Nurse Rostering and Integer Programming Revisited

John Thornton
School of Information Technology, Faculty of Engineering and Applied Science,
Griffith University Gold Coast, Southport, Qld. 4217, email : easjohnt@cit.gu.edu.au

Abdul Sattar
School of Computing and Information Technology, Faculty of Science and Technology,
Griffith University, Nathan, Qld. 4111, email : sattar@cit.gu.edu.au

ABSTRACT: Design and development of robust and reliable *scheduling algorithms* has been an active research area in Computer Science and Artificial Intelligence. Given that the general problem is computationally intractable, many heuristic-based techniques have been developed, while other approaches have used optimising techniques for specific and limited problem domains. In this paper, we consider a large, real world scheduling problem of nurse rostering at the Gold Coast Hospital, and we propose an optimising integer programming-based approach to solving the problem. The study extends previous work in the area by looking at a more complex problem domain and by introducing a mathematical model that is capable of capturing the important details of the domain without becoming unrealistically large. We also describe a problem decomposition heuristic to effectively manage the computational resources. Schedule quality and staff allocation quality measures are introduced to evaluate the automated nurse schedules. Finally, we conclude that an integer programming-based approach to nurse rostering is feasible for realistic problem sizes and is sufficiently flexible to handle overconstrained problems with competing goals.

Keywords: Scheduling, Nurse Rostering, Integer Programming.

1. Introduction

Design and development of robust and reliable *scheduling algorithms* has been an active research area in Computer Science and Artificial Intelligence [1]. Given that the general problem is computationally intractable [2], many heuristic-based techniques have been developed, while other approaches have used optimising techniques for specific and limited problem domains. In this paper, we consider a large, real world scheduling problem of nurse rostering at the Gold Coast Hospital.

Nurse rostering can be considered as a *partial constraint satisfaction problem* (PCSP) [3]. The task is to find a consistent allocation of shift values, for a group of nurses, over a fixed period of time, that satisfy *as many as possible* of a set of rostering constraints. There are two basic types of constraint: (i) *schedule constraints* defining acceptable shift combinations and (ii) *staff constraints* defining acceptable overall staffing levels. Some constraints are *hard* (ie must be satisfied) while others are *soft* (ie may be violated). Of the soft constraints some are more important than others. The objective of nurse rostering is to find a roster that satisfies all hard constraints and *minimises* the degree to which the soft constraints are violated.

The allocation of nursing staff is a critical task in hospital management. Typically, nursing salaries form the largest item on the hospital budget [4]. Conversely, the number and skill level of nurses are a primary determinant of the safety and quality of patient care. The conflict between providing adequate care and minimising costs means the allocation of the right number and skill mix of staff to each shift becomes crucial. In addition, nursing personnel are a scarce resource [5]. Nurse rostering policies can have a direct impact on nurse satisfaction and hence on turnover [6]. Rosters requiring nurses to work difficult and tiring combinations of shifts can again impact on the quality and safety of patient care. Hospital management is therefore further concerned with providing rosters that minimise nurse dissatisfaction.

Previous attempts to solve nurse rostering problems were focused on *optimising integer programming solutions* [7, 8, 9, 10] and developing *non-optimal heuristic solutions* [11, 12, 13]. More recent attention has been placed on a *constraint programming* approach [14, 15]. Concerns have been raised in the literature that integer programming lacks flexibility and is unable to reliably solve larger problems [11, 15, 16]. In the integer programming research relatively small or simplified rostering problems have been considered. Larger and more realistic rostering problems have been solved using non-optimal heuristic techniques. These include the use of a cyclic descent algorithm [13, 17, 18], the modeling of human expert knowledge [19] and the use of mixed integer and heuristic techniques [20, 21].

The missing link in the literature has been the application of an optimising integer programming approach to a large and realistic rostering problem. This paper addresses this omission, and seeks to answer the following questions:

i) Is integer programming a reliable and efficient method for solving the particular problem considered in the research?

ii) Can a rostering system be developed, based on integer programming, that is sufficiently flexible to handle the day to day rostering demands of the hospital considered in the study?

We use an integer programming (IP) [22] approach to develop a flexible and reliable nurse rostering system for a reasonably large real world problem. The *hard* problem constraints are formulated as a set of simultaneous equations using 0-1 decision variables (eg if a nurse is to work a particular shift in the roster, then a corresponding decision variable will equal 1, otherwise it will be 0). Constraints containing inequalities are transformed into equalities by adding dummy or slack variables. An IP algorithm then attempts to find the maximum or minimum value of an objective function, whilst satisfying all hard constraints. The objective function gives a quantitative measure of how well the *soft* constraints in the model are satisfied. Goal programming is a form of integer and linear programming where more than one objective (or goal) is optimised in the objective function. In the case of the rostering problem, either a single or dual objective function can be defined. The first objective is to minimise nurse dissatisfaction with their schedules, and the second (optional) objective is to minimise deviations of staffing levels from *desired* staffing levels. By relaxing the integer constraints on the variables, an initial non-integral (and optimal) solution to the problem can be found using a *linear* programming algorithm. By adding further constraints to the model and repeatedly solving, an IP algorithm will eventually find an optimal and fully integral solution (if one exists). As would be expected, integer programming is NP-hard [23].

