Doctor Rostering in Compliance with the New UK Junior Doctor Contract

Anna Lavygina[1,2](B), Kris Welsh[2], and Alan Crispin[2]

[1] Servicepower Business Solutions, Petersgate House, Stockport SK1 1HE, UK
a.lavygina@servicepower.com
[2] School of Computing, Mathematics and Digital Technology,
Manchester Metropolitan University, Manchester M1 5GD, UK
{k.welsh,a.crispin}@mmu.ac.uk

Abstract. In 2016 the UK government imposed a new contract on junior doctors working for the country's National Health Service. This new contract significantly changed the way in which hospitals and health trusts create rosters, introducing new constraints and a system of fines levied against employers should a doctor be required to work an undesirable or potentially unsafe shift pattern. In this paper, we present a new rostering problem set based upon this new junior doctor contract that models hospital departments varied in size, cover requirements, and contracted working patterns. We present the results of experiments in creating valid rosters for our problem set using a construction heuristic, and optimised using simulated annealing.

1 Introduction

The United Kingdom provides all citizens with free healthcare via its National Health Service (NHS). Although doctors working for the NHS have a number of job titles and roles, they can broadly be divided into three categories: *junior doctors*, *senior doctors*, and *consultants* [1]. Doctors categorised as junior doctors are those who have not yet completed training in their chosen specialty, which may take up to eight years from graduation. Doctors categorised as senior doctors are those who have completed their specialist training, whilst consultants are a subset of senior doctors who take overall responsibility for a patient's care. NHS junior doctors are employed under standardised terms and conditions, set out in the junior doctor contract [2], which was changed in 2016.

Before 2016 the junior doctor contract discouraged employers from rostering a doctor for a large number of hours, or for significant quantities of night and weekend work by increasing the doctor's pay based on the number of hours worked on average, and via an assessment of how *antisocial* the hours are. For example, doctors who worked 48 h a week on average would be paid an additional 20% of their stated salary, whereas those who worked 56 h a week on average would be paid an additional 50% of their stated salary if these hours were daytime weekday work. If the same hours were worked including significant numbers

of night or weekend shifts, the percentages could increase to 50% and 80%, respectively [3].

The previous junior doctor contract included a small number of constraints on working patterns (minimum rest period between shifts, maximum total hours worked), violations of which would result in the affected doctor's pay being doubled. This system of increasing pay to reflect the desirability of worked shift patterns provides a simple objective measure of the quality of a roster at both the individual doctor and the overall level: monetary cost. Thus, automated rostering of junior doctors under this previous contract was closely related to the classical nurse rostering problem [4].

The 2016 junior doctor contract was introduced with a number of aims, including: (i) encouraging hospital departments offering elective treatment to operate seven days a week by removing the pay premium associated with weekend work, and (ii) removing edge cases where a doctor who works a single additional hour a week more than another may be paid many thousands of pounds more. As a result, distinctions between weekday and weekend work have been reduced, and a number of new constraints on working patterns and rest periods have been introduced [2].

The constraints in the 2016 junior doctor contract are more fine-grained, and more complex, than those of the previous contract. For example, a junior doctor who works consecutive night shifts (defined as any shift involving 3 or more hours of work between the hours of 23:00 and 06:00) may work a maximum of four consecutive days. Furthermore, if the doctor has worked three or four consecutive night shifts, they must be followed by a 46 h minimum rest period [2].

Each NHS trust or hospital that employs junior doctors is required to appoint a "guardian of safe working hours", who monitors doctors' working schedules and enforces the constraints specified in the contract. Crucially, some of the new constraints are enforced by means of a fine which is levied by the guardian should they be violated. Thus, there are two objective measures of the quality of a proposed roster: the number of constraint violations, and the total monetary value of the fine that would be levied by the guardian for violating key constraints. Because the constraints enforced by guardian fine are a subset of the constraints overall, there will be a correlation between rosters with fewer constraint violations and lower guardian fines, but the two measures remain distinct.

The constraints that are not subject to a guardian fine remain important, with any violation representing a breach of a doctor's contract of employment. Thus, an employer may choose to adopt a roster that attracts a greater guardian fine in order to minimise the number of (non-fined) constraint violations in some circumstances. Conversely, the employer may instead choose to adopt a roster with a greater number of (non-fined) constraint violations in order to reduce the amount due in guardian fines.

In this paper we use a rostering approach based on a construction heuristic and a simulated annealing algorithm for rostering in compliance with the new contract. We present ten synthetic datasets of different complexity that model hospital departments of different sizes, cover requirements and doctors with differing contracted hours, working patterns and leave arrangements.

The paper is organised as follows. Section 2 describes related work. The new rota rules in the junior doctor contract, and fines for their violation, are discussed in Sect. 3. An example of rostering doctors in a hospital department that is used for testing our approach is introduced in Sect. 4. Section 5 outlines our approach to doctor rostering. Section 6 introduces a new set of ten doctor rostering problems of different complexity. Experimental results are presented in Sect. 7. Finally, Sect. 8 concludes the paper and outlines our future work.

