Journal of the Operational Research Society (2000) 51, 825–833 0160-5682/00 $15.00
www.stockton-press.co.uk/jors

Solving a nurse scheduling problem with knapsacks, networks and tabu search

KA Dowsland[1] and JM Thompson[2]*
[1]*Swansea University, Swansea and* [2]*Cardiff University, Cardiff*

This paper illustrates how a modern heuristic and two classical integer programming models have been combined to provide a solution to a nurse rostering problem at a major UK hospital. Neither a heuristic nor an exact approach based on a standard IP package was able to meet all the practical requirements. This was overcome by using a variant of tabu search as the core method, but applying knapsack and network flow models in pre- and post-processing phases. The result is a successful software tool that frees senior nursing staff from a time consuming administrative task.

Keywords: manpower scheduling; integer programming; tabu search; health

Introduction

Many manpower scheduling problems have natural models as integer linear programs. However, the numbers of variables and constraints involved may be very large and until recently this has precluded effective solution using off-the-shelf IP packages. Recent advances have extended the size and scope of problems amenable to solution in this way, so that today an IP approach is a sensible choice in many practical situations.[1,2] At the same time there has been considerable success in the use of meta-heuristic approaches such as simulated annealing, tabu search and genetic algorithms.[3–5] While IP approaches have the obvious advantage of guaranteed optimality without the expense of algorithm development, advocates of heuristic approaches point to their flexibility and the fact that they do not rely on sophisticated software packages. Both approaches also have their disadvantages. Advanced IP packages can be expensive and memory intensive, and solution times may vary considerably over different problem instances of a similar size. On the other hand heuristics may not give solutions of consistent quality, are often criticised for being slow, and may have difficulty in converging to good feasible solutions when applied to highly constrained problems. Therefore it is not surprising that there are many real-life problems where the combination of problem complexity and the operating environment means that neither approach is able to meet all the requirements.

This paper deals with a nurse rostering problem at a major UK hospital and shows how the disadvantages of a heuristic approach have been overcome by combining it with two classical integer programming models. The result is the Computer Aided Rostering Environment, (CARE), a successful software tool that is able to meet all the hospital's requirements and free senior nursing staff from time-consuming administrative tasks. A detailed description of the underlying tabu search algorithm has been given elsewhere.[6] Here we summarise the relevant details, and focus on the knapsack and network flow models used for pre- and post-processing. We conclude with comments on the robustness of the full system in the face of changes in problem specification.

*Correspondence: Dr JM Thompson, Cardiff University, Senghennydd Road, PO Box 926, Cardiff CF2 4YH, UK.
E-mail: ThompsonJM1@cardiff.ac.uk

Problem details

The problem is that of producing weekly rosters for wards of between 20 and 30 nurses. The rosters must ensure that there are sufficient nurses on duty at all times. They must also take account of financial considerations and the rules and guidelines on working practices, as well as attempting to meet the preferences and requests of each individual nurse. The days are divided into three shifts; two day shifts known as earlies and lates; and a longer night shift. In a given week a nurse will either work days or nights. Because of the differences in shift length, the number of day shifts comprising a full week's work is typically one more than the number of night shifts. A full time workload would be five days or four nights, while four days or three nights, and three days or two nights, are also popular options. However, there are also a number of nurses contracted to work three days or three nights, and other mixes occur from time to time as a result of sickness, annual leave or training.

There are three grades of nurse, grades 1, 2 and 3, with grade 1 being the most senior in terms of their experience

and expertise. Each of the 21 weekly shifts has a minimum requirement for nurses qualified up to the level of each grade. These requirements are expressed cumulatively. For example, there may be a requirement for at least one nurse of grade 1, at least 2 nurses of grades 1 to 2, and at least 4 nurses in total. When producing a roster the primary objective is to meet these covering requirements while allocating each nurse to the correct number of feasible shifts. This involves avoiding pre-booked annual leave or shifts incompatible with the previous week's work, for example, working Saturday night at the end of one week and a day shift on Sunday at the beginning of the next.

In addition to the above, the schedule needs to take account of requests and preferences. These can be partitioned into three types: (1) those inherent in the pattern of work; (2) those arising as a result of a request for a particular day off; and (3) those dependent on the recent work history. Examples in the first category include a preference for a consecutive stretch of work rather than a work day straddled by days off and, for those days on, a preference for days off to be preceded by an early and followed by a late. The third category is concerned with fairness and includes the amount of weekend and night working, and the frequency of requests granted over the medium term.

At the start of the project, rosters were produced manually by the senior member of the nursing staff on each ward or group of wards, a task that took an experienced person several hours per month. Initially they were sceptical of our claims that a computer could match the quality of rosters they produced themselves, and were worried about losing control over the shifts worked by their staff. They also had only a very limited understanding of the sort of information that was important in the design and development of the underlying algorithm. Nevertheless they agreed to co-operate in providing information and data, and testing our system once we had shown it to be worthwhile, but no financial support was to be available for development work. As a result of these factors, and other information gleaned during initial consultations, it became clear that the system should exhibit the following features.