The remainder of the paper details the development of an integer programming system capable of solving the rostering problem considered in the study. Firstly, the problem is described in more detail with reference to relevant research in the area. Then the overall solution strategy and the mathematical model are presented. In the results section, rosters generated by the IP system are compared with manually generated rosters for 52 roster problems taken from the Gold Coast Hospital archives. These results are analysed and the applicability of an IP system in a hospital environment is discussed.

2. Nurse Rostering

Building a nurse roster involves selecting a *schedule* for each nurse (see figure 1). A schedule is a complete specification of shift values for one nurse for the duration of the roster. In the current study there are three shift values covering each 24-hour period (an early, late and night shift) and each roster lasts for 14 days.

Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun
late	*late*	*early*	*off*	*off*	*late*	*late*	*night*	*off*	*off*	*late*	*early*	*early*	*early*

Figure 1: An Example 14 day Nurse Schedule

The problem can be modelled in two different ways. Firstly, each nurse can be considered as a variable, with the domain of each variable being the set of schedules that the nurse is able to work [7, 13]. The rostering problem then becomes one of selecting a schedule from the set of all possible schedules for each nurse such that an optimal roster is generated. This approach will be referred to as the *possible schedules* method (see figure 2). The second approach is to consider each nurse/day slot in the roster as a variable [8, 19]. In this case, each nurse is associated with *n* different variables representing each day of the roster. The task of solving the roster then becomes one of selecting individual shift values that satisfy all hard constraints and minimise violations of soft constraints. This approach will be referred to as the *nurse/shift* method (see figure 3).

The current rostering problem is taken from a Queensland public hospital on the Gold Coast. In this hospital nurses are required to work various combinations of shifts during the same work stretch. For example, a late shift can be followed by another late shift, an early shift, a night shift or a day off. Most nurse rostering systems have been concerned with US rostering problems where nurses typically work blocks of the same shift type, which are then separated by days off (e.g., a late shift can only be followed by another late shift or a day off). The greater flexibility of the Gold Coast's rostering policy means the domain of possible schedules for each nurse is considerably larger. For instance, in Warner's 1976 study of a hospital in Michigan [7] the total number of possible schedules for all nurses was not expected to exceed 400. In the current study, after the problem has been simplified to consider only two shift types (day or night), problem sizes vary from 822 to 48,782 possible schedules per roster.

In other aspects, the problem domain is typical of rostering problems existing in larger hospitals throughout the world. Two 20 bed medical wards are considered, each employing between 25 to 37 staff members in each roster. There are five levels of staff seniority with constraints defining the numbers and skill mix of staff required for each shift. There are also locally accepted guidelines for the construction of nurse schedules. For instance, no full-time nurse should work more than 7 or less than 3 days without intervening days off. Nurses are also allowed to request particular shifts or days off in the roster. Part-time *and* full-time staff are included in the roster, with part-time staff often having special arrangements as to the type of schedules they will work.