2 Background

Much of the research effort in automated rostering has concentrated on variations of the classical nurse rostering problem [5]. The nurse rostering problem involves finding a duty schedule for nurses in a hospital for a given planning horizon, considering both hard (essential) and soft (desirable) constraints. All hard constraints must be satisfied for the solution to be feasible. Examples of hard constraints in the problem are that each nurse may work only one shift per day, and that all shifts must be allocated to a nurse. Soft constraints must be satisfied as far as possible, with the number of soft constraint violations used as an objective measure of roster quality. Soft constraints are often categorised as either *contract-specific* or *employee*. Examples of contract-specific soft constraints include the minimum/maximum number of assignments during the planning horizon and the minimum/maximum number of consecutive working days. Examples of employee soft constraints include day off requests or shift off requests. Nurses have different skills and grades and these also need to be considered when constructing rosters.

The nurse rostering problem belongs to a class of non-deterministic polynomial-time hard (NP hard) problems which means that the amount of time required to solve a problem grows exponentially with problem size. To measure the quality of a schedule the number of soft constraint violations can be used as a cost measure when optimizing the schedule. Meta-heuristics coupled with local operators can be used to guide a search to a best roster solution using the cost function [6]. Different methods and approaches can tested using competition benchmark nurse rostering datasets [7]. A comprehensive review of the literature for personnel scheduling has been undertaken by Van den Bergh et. al. [8]. The problem has proven attractive, given the clear imperative to maintain appropriate staffing levels for a service that in many cases operates 24/7, and the obvious need to ensure individual staff members are allowed sufficient rest.

Although nurse rostering dominates the research landscape of automated rostering in the healthcare sector, there has been some previous research effort looking into the automated rostering of physicians and doctors [9–12]. This cluster of work is perhaps most similar to ours, but does not contribute any benchmark problem set to aid in testing and comparing. Also, as previously discussed, the new junior doctor contract has new aspects not used in the nurse rostering problem such as the guardian fine representing a breach of a doctors contract of employment. New constraints (see Sect. 3) have been introduced to ensure that doctors have a sufficient amount of rest. Employers are penalised with a fine when they ask doctors to work excessive hours. This means that rosters are not only assessed on the number of constraint violations but the fines generated. In at least some of the real-world scenarios we have encountered in discussions with hospitals, the fines could be so high as to exceed the cost of an additional doctor.

3 New Rota Rules and Safe Working Hours Fine

One of the main claimed purposes of the new junior doctor contract is to encourage safer working patterns for doctors, with adequate rest periods [13] and a greater work-life balance. This is attempted by codifying a number of constraints on doctors' working patterns in their contract [2], as follows:

1. Max 48 h average working week (56 if the doctor has opted out of the European Working Time Directive);
2. Max 72 h work in any 7 consecutive days;
3. Max 13 h length of any one shift;
4. Max 5 consecutive long shifts (>10 h), Min 48 h rest following the 5th long shift;
5. Max 4 consecutive long shifts finishing after 23:00, Min 48 h rest following the 4th shift;
6. Max 4 consecutive night shifts (at least three hours between 23:00 & 06:00), at least 46 h rest following the 3rd or 4th such shift;
7. Max 8 consecutive shifts, at least 48 h rest following the final shift;
8. Max frequency of 1 in 2 weekends can be worked (any shift involving any time between 00:01 Sat & 23:59 Sun);
9. Normally at least 11 h of continuous rest between shifts

Violations of all constraints are permitted, and sometimes unavoidable, but should be minimised. As discussed previously, NHS trusts are required to appoint a "guardian of safe working hours" who levies fines against hospitals if some of the constraints are violated. Fines are levied for violating the first and second constraint, and also if a doctor's rest between shifts is reduced to fewer than 8 h (codifying a stricter measure for violations of constraint 9). The total value of the fine is defined as 4x the doctor's equivalent hourly rate. Of this, 1.5x is paid to the doctor, and the rest is paid to the guardian [2] and used to benefit the education, training and working environment of junior doctors [14].

4 Example: Rostering Doctors in a Hospital Department

To illustrate a typical junior doctor rostering problem and its constraints we use the following scenario, which is a simplified version of a sample scenario we obtained from an NHS hospital. A hospital department uses a shift structure with four overlapping shifts each day, allowing for acute care to be handed over between shifts.

– early day shift (8am–5pm) – 2 doctors required;
– day shift (9am–5pm) – 6 doctors required;
– evening shift (5pm – 8.45pm) – 2 doctors required;
– night shift (8.30pm – 8.45am) – 1 doctor required.

The department employs 12 junior doctors, all of whom are subject to the 2016 junior doctor contract. All of the doctors have declined to opt out of the EUWTD, and are thus limited to working 48 h a week on average. The doctors are all equivalent for the purposes of rostering, with no specific skill requirements. Some doctors do, however, have specific contracted working patterns or conditions.