- Only produce rosters that meet the covering requirements at each grade;
- Be capable of running on the computers currently in use on each ward. These were typically low specification machines;
- Not rely on any expensive software packages;
- Allow the user to specify some requests as being more important than others;
- Produce a number of different solutions of consistent quality for the user to select from;
- Run quickly enough to allow the user to adjust the input if required;
- Allow for changes to the problem specification.

In the following section we model the problem within an optimisation framework and discuss our choice of solution approach.

Formalising the problem and selecting a solution strategy

Our initial meetings convinced us that we could place numeric values on the various sources of undesirability for each work pattern with respect to each nurse, defining a hierarchy of relative importance for requests that would be under the control of the person responsible for producing the rosters. Requests were classed as *vital, very important, important* or *preferred*, with patterns violating a *vital* condition being disallowed and those violating the others being allocated a decreasing sequence of costs. Preferences inherent in the patterns of days and nights worked were judged to be around the same level of importance as the *preferred* category of requests, but we were told that those arising from the early and late allocations should take a lower priority. The individual p_{ij} values were calculated as the sum of a series of pre-defined weights, relating to the different sources and levels of each undesirable feature in using pattern j for nurse i. Initial values were estimated as a result of comments from the hospital. These were used in the development phase and were then tuned using an iterative process. Schedules for several weeks of old data were produced with the current set of weights and hospital staff were invited to point out any problems or anomalies. These comments were then used to adjust the weights and start a new cycle of adjustment. Once the weights had been finalised a further period of close monitoring on three different wards, covering different levels of staff absences and volume of requests, ensured that the values were indeed suitable.

The allocation of numeric values allows us to specify the problem within the framework of an optimisation model as follows: The problem can be considered as a multiple-choice covering problem, in which we have to choose a feasible set of shifts for each nurse in such a way as to adhere to the covering requirements, while minimising the preference costs. This formulation can be expressed as an integer linear program as follows:

$$\min z = \sum_{i=1}^{n} \sum_{j \in F(i)} p_{ij} x_{ij} \tag{1}$$

subject to

$$\sum_{j \in F(i)} x_{ij} = 1 \,\forall i \tag{2}$$

$$\sum_{i \in G_r} \sum_{j \in F(i)} a_{jk} x_{ij} \geqslant R(k, r) \forall r, k \tag{3}$$

$$x_{ij} = 0 \text{ or } 1 \tag{4}$$

where: $x_{ij} = 1$ if nurse i works pattern j, $= 0$ otherwise; p_{ij} is the penalty associated with nurse i working pattern j; $F(i)$ is the set of patterns feasible for nurse i; $a_{jk} = 1$ if pattern j covers shift k; G_r is the set of nurses of grades 1 to r; and

$R(k, r)$ is the minimum acceptable number of nurses of grades 1 to r for shift k.

In fact, for financial reasons the night-shift must not be manned by more than the total number of nurses required. Therefore constraints (3) for $r = 3$ and those values of k corresponding to the night shifts will be strict equalities.

Each nurse typically has several hundred feasible patterns. This formulation requires the enumeration and costing of them all and will result in up to 20 000 variables. We will see later that this can be reduced by a factor of around 10, yielding modestly sized problems of between 1500 and 2000 variables and 60–70 constraints. Nevertheless, the use of an IP package to solve the problem was ruled out due to the requirements outlined in the previous section. Even problems of this size are likely to require the features of a modern software package in order to solve them consistently and reliably. Both the financial situation and the specifications of some of the ward computers meant that this was not considered to be a valid option. Even without these constraints, other factors pointed to an alternative approach. Firstly the possibility of changes in specification could render an IP formulation more difficult, or introduce non-linear expressions into the formulation. Secondly solution speed may vary considerably from problem to problem and may not be reliably fast, especially if a number of different optimal solutions are required, (subsequent experiments with a professional IP package have confirmed this view) and finally, given that our weights were to be based on a series of subjective judgements there was no merit in pursuing an optimal solution at the expense of unacceptably long computation times.

The scheduling heuristic

Having opted for a heuristic solution our choice of a sequential local search approach was based on our own experiences and work reported in the literature. There are many examples of simulated annealing and tabu search producing high quality solutions for time-tabling and scheduling problems involving multiple objectives or mixtures of hard and soft constraints. These include the use of simulated annealing for examination scheduling and school timetabling,[7–9] and the use of tabu search in a variety of educational and sports scheduling problems.[10–13] Dowsland[6] deals with the development of a tabu search approach to a smaller version of the problem in which it is assumed that sufficient nurses are available, and no distinction is made between early and late shifts, this was the approach used here. As detailed in that paper our eventual decision to use tabu search as opposed to simulated annealing was the result of further investigation into the problem structure. A full description of the algorithm will not be given here. Instead we briefly outline the features that are relevant to the rest of the discussion.