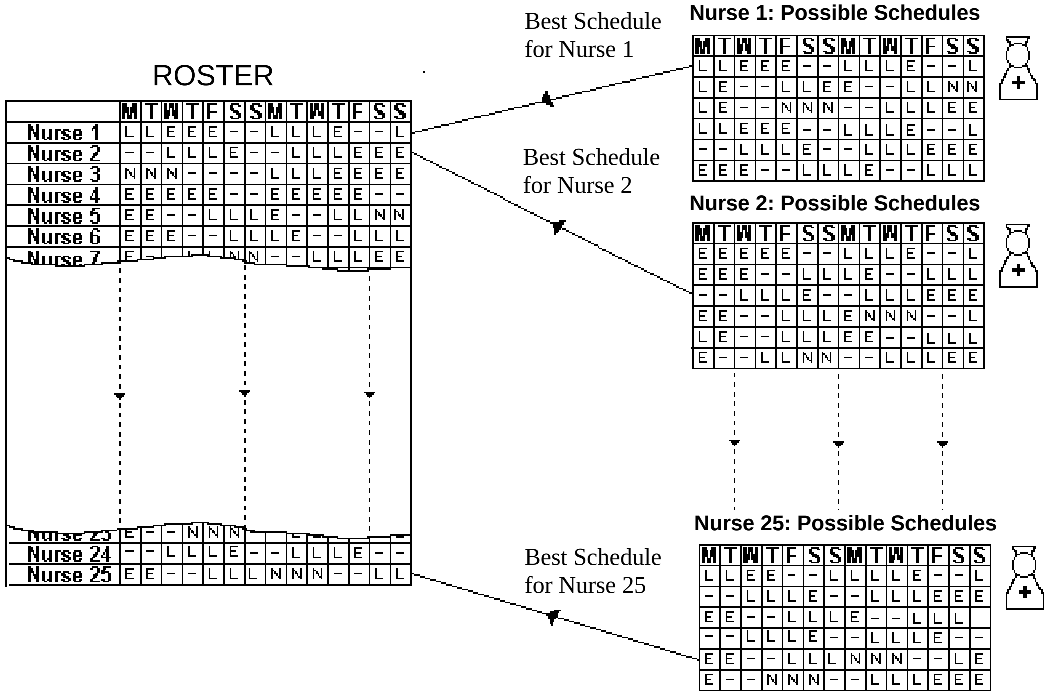

Figure 2: The Possible Schedule Selection Approach to Rostering

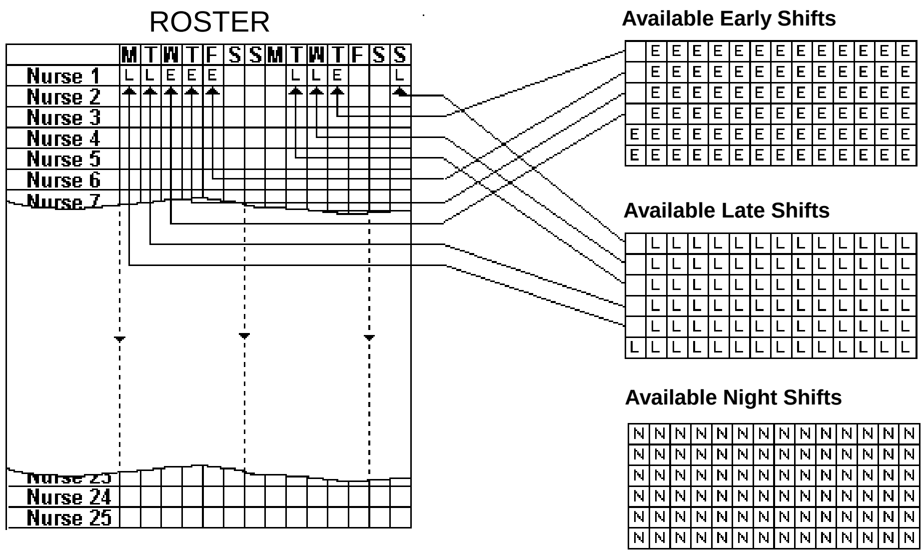

Figure 3: The Nurse/Shift Selection Approach to Rostering

Key: M = Monday, T = Tuesday, etc, E = Early Shift, L = Late Shift, N = Night Shift, - = Day Off

3. Overall Problem Formulation

3.1. Problem Decomposition

One of the constraints of the study is to develop a solution that is practical within the memory and computational resources of a 486 DX-50 personal computer with 8Mb of RAM. It was decided to start with Warner's model of the rostering problem, as this was the most complete and similar model to the problem in hand [7]. Other optimising models did not capture the full complexity of the problem or include an easy mechanism to minimise nurse dissatisfaction with schedules. The initial step in using Warner's model is to generate all possible schedules for each nurse in the roster. This requires the development of a schedule generating program that constructs all schedules that are allowable within the constraints. Using this program it became apparent that the number of schedules generated for the Gold Coast Hospital problem would exceed the memory and computational resources available. For this reason, the model was decomposed by considering the allocation of late and early shifts as a separate problem. This is possible, without loss of optimality, because the nurses considered in the study are able to interchangeably work late and early shifts. For example, if 7 nurses are required to work an early shift and 5 to work a late shift, this is equivalent to saying 12 nurses are required to work a day shift. The same problem decomposition would not apply to situations where nurses must work unbroken stretches of the same shift type. However, in such circumstances, fewer possible schedules are generated and the need to decompose the problem does not arise.

3.2. Two Modeling Approaches

In trying to solve the decomposed problem, it was still found that for $\frac{1}{3}$ of the rosters considered in the study there was insufficient memory to generate the schedules. These rosters were characterised by part-time nurses with few schedule constraints who were therefore able to work many thousands of schedules. Warner's *possible schedules* approach considers each nurse as a variable with a domain of possible schedules. In translating the model to an integer programming format, each *schedule* becomes a 0-1 decision variable. This is economical because the constraints needed to generate the nurse schedules are included in a schedule generation program rather than in the IP model. The schedule generation program adds flexibility to the system and allows the inclusion of sophisticated schedule constraints that could not otherwise be expressed as linear equations in the IP model. However, more variables are created than would be the case if each nurse/shift combination was considered as a variable[1]. A trade-off exists between the two modeling methods: Warner's possible schedule method typically requires more variables, whereas the nurse/shift variable method requires more constraints. In addition, the nurse/shift variable method is unable to model some of the heuristic constraints that can be included in Warner's schedule generation program, and on its own would produce too large and complex a model [11].