– Doctor #1 works Monday night shifts, Doctor #2 works Tuesday night shifts, Doctor #3 works Wednesday night shifts. None of these doctors may be assigned a night shift on other days.
– Six other doctors (doctors #4–9) may work night shifts on Thursdays only if this forms part of a full Thu-Sun weekend of night shifts.
– Doctors #10–12 cannot be assigned to night shifts at all.
– No doctor who works a night shift may be rostered for the Early or Day shift the next day.

5 Approach

5.1 Hard and Soft Constraints

Rostering is a highly constrained problem. Constraints are typically divided into two categories: *hard constraints* and *soft constraints*. Hard constraints define the feasibility of rosters and must be satisfied in any valid roster. In this paper, we consider cover requirements (minimum number of employees required for each shift) and the honouring of working patterns as hard constraints. The rostering constraints from the doctors' contracts, including those subject to a guardian fine, are treated as soft constraints.

We categorise types of working pattern that a doctor's contract may stipulate as *fixed* patterns, *conditional* patterns, or *forbidden* patterns. Doctors with *fixed* working patterns are contracted to work specific named shifts on specific days of the week. Doctors with *forbidden* working patterns may not be scheduled on certain (series of, or individual) shifts, on certain days of the week. Doctors may also have a *conditional* contracted working pattern, which stipulates that if they work a specific shift on a certain day they must also work other specified shifts on the following days. It is this type of pattern that we use to codify constraints such as the second in our example, as discussed in the preceding section.

Roster rules are treated as soft constraints, and we sum (unweighted) the number of hours worked in violation of each constraint, as described in Table 1, as a measure of solution quality. We also sum the total number of hours subject to a guardian fine as a second measure. The aim of the optimisation is to find rosters that minimise the number of hours worked in violation of constraints, and to minimise the number of hours subject to a guardian fine.

Table 1. Penalties for roster rules violations

Roster rule	Penalty for each violation occurrence
Max 48 h average working week	Total number of hours worked above the limit in the reference period, plus guardian fine
Max 72 h work in any 7 consecutive days	Total number of hours worked above 72-hour limit, plus guardian fine
Max 13 h shift length	Total number of hours worked above 13-hour limit
Max 5 consecutive long shifts, Min 48 h rest following the 5th shift	Total number of missing rest hours. For example, given 45 h rest after 5th shift, penalty = 48 − 45 = 3
Max 4 consecutive long day/evening shifts, Min 48 h rest following the 4th shift	Total number of missing rest hours
Max 4 consecutive night shifts. At least 46 h rest following the 3rd or 4th such shift	Total number of missing rest hours
Max 8 consecutive shifts, at least 48 h rest following the final shift	Total number of missing rest hours
Max frequency of 1 in 2 weekends can be worked	Total number of hours worked during a weekend that violates the rule
Normally at least 11 h continuous rest between rostered shifts	Total number of missing rest hours. If the rest is reduced to <8 h, a guardian fine will apply

We do not consider doctors working under the pre-2016 contract, nor do our problems contain any on-call work.

5.2 Generation of a Random Valid Roster

For a roster to be valid: (i) All cover requirements must be satisfied. (ii) Doctors must be assigned to the shifts for their fixed patterns, except when on leave. (iii) Any assignment which matches the condition of a conditional pattern must form part of the complete pattern's assignment. (iv) No shift or series of shift assignments must match the relevant doctor's forbidden patterns.

We developed a construction heuristic to generate valid rosters, which is depicted in Fig. 1. At the first stage (lines 1–4) of the heuristic all fixed patterns are assigned to the corresponding employees. At the next stage (lines 6–24) all other shifts are assigned moving day by day. For every day of the scheduling period, firstly, list *mustBeScheduled* is generated (line 7). This is a list of employees that must have a shift assigned on the day to avoid a forbidden pat- tern match, because of a previous assignment. Then, for each shift $shift_i$ of the day we generate list $available_i$. This is a list of employees that can be assigned to this shift i.e. not assigned to any shifts on that day and would not have a forbidden patten match if $shift_i$ is assigned (lines 8–9). Shift assignments are made by: (i) selecting shift $shift*$ with the smallest list of available employees $available*$, (ii) from $available*$ selecting an employee $employee*$ that is available for the least number of shifts, (iii) assigning $shift*$ to $employee*$, (iv) updating lists of employees' availability for all shifts (lines 10–14). The loop is repeated until all shifts have sufficient coverage, as per the problem definition. If any shift assignment matches a first entry of a conditional pattern, then the rest of the pattern is assigned to an employee. After assigning all shifts for the day, list *mustBeScheduled* is checked, and for each employee from that list that does not have any shift assignments, a random shift is selected from the list of shifts that this employee can do, and the employee replaces a random employee already assigned to this shift, removal of whom would not violate a pattern.

```
1  function createRoster(employees, shifts, patterns)
2  foreach fp ∈ fixed patterns:
3    foreach employee fe that have fp:
4      extract all shift series matching fp and assign to fe;
5
6  foreach date ∈ [startDate, endDate]
7    find employees that have to have a shift → mustBeSchedules
8    foreach shift_i ∈ shifts on the day date:
9      find employees that can do shift_i → available_i
10   while not all shifts are fully assigned
11     find shift − > shift* with the smallest available*
12     find an employee ∈ available* that can do the least #shifts → employee*
13     assign shift* to employee*
14     foreach fp ∈ conditional patterns of employee*:
15       if shift matches a first entry of fp
16         extract and assign a shift series matching fp, starting from shift*
17     update availability lists for all shifts
18   foreach employee ∈ mustBeSchedules
19   if employee have no shifts assigned on the date
20       do
21         randomly select a shift employee can do → shift
22         randomly select an employee ∉ mustBeScheduled assigned to shift → employee'
23         replace employee' by employee
24       until replacement is valid
```