As outlined by Dowsland there are a number of ways of modeling the problem within a local search framework. We chose a representation in line with the IP formulation The solution space is defined as the set of allocations of each nurse to a feasible shift pattern defining a week's work. Neighbourhood moves involve changing the pattern of a single nurse. Simply finding a feasible schedule is itself a difficult problem, and the covering constraints (3) were therefore relaxed in the solution space definition. Rather than penalise violations in a single evaluation function, we use strategic oscillation[14] to alternate between seeking out feasible solutions and reducing the penalty costs. The basic tabu search algorithm is further strengthened by the use of candidate list strategies, complex chains of moves, and tabu lists and diversification strategies based on problem specific knowledge. The result is an aggressive search that is able to home in on a large number of high quality local optima.

Nevertheless, some of the requirements of the system are tough goals for any heuristic. In particular there is a major difficulty relating to feasibility, and the requirement for solutions to be produced quickly compromises the goal of finding solutions of consistent quality in a large solution space. These will be dealt with in the following sections.

The problem of feasibility and the knapsack model

Are there sufficient nurses?

As stated above the covering constraints are non-negotiable. However, there are not always sufficient nurses available for work on a given ward in a given week. In this situation a ward utilises one or more bank nurses who are then 'costed' against the ward for the shifts worked there. Therefore a bank nurse should only be called upon if absolutely necessary. This situation causes problems for a heuristic approach as it is not possible to tell whether failure to find a feasible solution is because there is none, or because the search has not been successful. One way around this is to include bank nurses in the set of nurses available and to penalise their use in the evaluation function. However, this may result in unnecessary use of bank nurses. This can be avoided if the need for bank nurses are recognised, and the correct number of such nurses added to those already available, in a pre-processing phase. The difference between the numbers of shifts worked by the same nurse when on nights or days means that this cannot be determined by simply adding up the number of shifts available and comparing it with those needed. We need to solve the more complex problem of determining whether or not there is an allocation of nurses to days and nights such that the total day and night requirements at each grade are

met. If we ignore the different grades these requirements can be written

$$\sum_{i=1}^{T} e_i y_i \geqslant E \tag{5}$$

$$\sum_{i=1}^{T} d_i(N_i - y_i) \geqslant D \tag{6}$$

$$\sum_{i=1}^{T} y_i \leqslant N_i \tag{7}$$

$$y_i \text{ integer} \tag{8}$$

where $i = 1, \ldots, T$ define the different types of full time and part-time nurses according to the numbers of days and nights worked; y_i = the number of type i nurses on nights; d_i and e_i are the number of days or nights worked by nurses of type i; N_i is the number of nurses of type i; and D and E are the total day and night shifts required.

Rearranging (6) gives

$$\sum_{i=1}^{T} d_i y_i \leqslant \sum_{i=1}^{T} N_i d_i - D \tag{6'}$$

and there will be sufficient nurses if max $\sum_{i=1}^{T} e_i y_i$ is at least E subject to constraints (6'), (7) and (8).

This is a standard bounded knapsack problem and can be solved in negligible time using the method given in Martello and Toth.[15] This is a branch and bound algorithm, based on the well-known branch and bound method for integer programming. The structure of the problem means that the LP relaxations will never give rise to more than one fractional variable and can be solved trivially by sorting the variables into value/weight order. If at any stage the lower bound, given by rounding down the fractional variable, is greater than or equal to E the corresponding rostering problem is feasible. Similarly if the global upper bound falls below E then the problem is not feasible, therefore there is no need to search the whole tree.

If the problem is infeasible we need to determine the number of bank nurses required. This is achieved by adding a number of bank nurses equal to the difference between the global upper bound and E, and re-solving. The process is repeated until the number of additional nurses is sufficient for feasibility. In practice more than one pass is rarely required.

Dealing with grades

As there are separate covering constraints for the different grades we need to ensure that cover is adequate at all three levels. However, it is not sufficient to solve three separate problems, one for grade 1, one for grades 1 and 2, and one for all three, as the solutions may be incompatible with one another. This situation can be detected by executing a single branch and bound search in which the tree starts by considering only nurses and requirements of grade 1. Once a node corresponding to a feasible solution is reached, further branching involves fixing the numbers of grade 1 nurses on days/nights implied by this solution and expanding the tree to include nurses of grade 2, and finally incorporating grade 3 nurses. The disadvantage of this approach is that if the problem at higher grades is slack, then a feasible node for grade 1 may not define a unique feasible solution and further branching is required. This will obviously increase the size of the search, and as the constraints for higher grades tend to be slacker than those involving lower grades, many of the branches will fail to yield feasible solutions at all three levels.