3.3. Combining Methods

Fortunately, part-time nurses with the fewest and simplest schedule constraints also cause the generation of the largest number of possible schedules and the consequent exhaustion of computer resources in Warner's model. The same nurses can be easily modelled using the nurse/shift variable method, without adding a large burden of new constraints to the system. As simple schedule constraints are involved, there is also no problem in expressing the constraints as linear inequalities. The two models were therefore combined. Any nurse recognised as having relatively few schedule constraints was tagged in the initial input to the problem and then instead of generating possible schedules for that nurse, 28 variables were created representing the 14 possible day shifts and 14 possible night shifts that the nurse could work.

3.4. The Overall System

The overall operation of the roster system involves two kinds of constraints, *schedule constraints* and *staff constraints*. These are itemised in the following two lists:

<u>Schedule Constraints</u>[2] (defined separately for *each nurse in the roster*):

- the maximum and minimum number of shifts a nurse can work without a day off

---

[1] For instance, a roster with 30 nurses, 3 shift types, lasting 14 days would have exactly $30 \times 3 \times 14 = 1260$ 0-1 variables (using one 0-1 variable for each possible shift value), whereas Warner's model could have up to 50,000 variables whilst only considering two shift types.

[2] The system includes exception values which allow certain constraints to be violated, for example if the minimum days off value for a nurse is 3, and there is an exception value of 1, then schedules that have one isolated day off are allowable for that nurse but all other consecutive day off stretches in the same schedule should be at least three days long.

- the maximum and minimum number of consecutive days a nurse can have off
- the maximum and minimum number of shifts a nurse can work per week
- the maximum and minimum number of shifts a nurse can work in the total roster
- the maximum and minimum number of nights a nurse can work in the total roster
- the number of separate blocks of night shifts a nurse can work
- the maximum number of night shifts a nurse can work in a single block
- the minimum number of days off required after a block of night shifts
- any requests for particular days on or days off during the roster
- whether the nurse has a preference for late or early shifts

Staff Constraints

- maximum, minimum and desired total staff for each shift type for each day of the roster
- maximum, minimum and desired staff of each seniority level for each shift type for each day of the roster
- a weighted score measuring the importance of each constraint

In addition to the above, a score is defined to measure the amount of dislike a nurse will feel for each possible schedule. For the purposes of this study, a general measure has been chosen, based on interviews with nursing staff. The central concept is that the ideal work pattern is a stretch of five shifts on followed by two days off. The further a schedule deviates from this ideal, the higher it's schedule score will be. Alternative schedule scoring methods are described in the literature. For instance, nurses can allocate their own schedule scoring preferences which can then included in the model [20].

The schedule constraints are used to generate all possible schedules for each nurse and, in conjunction with the staff constraints, the problem is translated into a series of linear inequalities (specified in next section). These inequalities are processed by the LP Solve linear programming package (this is a freely available shareware package that uses a standard branch and bound IP algorithm). The output of LP Solve is then read by C Program which heuristically generates a solution to the allocation of the late and early shifts. The heuristic method was chosen to save computational time and because the final late/early allocation is frequently altered by an informal system of shift swapping between nurses. However, an additional IP model could have been used to solve this allocation, as discussed in section 6.

4. The Mathematical Model

4.1. Constraints

Given the generation of a set of possible schedules for each nurse, (with the exception of nurses tagged for modeling using the nurse/shift variable method), each schedule can be represented by the variable X_{ij}, (where i = 1 to n and n = total nurses with possible schedules, j = 1 to J_i and J_i = total number of possible schedules for nurse i, and X_{ij} = (0 or 1), 1 if schedule is worked, 0 otherwise). As each nurse can only work one schedule, we have the following constraint:

$$\sum_{j=1}^{J_i} X_{ij} = 1$$

To model the different shifts worked in each schedule, a vector $\mathbf{a}_{ij}$ is created with $p \times q = t$ elements, p representing the number of days in the roster and q representing the number of shift types. In the current study, the $\mathbf{a}_{ij}$ vector would have $14 \times 2 = 28$ elements, the first 14 representing whether or not a day shift is worked at the associated position in the vector, and the second 14 representing whether or not a night shift is worked. For example, if the third value in $\mathbf{a}_{ij} = 1$, then a day shift is being worked on day 3 of the roster in schedule X_{ij}. Schedules belonging to nurses tagged for modeling using the nurse/shift variable method are represented using a vector $\mathbf{y}_k$ again containing $p \times q = t$ decision variables, y_{k1} to y_{kt}. As with the $\mathbf{a}_{ij}$ elements, each variable defines which type of shift is to be worked on a particular day. The k subscript indicates which nurse the vector applies to, where $k = (n + 1)$ to m, and m = total nurses in roster. The various staffing constraints for each day of the roster are represented as t element $\mathbf{b}_{constraint}$ vectors. For instance, $\mathbf{b}_{\max tot}$ stores the maximum number of staff allowable on the 14 day shifts (elements 1-14) and the 14 night shifts (elements 15-28). Seniority constraints can be represented using t element $\mathbf{s}_{i,level}$ vectors, with each element representing whether nurse i works at a particular seniority level on a particular day. For instance, if element 1 of $\mathbf{s}_{12,senior} = 0$, this would mean nurse 12 is not working at a senior level on the first day of the roster. Putting these vectors and variables together we can construct the basic constraints of the model. The constraint controlling the maximum and minimum levels of staff on each shift is of the form:

$$\mathbf{b}_{\min tot} \le \sum_{i=1}^{n} \sum_{j=1}^{J_i} \mathbf{a}_{ij} X_{ij} + \sum_{k=n+1}^{m} \mathbf{y}_k \le \mathbf{b}_{\max tot}$$

Similarly, constraints defining how many staff of each level are to work each shift are of the form:

$$\mathbf{b}_{\min senior} \le \sum_{i=1}^{n}\sum_{j=1}^{J_i} \mathbf{s}_{i,senior}\,\mathbf{a}_{ij}\,X_{ij} + \sum_{k=n+1}^{m} \mathbf{s}_{k,senior}\,\mathbf{y}_k \le \mathbf{b}_{\max senior}$$

In addition there are constraints describing the type of schedules allowable for nurses using the nurse/shift variable method (nurses $n + 1$ to m). Firstly, constraints are required to limit the number of shifts each nurse is to work in total. Using vectors of type $\mathbf{e}_{constraint}$, each with v elements, where $v = m - n + 1$, the vector $\mathbf{e}_{shifttot}$ can be used to represent the total number of shifts to be worked by each of the v nurses. In this case, the element $\mathbf{e}_{shifttot,1}$ = total number of shifts to be worked by nurse $n + 1$. Using this notation, the total shifts constraint is of the form:

$$\sum_{s=1}^{28} y_{r+n,s} = e_{shifttot,r},\ r = 1 \ldots v$$

Similarly, the constraints for the number of shifts to be worked each week are of the form:

$$e_{week\,\min,r} \le \sum_{s=1}^{7} y_{r+n,s} + \sum_{s=15}^{21} y_{r+n,s} \le e_{week\,\max,r},\ r = 1 \ldots v$$

$$e_{week\,\min,r} \le \sum_{s=8}^{14} y_{r+n,s} + \sum_{s=22}^{28} y_{r+n,s} \le e_{week\,\max,r},\ r = 1 \ldots v$$

Constraints for the number of night shifts are of the form:

$$e_{night\,\min,r} \le \sum_{s=15}^{28} y_{r+n,s} \le e_{night\,\max,r},\ r = 1 \ldots v$$

Constraints to ensure the maximum consecutive number of night shifts are not exceeded are of the form:

$$\sum_{s=1}^{e_{block\max,r}} y_{r+n,s+d} \le e_{block\,\max,r},\ r = 1 \ldots v,\ d = 13 \ldots (27 - e_{block\max,r})$$

Constraints to ensure a night shift and day shift do not clash are of the form:

$$y_{r+n,s} + y_{r+n,s+14} \le 1,\ \ r = 1 \ldots v,\ s = 1 \ldots 14$$

Constraints to ensure a night shift is not immediately followed by a day shift are of the form:

$$y_{r+n,s} + y_{r+n,s+13} \le 1,\ \ r = 1 \ldots v,\ s = 2 \ldots 14$$

If two days off are required after each night block then an additional constraint can be added of the form:

$$y_{r+n,s} + y_{r+n,s+12} \le 1,\ \ r = 1 \ldots v,\ s = 3 \ldots 14$$

Further schedule constraints can be added, but the above were found sufficient for the Gold Coast Hospital rostering problem.

4.2. Setting the Objectives

The secondary objective of the model, after satisfying the constraints, is to minimise nurse dissatisfaction with the schedules allocated. This can be incorporated into the model by including an objective function of the form [7]:

$$\text{minimise} \sum_{i=1}^{n}\sum_{j=1}^{J_i} c_{ij}\,X_{ij}$$

A c_{ij} coefficient is generated for each schedule in the model, representing a score as to how much nurse i dislikes schedule j. Given a rostering problem with a set of feasible constraints, the system so far described is able to find solutions to all the rostering problems generated from the Gold Coast Hospital study. Additional goal programming features may be added to the model, allowing for greater flexibility and also suggesting solutions when the problem is overconstrained. This is achieved by adding constraints defining a desired level of staff (in addition to maximum and minimum levels). Firstly a further series of 28 element $\mathbf{b}_{desired}$ vectors are defined, holding either the desired number of staff for each level of seniority or the desired overall number of staff required. Then a series of 28 variable $\mathbf{d}^{-}_{desired}$ vectors are defined each matching a corresponding $\mathbf{b}_{desired}$ vector. Each variable in the $\mathbf{d}^{-}_{desired}$ vector measures how far from the desired number of staff each shift constraint has deviated. The desired constraints are of the form:

$$\sum_{i=1}^{n}\sum_{j=1}^{J_i} \mathbf{a}_{ij}\,X_{ij} + \sum_{k=n+1}^{m} \mathbf{y}_k + \mathbf{d}^{-}_{desiredtot} \ge \mathbf{b}_{desiredtot}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{J_i} \mathbf{s}_{i,senior}\,\mathbf{a}_{ij}\,X_{ij} + \sum_{k=n+1}^{m} \mathbf{s}_{i,senior}\,\mathbf{y}_k + \mathbf{d}^{-}_{desiredsenior} \ge \mathbf{b}_{desiredsenior}$$

where each variable in the $\mathbf{d}^{-}_{desired}$ vectors is integral and ≥ 0

The new model can now be used to minimise the deviations of staff from the desired levels. More important constraints can be given appropriate weights in the objective function. For instance, it is usually more important to have sufficient total staff on a shift than to have sufficient of a particular seniority level of staff. An example objective function would be of the form:

$$\text{minimise} \sum_{k=1}^{28} w_{senior,k}\, d^{-}_{desiredsenior,k} + \sum_{k=1}^{28} w_{tot,k}\, d^{-}_{desiredtot,k}$$

where $w_{senior,k}$ and $w_{tot,k}$ represent the relative weights applied to the respective constraints. The minimum values for the $\mathbf{d}^{-}_{desired}$ vector variables derived from solving this model, represent the best possible allocation of staff within the given constraints. By setting the minimum staff constraints equal to this best possible allocation of staff, the original model (i.e. minus the desired constraints) can be used to find the best combination of schedules using the original objective function. In addition, if no feasible solution is found to an original problem, then the minimum staff constraints can be either be removed or reduced. By solving this relaxed model, the minimum deviations in the $\mathbf{d}^{-}_{desired}$ vectors will indicate which constraints are unattainable, while providing a possible solution to the original problem.

5. Experimental Results

The integer programming model described above was used to solve the 52 roster problems taken from the Gold Coast Hospital. Existing solutions to the problems were provided by the hospital. These were the actual rosters developed by hospital staff and used on two hospital wards during 1993. From these solutions the original problem parameters were reconstructed and fed into the IP rostering system. Using the schedule scoring method described in section 3.4 and quantified in table 1, an average schedule score for each roster solution was calculated. Through interviews with hospital staff a measure of shift allocation quality as also developed. This involved creating penalty weights for deviations away from desired levels of staff (see table 2). In addition to measuring the quality of rosters generated, the execution times for the branch and bound integer programming algorithm were recorded. The results obtained using these three measures are summarised in table 3.

Using a Multiple Analysis of Variance (MANOVA) statistical analysis the differences in shift allocation score and schedule quality score between manual and computer generated methods were found to be statistically significant ($\alpha = 0.01$) [18]. The calculation times for the IP algorithm did vary considerably, showing an approximately exponential growth in execution time as problem size increases (see figure 4).

Number of Days in Stretch	Day Off Cost	Day On Cost
1	15	10
2	0	5
3	5	2
4	10	1
5		0
6		1
7		5
8		15
9		20
10		30

Table 1: Costs used to Calculate Schedule Scores

Deviation	Total Staff Day Shift	Total Staff Night Shift	Senior Staff Day Shift	Senior Staff Night Shift	RN Day Shift
3 over desired level	15				
2 over desired level	5				
1 over desired level	1	50			
0 over desired level	0	0	0	0	0
1 under desired level	10	100	1	2	5
1 under minimum level	75	100	1	2	5
2 under minimum level	250				

Table 2: Cost used to Calculate Shift Scores

	Computer Rosters	Manual Rosters
Mean Schedule Score	11.86	18.23
Mean Shift Score	42.91	109.87
Mean Execution Time (seconds)	2553.62	n/a