Fig. 1. Construction heuristic.

5.3 Optimisation. Simulated Annealing

After the initial random roster is generated, it is improved by using simulated annealing (SA) [15]. The algorithm of SA is shown in Fig. 2. The total penalty for soft constraints violations is used as an objective function for optimisation. SA takes an initial roster (in our case it is the roster generated by the construction heuristic) as an input, and repeatedly applies local operators to make adjustments to this roster in order to find the best combination of shift assignments. Each local operator guarantees its output will be a valid roster, if its input was valid. Thus we optimise solely within the scope of feasible solutions.

– swapping shifts (or series of shifts) between two employees (Fig. 3);
– replacing an employee on the shift by another employee (Fig. 4).

Both operators are applied during optimisation. Parameter $swapProbability \in [0, 1]$ defines the probability of swapping shifts. Employee replacement is applied with probability $1 - swapProbability$. Unlike "greedy" optimisation methods (e.g. hill climbing), simulated annealing (Fig. 2) can accept, with a certain probability, alterations that affect the objective function score adversely. This reduces the risk of getting stuck in local optima, particularly in early iterations. The probability of accepting such alterations $p = e^{\frac{-delta}{T*(1-\frac{i}{it})}}$, where $delta =$

$objective(roster') - objective(roster)$, $objective(roster)$ and $objective(roster')$ are objective values for the current and altered rosters respectively, T is a parameter of the simulated annealing algorithm (initial temperature), i is the number of the current iteration and it is the total number of iterations.

```
function simulatedAnnealing(T, it)
  for $i$ = 0 to $it - 1$
    if $random < swapProbability$
      $roster' = swapShifts(roster)$
     else $roster' = replaceWithAnotherEmployee(roster)$
    $delta = objective(roster') - objective(roster)$
    if $delta <= 0$ or $e^{\frac{-delta}{T*(1-\frac{i}{it})}} > random(0, 1)$
      $roster = roster'$
```

Fig. 2. Simulated annealing algorithm.

```
function swapShifts()
  $swapped = false$
  do
    randomly select an employee $-> emp_1$;
    randomly select a shift assigned to $emp_1 -> shift$;
    if $shift$ does not belong to any fixed pattern instances;
      if shift belongs to a conditional pattern instance $pi$;
        $shifts_1 = pi$
      else $shifts_1 = shift$
      find an employee that can swap their shifts to $shifts1 -> emp2$;
      if $emp2$ found
        find shifts assigned to $emp2$ on the days of $shifts1$ $shifts_1 -> shifts2$;
        if $emp1$ can swap $shifts_1$ to $shifts2$;
            swap $shifts_1$ and $shifts2$ between $emp1$ and $emp2$;
            $swapped = true$;
  while $swapped = false$
```

Fig. 3. Swapping shifts.

Swapping shifts. This operator swaps shifts or a series of shifts between two employees (see Fig. 3). First, an employee emp_1 and a shift assigned to this employee $shift$ are randomly selected. If $shift$ forms part of a fixed pattern, it cannot be swapped, and a new $doctor1 - shift_1$ pair has to be selected. If $shift$ belongs to a conditional pattern instance pi, then with probability 0.5 a swap for the whole pattern instance is attempted: $shifts_1 = pi$, otherwise a swap is sought for the initial shift only: $shifts_1 1 = shift$ (if shift does not belong to any conditional pattern instances, then $shifts1 = shift$ too). Next, an employee

```
function replace()
replaced = false;
do
  randomly select an employee → $emp_1$;
  randomly select a shift assigned to $emp_1$ → shift;
  if shift does not belong to any fixed and conditional pattern instances
  and removing shift will not create a forbidden pattern instance
    do
      randomly select an employee → $emp_2$
      if $emp_2$ can do shift
        unassign $emp_1$ from shift
        assign $emp_2$ to shift
        replaced = true;
    while replaced == false and not all employees are checked
while replaced == false
```