We therefore use an alternative approach that starts by ignoring the grades. The tree relating to the corresponding knapsack problem is searched until a node yielding a solution of value at least E is found, we call this the overall solution. Such a solution indicates the numbers of nurses of each type who should work days or nights, but does not stipulate their grades. In order to see if there is a suitable allocation to grades we use this solution to define a knapsack problem for grades 1 and 2 that will ensure compatibility as follows.

Let D_g and E_g be the cumulative day and night requirements for grades 1 to g; N_{ig} the number of nurses of type i available of grade g; and Q_i the number of type i nurses allocated to nights in the overall solution and e_i and d_i as above.

We will use z to denote the target value for the knapsack objective and U_i to represent the upper bound on the number of nurses of type i available. At the start we set $z = E_2$. If

$$Q_i \leqslant N_{i3}, \quad \text{set} \quad U_i = N_{i1} + N_{i2} \tag{A}$$

otherwise set

$$U_i = N_{i1} + N_{i2} - (Q_i - N_{i3}) \quad \text{and} \quad z = E_2 - (Q_i - N_{i3}) \tag{B}$$

If

$$Q_i < U_i \quad \text{set} \quad U_i = Q_i \tag{C}$$

In the case of (A) there are sufficient nurses of type i and grade 3 to cover all the night allocation for this type. If necessary, we could therefore allocate all grade 1 or 2 nurses of this type to days. Therefore there is no lower bound on the numbers of grade 1 or 2 nurses of type i working nights. Case (B) applies to the opposite situation in which there are not sufficient grade 3 nurses to provide all the night-workers of type i. A minimum number of higher grade nurses must therefore be fixed on nights, and this is modeled by removing these nurses from the problem and reducing the night requirements accordingly. Case (C) applies if the total number of nurses of type i on nights is less than the total available in grades 1 and 2. In this case the upper bound is adjusted, effectively forcing some of these nurses onto days. If the solution to the resulting knapsack is at least z, then

there is a compatible grade 1 and 2 solution. Similar calculations can then be used to determine new upper and lower bounds to check for the existence of a compatible grade 1 solution. If the knapsack shows that there is no compatible solution, the search at the previous level is continued until another feasible solution is determined. The constraint on the exact covering of the night shifts when $g = 3$ implies that each node represents a unique feasible solution, and as the higher grade problems tend to be slack, a compatible solution is usually found for the first feasible node. Note that this means that an objective value $> E_3$ is not acceptable, and the search must be modified to accept only solutions where the objective value is equal to E_3. (If the knapsack calculations indicate that there are sufficient nurses but equality is not possible, then the operator is informed and the availability of one of the nurses is adjusted on a one-off basis.)

A change in specification

On weeks when bank nurses are not required, there may be enough manpower to provide extra cover. We were originally told that there was no requirement to spread this evenly, however, this was not quite true. The early shifts are the busiest of the day, and the minimum required level of cover is usually one nurse short of the preferred level. In this case, any spare cover after the minimum requirements have been met should be used to reach the preferred cover on as many days as possible. This could be dealt with in the tabu search phase by adding an extra penalty to days whose cover fails below the preferred level. However, it can be dealt with more effectively using the knapsack routine. If the knapsack calculations show that there are more than enough nurses to meet the required cover then the right-hand side of the knapsack constraint (6′) is reduced by 7 units, and the resulting shortfall in cover is calculated as for the bank nurses. This is equivalent to increasing the required cover on each day. Instead of meeting this shortfall by a number of bank nurses, a single dummy nurse, working any pattern made up of the required number of earlies is added. As this nurse can cover at most one shift on any day this ensures that real nurses meet the required level on days 'worked' by the dummy nurse, and the preferred level on all other days. Because the knapsack ensures that the number of dummy shifts is minimised, the maximum number of days are covered at the preferred level. On some wards the preferred cover only applies on weekdays. In this case the feasible shifts for the dummy nurse are restricted to those that include both Saturday and Sunday.

The knapsack procedure is run as the first phase of the solution process. If there are insufficient nurses at any grade, the operator is given the choice of reducing the requirements or adding the required number of bank nurses. If there are more nurses than required a dummy nurse is added and the requirements adjusted accordingly. The advantages of this pre-processing phase are:

- The problem tackled in the subsequent tabu search stage is known to be feasible;
- All feasible solutions implicitly satisfy the objective of maximising the number of days meeting the preferred cover;
- The user is provided with information concerning the level of cover and the need for bank nurses at an early stage in the solution process.

However, the problems passed on to the tabu search stage are still large and exhibit features that need special handling within a local search framework. In the following section we show how these problems were overcome using a post-processing phase based on a network flow model.

