive comments and suggestions.

References

1. Adbennadher S. and Schlenker H. (1999), Nurse Scheduling using Constraint Logic Programming. Eleventh Annual Conference on Innovative Applications of Artificial Intelligence, IAAI-99, 838-843. July, 1999, Orlando, Florida, USA.
2. Aickelin U. and Dowsland K. (2003), An Indirect Genetic Algorithm for a Nurse Scheduling Problem. *Journal of Operations Research Society*, **31**(5): 761-778.

3. Aickelin U. and Dowsland K. (2000), Exploiting Problem Structure in a Genetic Algorithm Approach to a Nurse Rostering Problem. *Journal of Scheduling*, **3**(3): 139-153.

4. Aickelin U. and Li J. (2006), An Estimation of Distribution Algorithm for Nurse Scheduling. To appear in *Annals of Operations Research*.

5. Bard J. and Purnomo H.W. (2005), Preference Scheduling for Nurses using Column Generation. *European Journal of Operational Research*, **164**: 510-534.

6. Beaumont N. (1997), Scheduling Staff using Mixed Integer Programming. *European Journal of Operational Research*, **98**: 473-484.

7. Beddoe G. and Petrovic S. (2006), Determining Feature Weights Using a Genetic Algorithm in a Case-Based Reasoning Approach to Personnel Rostering. *European Journal of Operational Research*, **175**: 649-671.

8. Brusco M.J. and Jacobs L.W. (1995), Cost Analysis of Alternative Formulations for Personnel Scheduling in Continuously Operating Organisations. *European Journal of Operational Research*, **86**: 249-261.

9. Brucker P., Qu R., Burke E.K. and Post G. (2005), A Decomposition, Construction and Post-processing Approach for a Specific Nurse Rostering Problem. In: Kendall G., Lei L. and Pinedo M. (eds.), Proceeding of the 2nd Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA'05), 397-406. New York, USA, July, 2005.

10. Burke E.K., Cowling P., De Causmaecker P. and Vanden Berghe G. (2001), A Memetic Approach to the Nurse Rostering Problem. *Applied Intelligence*, **15**: 119-214.

11. Burke E.K., De Causmaecker P., Petrovic S. and Vanden Berghe G. (2004), Variable Neighborhood Search for Nurse Rostering Problems. Chapter 7 in Metaheuristics: Computer Decision-Making (Combinatorial Optimization Book Series), Resende M.G.C. and de Sousa J.P. (eds.) Kluwer, 153-172.

12. Burke E.K., De Causmaecker P. and Vanden Berghe G. (1999), A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. Selected Papers from the 2nd Asia Pacific Conference on Simulated Evolution and Learning (SEAL'98), Lecture Notes in Artificial Intelligence, vol. 1585, 187-194.

13. Burke E.K., De Causmaecker P. and Vanden Berghe G. (2004), Novel Meta-heuristic Approaches to Nurse Rostering Problems in Belgian Hospitals. Chapter 44 in J. Leung: Handbook of Scheduling: Algorithms, Models and Performance Analysis, CRC Press, 44.1-44.18.

14. Burke E.K., De Causmaecker P., Vanden Berghe G. and Van Landeghem H. (2004), The State of the Art of Nurse Rostering. *Journal of Scheduling*, **7**(6): 441-499.

15. Burke E.K., Curtois T., Post G., Qu R. and Veltman B. (2005), A Hybrid Heuristic Ordering and Variable Neighbourhood Search for the Nurse Rostering Problem. To appear at *European Journal of Operational Research*, 2007.

16. Burke E.K., Hart E., Kendall G., Newall J., Ross P. and Schulenburg S. (2003), Hyperheuristics: an Emerging Direction in Modern Search Technology. In: Glover F. and Kochenberger G. (eds). Handbook of Meta-Heuristics, Kluwer 457-474.

17. Burke E.K., Kendall G., and Soubeiga E. A Tabu-Search Hyperheuristic for Timetabling and Rostering. *Journal of Heuristics*, **9**(6):451-470, 2003.