Fig. 4. Replacing a doctor.

whose shifts could be swapped to $shifts1$ is identified. The swap must not lead to breaking any fixed or conditional patterns, or violate a forbidden pattern. If such an employee emp_2 is found with shifts $shifts_2$ that can be replaced by $shifts_1$, then emp_1 is checked to ensure that they can work $shifts_2$ instead of $shifts_1$. If the swap is possible, then $shifts_1$ and $shifts_2$ are swapped between emp_1 and emp_2. If swapping shifts is not possible for any reason (e.g. $shift$ belongs to a fixed pattern, or an employee whose shifts could be swapped with $shifts_1$ is not found, or emp_1 cannot do $shifts_2$), then the whole process is repeated for a new $doctor1 - shift_1$ pair.

Replacing an employee. This operator replaces an employee on a single shift with another employee who is available on the day of the shift and can be assigned to it (see Fig. 4). First, an employee (emp_1) and shift ($shift$) for replacement are selected. If $shift$ does not belong to any fixed or conditional pattern instances and removing it would not violate a forbidden pattern for emp_1, then a replacement doctor for the shift is sought. For this, a random employee emp_2 is selected, and if emp_2 can be assigned to $shift$ (i.e. shift assignment would not break any fixed or conditional patterns, nor violate a forbidden pattern), then the replacement takes place, otherwise a new replacement is sought from the remaining employees. If no replacement can be found, replacement for another pair of (emp_1) and ($shift$) is attempted, until a replacement is made.

6 Reference Problem Sets

We have created ten reference problems to allow researchers to compare solutions generated by different rostering approaches on standardised benchmark problems which require a full year's roster to be created. The problem sets model a range of scenarios, varying in size, complexity and difficulty. All ten datasets use the same basic pattern for coverage requirements with specific numbers of doctors needing to be working during defined *early day* 08:00–17:00, *day* 09:00–17:00, *evening* 17:00–20:45, and *night* 20:30–08:45 time periods. Doctors' shifts, however, are not required to align precisely with these time periods and a roster is valid providing that the minimum numbers of doctors for each time period is met or exceeded for its full duration, regardless of doctors' starting and finishing times.

Each problem in the set varies in terms of the number of doctors required for each of the specified time periods, as well as the number of doctors available and the ratio of doctors required each day to the total available. Doctors in the reference problem sets have also pre-specified their study leave and annual holiday leave arrangements for the time period, with doctors in the later problem sets being more likely to take longer contiguous periods of leave, increasing rostering difficulty in and around these periods.

Doctors in the reference problem sets also vary in whether they have opted out of the European Working Time Directive (EWTD), with doctors who have not opted out limited to a 48 h maximum working week, and those who have opted out limited to a 56 h maximum working week on average. Certain doctors also have individual constraints written into their employment contracts, in one of three forms. Some doctors have one or more fixed, conditional or forbidden working patterns written into their contracts.

Table 2 depicts the combined number of doctors required for each coverage period for each of the problems. Also depicted is the number of available doctors for each problem, how many of these doctors have opted out of the EWTD, and how many of these doctors have one or more contracted working patterns.

Table 2. Summary of key differences between problems in problem set

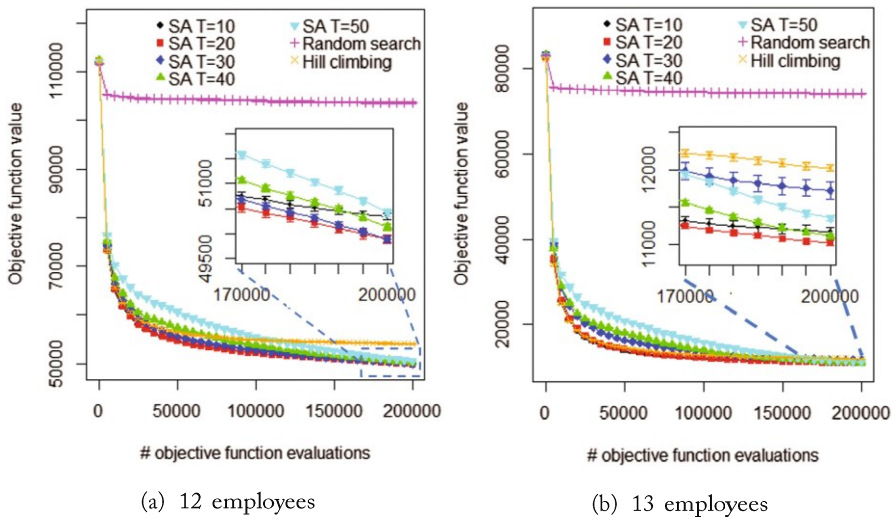
Problem	Combined coverage	Doctors	EUWTD Opt-Outs	Patterns	Average # patterns per doctor
1	7	12	10	6	0.75
2	13	20	15	10	0.55
3	20	30	22	17	0.5
4	7	12	9	10	1.08
5	13	20	9	6	0.8
6	20	30	15	21	0.87
7	28	40	20	26	0.98
8	7	12	7	6	0.83
9	20	30	13	19	0.97
10	13	20	8	14	0.7