The network flow model

A basic model

The remaining problems relate to the allocation of earlies and lates. Sources of preference cost resulting from this aspect of the roster can be summarised as follows:

- Rather than request a day off nurses may ask not to work an early or late on a particular day;
- There is a preference for an early to precede a day off and a late to follow it;
- For nurses working 4 or 5 days there is a preference for earlies and lates to be split evenly.

Apart from the first of these, which usually corresponds to a request with the *preferred* rating, these preferences are considerably less important than any of the other criteria, and are allocated costs that reflect this. Therefore the quality of a solution depends largely on the days and nights worked by each nurse, with the early and late allocation of those on days making only a marginal contribution. If the whole solution space is included in the search and all moves are treated in the same way, time will be wasted evaluating moves that are effectively fine-tuning the current solution. This could be overcome with a candidate list strategy in which only moves involving a change of days or nights would be considered, until no further improvement with such moves are possible. Other moves could then be allowed in order to find the best allocation of earlies and lates. However, once the days and nights worked by each nurse are known there is no need to use a heuristic search to find the best allocation of earlies and lates, as this problem can be modeled as that of finding a minimum cost flow in a closed network.

In order to simplify the discussion we start by ignoring the difference in grades and treat the preference for an even balance of earlies and lates as a binding constraint. This implies that all nurses have upper and lower bounds on the

number of early shifts worked and there is no cost associated with the number of earlies worked within these bounds.

Let *Days* be the set of nurses working days; $x_{ij} = 1$ if nurse i works an early on day j, $= 0$ if working a late; a_i and b_i the upper and lower bounds on the number of early shifts for nurse i (for nurses working less than four days a_i = number of days worked and $b_i = 0$); Re_j the number of nurses required on earlies on day j; Rl_j the number of nurses required on lates on day j; $W(i)$ the set of days worked by nurse i; $v(j)$ the number of nurses working day j; and c_{ij} = penalty cost if lates preferred for nurse i on day j, $=$ − penalty cost if earlies preferred, $= 0$ if no preference.

The problem can be formulated as follows:

$$\min \sum_{i \in \text{Days}} \sum_{j \in W(i)} c_{ij} x_{ij} \tag{9}$$

subject to

$$\sum_{j \in W(i)} x_{ij} \leqslant a_i \ \forall i \tag{10}$$

$$\sum_{j \in W(i)} x_{ij} \geqslant b_i \ \forall i \tag{11}$$

$$\sum_{i \in \text{Days}} x_{ij} \geqslant Re_j \ \forall j \tag{12}$$

$$\sum_{\substack{i \in \text{Days} \\ j \in W(i)}} (1 - x_{ij}) \geqslant Rl_j \ \forall j \tag{13}$$

$$x_{ij} = 0 \text{ or } 1 \ \forall i, j \tag{14}$$

Constraints (13) can be re-arranged to give:

$$\sum_{\substack{i \in \text{Days} \\ j \in W(i)}} x_{ij} \leqslant v(j) - Rl_j \ \forall j \tag{13'}$$

This is a transportation problem with added constraints of lower bounds on the supplies, upper bounds on the demands, and an upper bound on the amount supplied from i to j. These changes are easily incorporated into the standard network flow representation of the transportation problem, (see for example Smith[16]). Using source node S, sink node T, nurse nodes, i, for all $i \in$ Days, and day nodes $j = 1, 7$, the arcs and their lower and upper bounds and costs are as defined in Table 1.

Because it is convenient for a perfect solution to correspond to a zero cost solution we dispense with the negative costs that result when earlies are preferred as follows: For all arcs with $c_{ij} < 0$ we increase the lower bound to 1 and reduce the cost to 0. We then add a reverse arc (j, i) with lower bound = 0, upper bound = 1, and cost = c'_{ij}, where $c'_{ij} = -c_{ij}$. Therefore if nurse i works lates on day j the unit of flow that is now forced through (i, j) must be cancelled by flow through (j, i), therefore incurring a positive cost.

Table 1 The basic network flow model

Arcs	*Lower bound*	*Upper bound*	*Cost*
(S, i) $\forall i \in$ Days	b_i	a_i	0
(i, j) $\forall i \in$ Days, $\forall j \in W(i)$	0	1	c_{ij}
(j, T) $\forall j$	Re_j	$v(j) - Rl_j$	0
(T, S)	$\sum_{j=1}^{7} Re_i$	$\sum_{j=1}^{7} v(j) - Rl_j$	0

Generalising the model

We now need to adapt the model to allow for the restriction on the numbers of earlies worked by those working more than three days to be relaxed, and to include the cumulative constraints for the grades. The former is achieved by adding two additional arcs between nodes S and i. A forward arc (S, i) allows nurse i to work additional early shifts, beyond the upper bound. In order to achieve the desired effect this arc must have a zero lower bound and an upper bound given by $|W(i)| - a_i$. A backward arc (i, S) allows the nurse to work fewer early shifts than defined by the lower bound by carrying flow back from i to S, cancelling the corresponding flow from S to i. This arc has zero lower bound, and upper bound given by b_i. Both arcs have a cost equal to the penalty associated by failure to meet constraints (10) or (11).