Table 3: Comparison of Mean Scores for the 52 Roster Sample

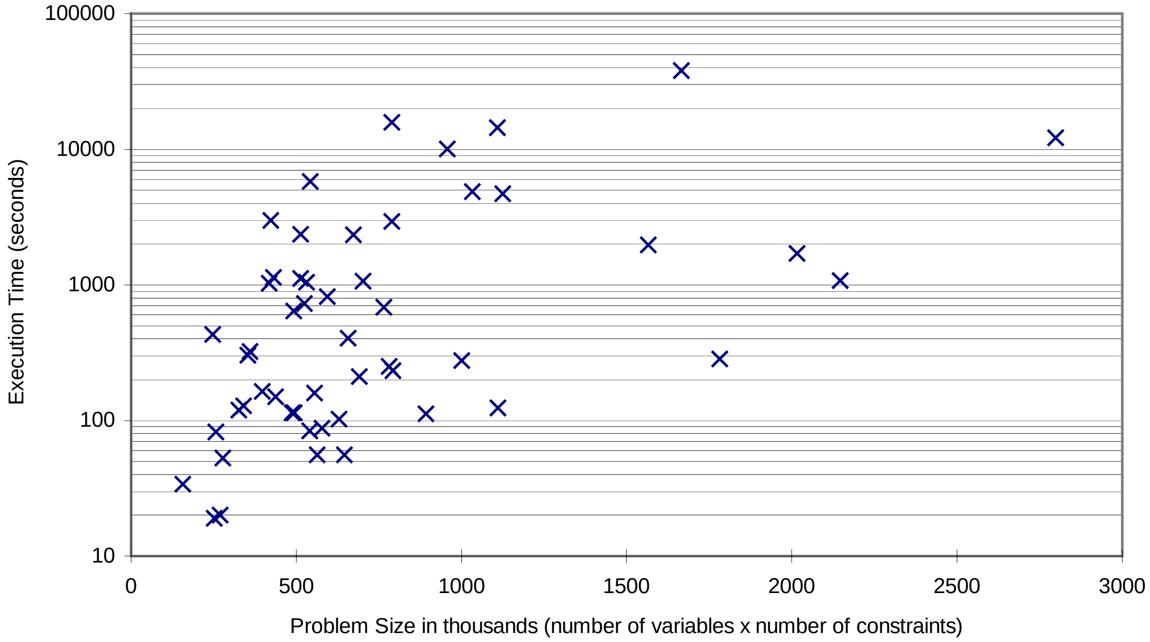

Figure 4: Graph of System Execution Times against Problem Size

6. Analysis

6.1. Overall Performance

The IP algorithm successfully solved all 52 rostering problems presented to it. According to the measures used in the study, these solutions were found to be of a higher quality than those generated by hospital staff. However, conclusions about differences in quality depend on the validity of the instruments used. The shift allocation and schedule quality scores are generalised and approximate measures only. It is quite feasible that hospital rostering staff had other situation specific priorities that are not captured by the measures used in this study. Therefore it cannot be concluded that the computerised rosters are better than the manually generated rosters in any absolute sense. However, further experience using the computerised rostering package at the hospital has shown that the system produces acceptable and workable solutions.

The important result is that an PC-based integer programming system is capable of solving a realistic rostering problem. The measures used to optimise the solution can easily be changed to suit the preferences of other nurses and ward situations.

Given these qualifications, the results do show the average schedule generated by the integer programming system to be considerably closer to an ideal of 5 days on and 2 days off than the average manually developed schedule. The difference in mean schedule scores of 6.37 (from table 3) is equivalent to reducing a 7 day stretch to a 5 day stretch in each schedule worked. The difference in mean shift quality of 66.96 (from table 3) is equivalent to reducing an understaffing of one nurse below the desired level of staff back to the desired level for 6
shifts in each roster. These differences would translate to noticeably better staff allocations and to more balanced schedules.

6.2. Unpredictability

The main drawback to the IP approach, as pointed out by other researchers, is the unpredictability of program execution time [16]. Whilst many problems are solved in a few minutes or even seconds, solution times of several hours are not unusual. Such lengthy and expensive calculations do not appear practical in a hospital ward situation (some organisations do consider the expense worthwhile, for instance see the airline crew scheduling problem [24, 25]). Use of more powerful computer resources and more sophisticated IP algorithms will reduce solution times but still cannot guarantee a particular problem is solved within a practical time frame. A promising avenue is to develop an IP algorithm that can predict how long a problem should take to solve. In the case of a branch and bound algorithm, this can be assessed by keeping track of the number of non-integral variables in the current solution. Harder problems can then be identified and solved using faster, but non-optimal, heuristic algorithms [13, 18].

6.3. Flexibility

Integer programming approaches to scheduling have also been criticised for a lack of flexibility [16, 11]. The current research model has built in flexibility. Firstly, the complex constraints used to define each nurse schedule are interactively defined by the user in the creation of a nurse input file. The values in this file also determine which schedules will be preferred in the optimisation process. Other definitions of schedule quality can be easily inserted into the model. Secondly, the constraints used to define staff levels can be selectively relaxed, using the goal programming model to minimise deviations below the desired levels of these constraints. The staffing constraints are also interactively defined via the creation of a staff constraint input file. The design of the model means it can be adapted to other rostering problems without changing the basic structure.