Table 2 depicts the principal differences between the problems in the set, but there are other properties that contribute to the later problems posing a generally greater level of difficulty than the earlier ones. For example, the complex conditional patterns are more prevalent in later problems, and doctors take leave in larger blocks in the later problems. We have encoded each of the problems in the set in JSON format, allowing for relatively efficient parsing using standard libraries for most languages and platforms. The complete problem set, including implementation documentation to assist developers in understanding and reading the files, is available at http://bit.ly/2tP181b without restriction.

7 Experimental Results

7.1 Example

In this section we present the results of evaluation of our rostering approach for the example introduced in Sect. 4. For comparison, we also present results for the same problem with an additional, 13th, doctor available. For both scenarios, random valid rosters were generated using the proposed construction heuristic and then improved by optimisation methods. We compared the performance of the simulated annealing algorithm with different initial temperature values and two other optimisation methods: hill climbing algorithm and random search. Figures 5a and b depict the convergence of average objective function across 30 runs per setting for 12 and 13 employees respectively. Figures 6a and b show the convergence of the corresponding average fine value.



(a) 12 employees (b) 13 employees

Fig. 5. Convergence of average objective function

The results show that in the example the cost of a 13th doctor would likely be less than the guardian fine for dangerous working patterns if the department has only 12 doctors. This analysis would prove useful during the introduction of the new working arrangements, and during the planning of new departments.

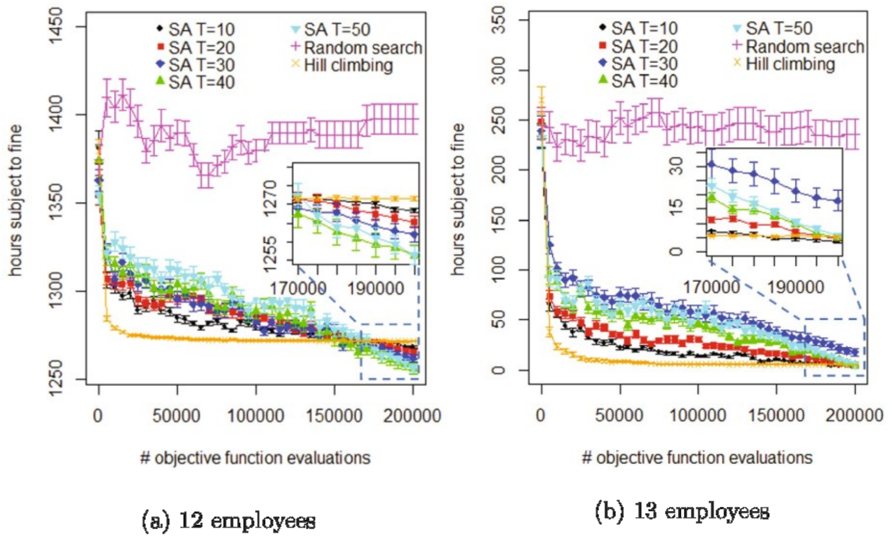


Fig. 6. Convergence of # hours subject to fine

7.2 Results for the Reference Problem Sets

Simulated Annealing, at some initial temperatures, slightly outperform hill climbing for some problems in the reference set, whilst in other performance differences are insignificant. This is likely due to a combination of: (i) a relatively smooth problem space, with few local optima, (ii) the local operators only make relatively small adjustments to the roster, and (iii) the tightly-constrained nature of the problem means that there are relatively few valid solutions to each problem.

In the problem instances in which the performance of hill climbing most closely matches or beats that of simulated annealing, improvement in the objective function comes at the expense of a greater guardian fine. It appears that this typically occurs when the optimiser violates the final constraint used in the calculation of guardian fines: the 8 h minimum rest period between shifts. In circumstances where a doctor has been assigned a series of long/night shifts, they become entitled to a long rest period. If the optimiser replaces one of the long/night shifts in the sequence with an early or day shift, the doctor in question may now be eligible to work shifts on two days following the sequence, as they are no longer entitled to the long rest period. This switch violates the constraint on an 11 h rest period between shifts, but the total number of hours worked in violation of a constraint will be lower. The dangerously low amount of rest between the newly-allocated early/day shift and the preceding night shift attracts a guardian fine, despite the improvement in the objective function score.