The grades are incorporated by splitting the single day node, j, into three separate nodes j_1, j_2, and j_3, one for each grade. Arcs from i to j are replaced by arcs from i to $j_{g(i)}$, where $g(i)$ denotes the grade of nurse i. Additional arcs from j_1 to j_2 and j_2 to j_3 are included and the original arc from j to T is replaced by an arc from j_3 to T. Total flow into j_1 defines the number of grade 1 nurses on earlies on day j. Therefore arc (j_1, j_2) can be used to impose the early and late requirements at grade 1. Upper and lower bounds on this arc are defined in the same way as those on (j, T), but the calculations are limited to the grade 1 nurses and requirements. Similarly the flow into j_2 is the sum of the grade 1 and 2 nurses working earlies, and bounds on arc (j_2, j_3) impose the early and late requirements using similar formulae applied to these nurses and requirements.

The resulting network is illustrated in Figure 1. The labels on each arc represent (lower bound, upper bound, cost). All penalties for violated constraints are denoted by c. In practice different values of c would be used to reflect the importance of each constraint. For clarity, only part of the network is shown. Nurse i represents a full time nurse of grade 1 who has requested to work an early shift on day j and a late shift on day l. The nodes j_1, j_2, and j_3 impose the

Table 2 Availabilities and requirements for day j

	Grade 1	*Grades 1 and 2*	*Total*
Early requirement	1	2	4
Late requirement	1	2	4
Total available	2	5	9

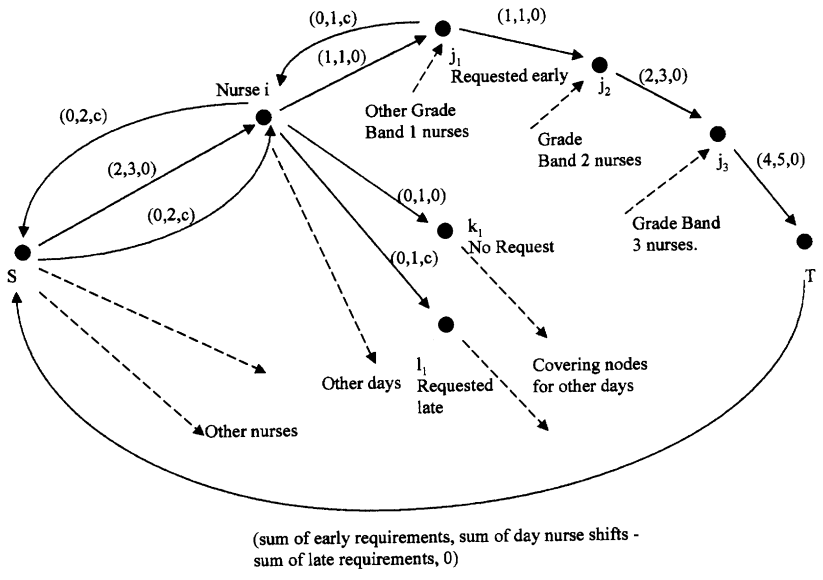

Figure 1 The network flow model.

covering constraints on day j assuming the availabilities and requirements as given in Table 2.

In practice, a typical network consists of less than 50 nodes and 100 to 150 arcs, most of which have only small gaps between their upper and lower bounds. It is therefore solved very quickly using an implementation of the standard out-of-kilter algorithm, based on the code given in Smith,[16] starting from an initial solution in which all flows and multipliers are set to zero.

Incorporating the model into the solution process

The ease with which optimal early and late allocations can be found, coupled with the fact that the day night allocation is the major factor in determining solution quality, allows a considerable reduction in the size of the solution space. This is achieved by amalgamating the early and late requirements into a single requirement for each day, therefore reducing the number of patterns considerably, and removing the early/late costs from the evaluation function. High quality solutions found by the search can then be run through the network flow model in order to determine the best allocation of earlies and lates. It remains to decide how such solutions are identified.

As the costs involved in the network flow problem are typically an order of magnitude lower than the other costs, the application of the network flow optimisation stage will not change the ranking of solutions by more than a few places. The aggressiveness of the tabu search phase means that it typically visits a large number of high quality local optima. Therefore it is not necessary to incorporate the network flow calculations into the search. Instead we can store the k best solutions, for an appropriate choice of k, and evaluate these in a post-processing phase. As we produce five tabu search solutions for the user to select from, and empirical evidence suggested that low cost early/late allocations were usually achieved, we have found it sufficient to use a value of $k = 1$.