18. Chen J.G. and Yeung T. (1993), Hybrid Expert System Approach to Nurse Scheduling. *Computers in Nursing*, **11**(4): 183-192.

19. Dowsland K. (1998), Nurse Scheduling with Tabu Search and Strategic Oscillation. *European Journal of Operational Research*, **106**: 393-407.

20. Ernst A.T., Jiang H., Krishnamoorthy M. and Sier D. (2004), Staff Scheduling and Rostering: A Review of Applications, Methods and Models. *European Journal of Operational Research*, **153**: 3-27.

21. Ikegami A. and Niwa A. (2003), A Subproblem-centric Model and Approach to the Nurse Scheduling Problem. *Mathematical Programming*, **97**(3): 517-541.

22. Jan A., Yamamoto M. and Ohuchi A. (2000), Evolutionary Algorithms for Nurse Scheduling Problem. Proceedings of the 2000 Congress on Evolutionary Computation (CEC'00), 196-203. San Diego, USA, 2000.

23. Jaszkiewicz, A. (1997), A Metaheuristic Approach to Multiple Objective Nurse Scheduling. *Foundations of Computing and Decision Sciences*, **22**(3): 169-184.

24. Jaumard B., Semet F. and Vovor T. (1998), A Generalized Linear Programming Model for Nurse Scheduling. *European Journal of Operational Research*, **107**: 1-8.

25. Mason A. and Smith M. (1998), A Nested Column Generator for Solving Rostering Problems with Integer Programming. In: Caccetta L., Teo K.L., Siew P.F., Leung Y.H., Jennings L.S. and Rehbock V. (eds.) Proceedings of the 4th International Conference on Optimisation: Techniques and Applications. 827-834. July, 1998, Perth, Australia.

26. Meisels A., Gudes E. and Solotorevsky G. (1996), Employee Timetabling, Constraint Networks and Knowledge-Based Rules: A Mixed Approach. In: Burke E.K. and Ross P. (eds.) Selected Papers from the 1st International Conference on Practice and Theory of Automated Timetabling. Lecture Notes in Computer Science, vol. 1153, 93-105.

27. Meyer auf'm Hofe H. (2000), Solving Rostering Tasks as Constraint Optimisation. In: Burke E.K. and W. Erben. (eds.) Selected Papers from the 3rd International Conference on Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2079, 280-297.

28. Millar H.H. and Kiragu M. (1998), Cyclic and Non-cyclic Scheduling of 12h Shift Nurses by Network Programming. *European Journal of Operational Research*, **104**: 582-592.

29. Petrovic S., Beddoe G. and Vanden Berghe G. (2003), Storing and Adapting Repair Experiences in Personnel Rostering. In: Burke E.K. and P. De Causmaecker (eds.) Selected Papers from the 4th International Conference on Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2740, 148-165.

30. Pierskalla W. and Rath G. (1976), Nurse Scheduling using Mathematical Programming. *Operational Research*, **24**(5): 857-870.

31. Post G. and Veltman B. (2004), Harmonious Personnel Scheduling. In: Burke E.K. and Trick M. (eds.), Proceedings of the 5th International Conference on the Practice and Automated Timetabling, 557-559. August 2004, Pittsburgh, PA USA.

32. Scott S. and Simpson R.M. (1998), Case-Based Incorporating Scheduling Constraint Dimensions: Experiences in Nurse Rostering. In: Smyth and Cunningham (eds.) Advances in Case Based Reasoning. Lecture Notes in Artificial Intelligence, vol. 1488, 392-401.

33. Sitompul D. and Randhawa S. (1990), Nurse Rostering Models: A State-of-the-Art Review. *Journal of the Society of Health Systems*. **2**(1): 62-72.

34. Valouxis C. and Housos E. (2000), Hybrid Optimisation Techniques for the Workshift and Rest Assignment of Nursing Personnel. *Artificial Intelligence in Medicine*. **20**: 155-175.

35. Warner M., Keller B. and Martel S. (1990), Automated Nurse Scheduling. *Journal of the Society for Health Systems*. **2**(2): 66-80.