The difference between the performance of SA on the reference problems and the example problem is almost certainly related to this observation, as the example problem has an additional soft constraint on the allocation of an early or

Table 3. Rostering results obtained for synthetic rostering problems

Instance	Algorithm	Total # hours violating constraints		Total #fined hours	
		Average	st. error	Average	st. error
Instance 1	Construction heuristic	14392.37	176.63	1125.11	12.36
	SA $T = 1$	7449.30	178.81	549.50	14.21
	SA $T = 5$	7230.60	177.51	509.83	15.25
	SA $T = 10$	7154.25	155.70	509.45	14.40
	SA $T = 20$	7203.53	163.28	544.01	11.83
	Hill climbing	7227.35	136.82	552.61	15.01
Instance 2	Construction heuristic	45508.90	289.36	2590.96	91.50
	SA $T = 0.5$	26220.38	328.12	1669	23.83
	SA $T = 1$	26385.78	277.21	1674.33	20.88
	SA $T = 5$	26646.42	363.01	1637.92	18.31
	SA $T = 10$	26256.35	244.42	1629.91	17.61
	Hill climbing	26631.13	292.35	1662.22	24.47
Instance 3	Construction heuristic	75158.94	372.46	3997.50	21.67
	SA $T = 1$	35275.21	288.862	2070.42	22.26
	SA $T = 5$	34625.76	341.06	2006.49	24.13
	SA $T = 10$	34804.16	402.82	2088.55	19.12
	SA $T = 20$	36168.75	302.00	2216.43	21.59
	Hill climbing	34837.91	226.04	2081.90	16.13
Instance 4	Construction heuristic	15691.08	186.19	1118.20	12.43
	SA $T = 1$	10759.83	171.28	785.56	14.90
	SA $T = 5$	10645.29	195.90	752.88	16.50
	SA $T = 10$	10916.20	232.44	738.03	15.10
	SA $T = 20$	10725.80	209.87	775.98	15.33
	Hill climbing	10666.61	193.90	772.67	14.69
Instance 5	Construction heuristic	47209.34	321.82	2745.52	18.70
	SA $T = 1$	28077.16	276.50	1692.92	14.73
	SA $T = 5$	28034.43	307.42	1670.66	15.77
	SA $T = 10$	28230.94	268.04	1669.04	22.27
	SA $T = 20$	28402.98	211.79	1735.44	16.25
	Hill climbing	27659.40	228.98	1708.40	11.77
Instance 6	Construction heuristic	76787.06	342.21	4319.37	23.76
	SA $T = 0.5$	32706.85	379.68	2308.99	24.52
	SA $T = 1$	31310.53	338.70	2212.53	24.97
	SA $T = 5$	32348.37	389.96	2210.53	24.67
	SA $T = 10$	32264.98	427.64	2252.43	28.20
	Hill climbing	32320.89	366.99	2257.7	35.54
Instance 7	Construction heuristic	135852.86	480.06	6647.58	25.43
	SA $T = 1$	52759.67	466.08	3766.58	27.80
	SA $T = 5$	52839.51	399.91	3585.32	21.87
	SA $T = 10$	52527.78	309.55	3642.91	17.78
	SA $T = 20$	54050.58	342.35	4000.37	28.48
	Hill climbing	52918.33	383.51	3801.19	27.76
Instance 8	Construction heuristic	15695.73	203.99	1027.85	14.93
	SA $T = 1$	6953.34	158.74	477.82	13.90
	SA $T = 5$	6751.71	123.26	456.56	14.48
	SA $T = 10$	7038.60	137.84	479.45	14.38
	SA $T = 20$	7180.64	143.61	491.68	14.96
	Hill climbing	6832.12	146.64	511.99	11.82
Instance 9	Construction heuristic	82628.66	413.85	5239.20	28.13
	SA $T = 1$	31162.35	272.02	3025.29	25.11
	SA $T = 5$	30947.47	283.55	3004.49	17.80
	SA $T = 10$	31927.78	275.92	3062.31	23.69
	SA $T = 20$	32913.02	375.69	3243.34	19.80
	Hill climbing	31400.26	247.82	3029.21	26.52
Instance 10	Construction heuristic	53600.38	344.68	2690.08	26.69
	SA $T = 1$	25587.85	224.02	1727.23	24.30
	SA $T = 5$	25426.45	203.21	1688.36	27.19
	SA $T = 10$	25443.54	247.17	1756.32	26.46
	SA $T = 20$	26263.30	251.38	1820.77	23.66
	Hill climbing	25992.72	230.13	1774.82	25.17

day shift immediately following a night shift, removing the optimiser's incentive to make such an allocation. We would recommend that others studying the problem adopt this as a standard constraint (Table 3).

8 Conclusion and Future Work

The 2016 NHS Junior Doctor Contract changes significantly the way in which UK hospitals must approach staff rostering. The new scheduling constraints significantly affect the shape and complexity of the solution space, presenting a challenging optimisation problem.

Our discussions with real-world hospitals have emphasized that there is no single objective measure of roster quality for the problem, with hospitals willing to accept a higher guardian fine in pursuit of fewer overall constraint violations in some cases, and *vice versa*. This means that real-world systems would need to present a selection of potential rosters to administrators for consideration. For researchers, this also means that the evaluation of automated rostering approaches must consider the effectiveness in improving solutions by both metrics.