The main benefit of allocating the earlies and lates in a post-processing phase is a dramatic reduction in the size of the solution space. For example the number of feasible patterns for a full time nurse is reduced from 707 to 56, and those for a nurse working four days or three nights from 595 to 70. This means that the search can cover the solution space effectively within a reasonable amount of time and produce solutions of consistent quality.

The full system

The whole solution procedure can now be summarised as follows:

- Convert the raw data on requests, history etc. into a set of numerical penalty values;
- Run the problem through the knapsack model adding bank nurses or dummy nurses as necessary;
- Use the tabu search algorithm to find a set of shift patterns in terms of the days and nights worked;
- Use the network flow algorithm to allocate the day shifts to earlies and lates.

The real test of the system is the reaction of the hospital, where staff agreed that it was able to produce schedules at least as good as those produced by manual methods. However, from an academic point of view it is also interesting to see how good the solutions are as compared with the optimal solution to the IP formulation. Fifty-one data sets originating from three different wards and covering different periods of the year were used as test problems and

the results of the above system, using 10 different random starts and random number strings were compared with the results of replacing the tabu search phase by an IP solution. (The latter required an overnight run using a modern IP package on a pentium II pc in order to solve several of the problems), these results are reported in Table 3. They show that the above approach was able to find an optimal solution for all 10 starts on many of the data sets. Even on the hardest data sets the optimum was reached in 7 runs out of 10, and the mean cost column indicates that the discrepancy in the remaining cases was small. Each tabu search run takes just a few seconds on the same machine and less than one minute on the low specification machines available at the hospital.

Tests of robustness

As a result of the underlying structure of the knapsack and network models, and the problem specific information incorporated into the tabu search, the robustness of the overall package must be brought into question. During the development and testing stages four changes in problem specification occurred. That of maximising the number of shifts covered to the preferred level has already been discussed. In this case the use of the knapsack pre-processing phase allowed us to incorporate the change in such a way as to make the extra objective implicit in the problem formulation. The other changes were;

- allowing some (prespecified) nurses to work a mixture of days and nights;
- ensuring that no more than one highly qualified nurse was on duty on a Saturday or Sunday;
- the introduction of a team covering strategy.

The first two required only minor adjustments to one or more stages, but the final change was potentially more awkward. It involves allocating each nurse and each patient to a team. At least one nurse from each team must be on duty at all times, so that patients will see a familiar face on every shift. We had anticipated that we would need to make significant changes as the chain moves used in the tabu search stage are incompatible with the team constraint. However, the balance between the aggressive pursuit of local optima and the diversification into new areas of the search space meant that all that was required was an additional term in the evaluation function to nudge the search in the right direction. The chain moves can cause violations of the team constraints and some of the local optima visited were therefore infeasible in this respect. Empirical evidence showed that feasible solutions of equivalent quality were usually visited shortly afterwards. Where this was not the case, comparison with exact solutions obtained from an IP model confirmed that the search was consistently converging to optimal solutions. Further difficulties arise from the fact that the team constraint cannot be incorporated into the network flow model. Once again we found that the best policy was to solve the relaxed problem. If the constraint was violated in the optimal solution a simple exchange procedure was used to correct this. The resulting increase in cost was frequently equal to zero, and was always very small.

Conclusions

We have shown how a real-life scheduling problem has been solved using a combination of classical IP algorithms and a tabu search heuristic. The structure of the knapsack

Table 3 A comparison of tabu search and IP results for 51 data sets

Prob no.	*Tabu best*	*No. best*	*Mean cost*	*IP optimal*	*Prob no.*	*Tabu best*	*No. best*	*Mean cost*	*IP optimal*	*Prob no.*	*Tabu best*	*No. best*	*Mean cost*	*IP optimal*
1	8	10	8.0	8	18	18	7	19.6	18	35	35	10	35.0	35
2	49	9	49.1	49	19	1	10	1.0	1	36	32	10	32.0	32
3	50	10	50.0	50	20	7	10	7.0	7	37	5	10	5.0	5
4	17	10	17.0	17	21	0	10	0.0	0	38	13	10	13.0	13
5	11	10	11.0	11	22	25	10	25.0	25	39	5	10	5.0	5
6	2	10	2.0	2	23	0	10	0.0	0	40	7	10	7.0	7
7	11	10	11.0	11	24	1	10	1.0	1	41	54	10	54.0	54
8	14	10	14.0	14	25	0	10	0.0	0	42	38	7	38.3	38
9	3	10	3.0	3	26	48	10	48.0	48	43	22	7	23.2	22
10	2	10	2.0	2	27	2	10	2.0	2	44	19	9	19.1	19
11	2	10	2.0	2	28	63	10	63.0	63	45	3	10	3.0	3
12	2	10	2.0	2	29	15	10	15.0	15	46	3	9	3.1	3
13	2	10	2.0	2	30	35	10	35.0	35	47	3	10	3.0	3
14	3	10	3.0	3	31	62	9	62.1	62	48	4	10	4.0	4
15	3	10	3.0	3	32	40	10	40.0	40	49	27	6	27.4	27
16	37	7	37.3	37	33	10	10	10.0	10	50	107	9	107.1	107
17	9	7	9.3	9	34	38	10	38.0	38	51	74	10	74.0	74