We have presented a benchmark problem set, that can be used by researchers to compare the effectiveness of various optimisation techniques on standard problems using the new constraints and the metrics from the contract.

We have discussed the performance of two known baseline approaches (random search and hill climbing), and one optimisation metaheuristic (simulated annealing) on the benchmark problem sets, allowing the performance of other approaches to be more easily placed into context.

Our vision of future work includes the use of multi-objective optimisation methods to allow a degree of automated balancing between the two objective solution quality metrics during optimisation. We are also interested in automated rostering during the transition period between old and new contracts, where individual doctors may be subject to a vastly different system of constraints.

Finally, we are also interested in the way in which on-call periods are handled in the new junior doctor contract. Doctors who work some shifts on-call are subject to further constraints in their working pattern, some applying only if a doctor is actually called into work during the on-call period. This case is particularly interesting, as the information cannot possibly be known *a priori*, requiring rosters to be dynamically re-generated in response to real-world events. This would require consideration of a roster's resilience: the likelihood of a situation arising where a guardian fine or constraint violation is unavoidable should an on-call doctor be required. One final compounding factor would be that this dynamic re-generation of a roster may well need to be completed at short notice.

Acknowledgment. This work is supported by the Knowledge Transfer Partnerships scheme (project No.10256). Authors thank Andrew Tapp (Shrewsbury and Telford Hospital NHS Trust) for advising on the problem and providing the example.

References

1. British Medical Association: Doctors' titles: explained. https://www.bma.org.uk/-/media/files/pdfs/about%20the%20bma/how%20we%20work/professional%20committees/patient%20liaison%20group/plg-doctors-titles-explained.pdf ?la=en. Accessed 28 June 2017
2. NHS Employers: Terms and Conditions of Service for NHS Doctors and Dentists in Training (England) 2016. Version 2. 30 March 2017. www.nhsemployers.org/~/media/Employers/Documents/Need%20to%20know/Terms%20and%20Conditions%20of%20Service%20for%20NHS%20Doctors%20and%20Dentists%20in%20Training%20England%202016%20Version%202%20%2030%20March%202017.pdf. Accessed 19 June 2017
3. NHS Employers: Pay banding criteria. http://webarchive.nationalarchives.gov.uk/20130107105354/www.dh.gov.uk/prod consum dh/groups/dh digitalassets/@dh/@en/documents/digitalasset/dh 4053877.pdf. Accessed 28 June 2017
4. Haspeslagh, S., De Causmaecker, P., Schaerf, A., Stølevik, M.: The first international nurse rostering competition 2010. Ann. Oper. Res. **218**(1), 221–236 (2014)
5. Burke, E.K., De Causmaecker, P., Berghe, G.V., Van Landeghem, H.: The state of the art of nurse rostering. J. Sched. **7**(6), 441–499 (2004)
6. Ernst, A.T., Jiang, H., Krishnamoorthy, M., Sier, D.: Staff scheduling and rostering: a review of applications, methods and models. Eur. J. Oper. Res. **153**(1), 3–27 (2004)
7. De Causmaecker, P.: Nurse rostering competition. https://www.kuleuven-kulak.be/nrpcompetition. Accessed 24 July 2017
8. Van den Bergh, J., Belien, J., De Bruecker, P., Demeulemeester, E., De Boeck, L.: Personnel scheduling: a literature review. Eur. J. Oper. Res. **226**(3), 367–385 (2013)
9. Bruni, R., Detti, P.: A flexible discrete optimization approach to the physician scheduling problem. Oper. Rese. Health Care **3**(4), 191–199 (2014)
10. Van Huele, C., Vanhoucke, M.: Analysis of the integration of the physician rostering problem and the surgery scheduling problem. J. Med. Syst. **38**(6), 43 (2014)
11. Adams, T., O'Sullivan, M., Christiansen, J., Muir, P., Walker, C.: Rostering general medicine physicians to balance workload across inpatient wards: a case study. BMJ Innovations **3**(2), 84–90 (2017). bmjinnov-2016 AQ4
12. Frey, L., Hanne, T., Dornberger, R.: Optimizing staff rosters for emergency shifts for doctors. In: IEEE Congress on Evolutionary Computation 2009, CEC 2009, pp. 2540–2546. IEEE (2009)
13. Department of Health, The Rt Hon Jeremy Hunt MP: Junior doctors contract agreement. https://www.gov.uk/government/speeches/junior-doctors-contract-agreement. Accessed 19 June 2017
14. BMA, NHS Employers and Department of Health: Junior doctors contract agreement. http://www.acas.org.uk/media/pdf/g/6/Junior-doctors-contract-agreement-18-May-2016.pdf. Accessed 19 June 2017
15. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., et al.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)