Tabu best is the best solution found in 10 runs; no. best is the number of times the best solution was obtained; mean cost is the mean over 10 runs; IP optimal is the cost obtained from the IP formulation.

and network models means that they can be solved efficiently without the need for LP or IP software. As well as providing almost instantaneous information on the level of cover available, the knapsack model ensures that the search does not waste time attempting to solve an infeasible problem, or allocate bank nurses when they are not required. It also allows the objective of maximising the days covered to the preferred level to be included implicitly. The network flow model takes care of the fine-tuning resulting from the allocation of earlies and lates, leaving the tabu search to focus on the more complex problem of allocating each nurse to a pattern of days or nights.

The algorithm has been incorporated into a full scheduling system, CARE, that is able to produce results at least as good as those of a manual scheduler, and has not resulted in any complaints of unfair treatment by individual nurses. Indeed the majority of nurses have welcomed the flexibility shown by CARE, for example in allowing each nurse to select their preferred shift patterns. Senior staff are delighted with the system, it enables them to maintain control over the schedules by grading the requests as they are input into the system, being consulted about the addition of bank nurses, and being presented with several different schedules (with a brief report on the shortcomings of each) from which they make the final selection. Ongoing experiments using data provided by a further group of hospitals have indicated that CARE is sufficiently robust to solve problems with different characteristics, and up to at least 50 nurses. This robustness is due to the ability of the tabu search phase to seek out a variety of high quality local optima. This, together with the fast processing times achieved, despite the limitations of the equipment and software available, is due to the combination of exact and heuristic approaches. We are currently having discussions with other interested UK hospitals concerning improvements in the user interface that will transform CARE into a commercially available software package.

References

1 Aykin T (1998). A composite branch and cut algorithm for optimal shift scheduling with multiple breaks and break windows. *J Opl Res Soc* **46**: 698–707.

2 Mitra G, Darby-Dowman K, Lucas C and Smith JW (1995). Maritime scheduling and artificial intelligence techniques. In: Sciomachen A (ed). *Optimization in Industry 3*. Wiley: Chichester, pp 1–17.

3 Easton F and Mansour N (1993). A distributed genetic algorithm for employee staffing and scheduling problems. In: Schaffer JD (ed). *Proceedings of the Fifth International Conference on Genetic Algorithms*. Morgan Kaufmann: Illinois, pp 360–367.

4 Brusco MJ and Jacobs LW (1993). A simulated annealing approach to the solution of flexible labour scheduling problems. *J Opl Res Soc* **44**: 1191–1200.

5 Ferland JA (1998). Generalised assignment type problems: A powerful modelling scheme. In: Burke E and Carter M (eds). *Practice and Theory of Automated Timetabling II, Lecture Notes in Computer Science* **1408**, Springer Verlag: Berlin, pp 53–77.

6 Dowsland KA (1998). Nurse scheduling with tabu search and strategic oscillation. *Eur J Opl Res* **106**: 393–407.

7 Thompson JM and Dowsland KA (1996). Variants of simulated annealing for the examination timetabling problem. *Annals of Opns Res* **63**: 105–128.

8 Thompson JM and Dowsland KA (1998). A robust simulated annealing based examination timetabling system. *Comps & Opns Res* **25**: 637–648.

9 Dige P, Lund C and Ravn HF (1993). Timetabling by simulated annealing. In: Vidal RVV (ed). *Applied Simulated Annealing—Lecture Notes in Economics and Mathematical Systems*, **396**, Springer Verlag: Berlin, pp 151–174.

10 Hertz A (1991). Tabu search for large scale timetabling problems. *Eur J Opl Res* **54**: 39–47.

11 Wright M (1996). School timetabling using heuristic search. *J Opl Res Soc* **47**: 347–357.

12 Wright M (1994). Timetabling county cricket fixtures using a form of tabu search. *J Opl Res Soc* **45**: 758–770.

13 Costa D (1995). An evolutionary tabu search algorithm and the NHL scheduling problem *INFOR* **33**: 161–178.

14 Glover F and Laguna M (1997). *Tabu Search*. Kluwer: Boston.

15 Martello S and Toth P (1990). *Knapsack Problems*. Wiley: Chichester.

16 Smith DK (1982). *Network Optimisation Practice*. Ellis Horwood: Chichester.

Received June 1999;
accepted February 2000 after two revisions