6.4. Limitations

The model assumes a three shift working day and that late and early shifts can be interchanged. A different system of schedule generation would be required for hospitals where nurses must work blocks of the same shift type [26]. The staffing constraints would also have to be extended to include an additional shift type. Nevertheless, the basic model is still applicable. Hospitals allowing more than three shift types are not considered in the current system, unless the extra shift types belong to an existing shift category. For instance, the Gold Coast Hospital requires some nurses to work a 12.30pm to 9.00pm shift, but such shifts are counted as part of the late shift and do not require the model to be modified. More complex shift systems have been developed with as many as 12 different start times and shift durations of between 4 and 12 hours. A goal programming application has been developed for the multiple start time problem, but only one seniority level of staff is considered [10].

6.5. Extensions to the Model

The IP model can also be used to solve the separate problem of allocating late and early shifts (the main model solves the problem in terms of day and night shifts). In this case, a new set of possible schedules can be developed for each nurse. These schedules would be based on all possible allocations of late and early shifts within the schedules selected in the main model solution. A different set of preferences would be used to develop schedule scores for the new schedules. For instance, nurses generally prefer to have an early shift before a day off, a late shift after a day off and to work as few late shifts followed by early shifts as possible. As in the main model, the objective of the late/early model is to minimise the overall schedule score subject to the staff constraints (the staff constraints now being the number and seniority of staff required on each early and late shift, the allocation of days off and night shifts having already been made).

In addition, the IP model can be adapted to cope with revisions to existing roster solutions. Revisions are frequently required due to staff sickness, absenteeism, resignation etc. The requirement in such circumstances is that the existing roster is changed as little as possible, as staff will already have made plans based on that roster. Other IP models have been criticised for being unable to handle such situations [16]. In the present case, the model can be rerun with any new constraints that have arisen (e.g. nurse *i* cannot work day *j*). The only change required to the model is that schedules are now scored in terms of haw far they deviate from the corresponding schedule in the existing solution. The objective of the model then becomes to minimise the deviation from the existing roster subject to the amended set of constraints.

7. Conclusions

The main conclusion to be drawn from this case study is that an IP approach to complex scheduling problems should not be dismissed without careful consideration. The current research has shown that a combination of modeling and problem decomposition techniques can break down the nurse rostering problem to a manageable size. The advantage of an IP approach over other heuristic techniques is that an optimal solution is obtained. In the case of nurse rostering, even a small improvement in roster quality can result in significant long term savings in staffing costs and/or an improved level of patient care.

The main and on-going problem with IP applications is the inherent unpredictability of execution times. Problems requiring many hours to solve may be acceptable in one-off situations, but experience in solving actual rostering problems has shown that nursing staff often decide to make several changes to an original roster solution. Requirements for problem reruns, and quick feedback to what-if staffing questions mean the *exclusive* use of an IP algorithm may prove impractical in an actual hospital situation. This research suggests further work is required in developing IP software that is able to recognise difficult and time consuming problems. Such problems can then processed by faster, non-optimising heuristic approaches.

8. References

[1]. M. Zweben and M. S. Fox: Intelligent Scheduling, Morgan Kaufmann, San Francisco, 1994.

[2]. U. Montanari: "Networks of Constraints: Fundamental Properties of and Applications to Picture Processing," Information Science, Vol. 7, pp. 95-132, 1974.

[3]. E. C. Freuder and R. J. Wallace: "Partial Constraint Satisfaction," Artificial Intelligence, Vol. 58, pp. 21-70, 1992.

[4]. D. Sitompul: "Design and Implementation of a Heuristic-Based Decision Support System for Nurse Scheduling," Doctoral Thesis, Oregon State University, 1992.

[5]. R. Hung: "Hospital Nurse Scheduling," Journal of Nursing Administration, Vol. 25(7), pp. 21-23, 1995.

[6]. M. Kostreva and P. Genevier: "Nurse Preferences vs. Circadian Rhythms in Scheduling," Nursing Management, Vol. 20(7), pp. 50-62, 1989.

[7]. D. M. Warner: "Scheduling Nursing Personnel According to Nursing Preference: A Mathematical Programming Approach," Operations Research, Vol. 24(5), pp. 842-856, 1976.

[8]. J. L. Arthur and A. Ravindran: "A Multiple Objective Nurse Scheduling Model," AIIE Transactions, Vol. 13(1), pp. 55-60, 1981.

[9]. A. A. Musa and U. Saxena: "Scheduling Nurses Using Goal-Programming Techniques," IIE Transactions, Vol. 16(3), pp. 216-221, 1984.

[10]. I. Ozkarahan and J. E. Bailey: "Goal Programming Model Subsystem of a Flexible Nurse Scheduling Support System," IIE Transactions, Vol. 20(3), pp. 306-316, 1988.

[11]. K. P. Chow and C. K. Hui: "Knowledge-Based System for Rostering," Expert Systems with Applications, Vol. 6, pp. 361-375, 1993.

[12]. L. D. Smith and A. Wiggins: "A Computer-Based Nurse Scheduling System," Computers and Operations Research, Vol. 4, pp. 195-212, 1977.

[13]. H. E. Miller, W. P. Pierskalla and G. J. Rath: "Nurse Scheduling Using Mathematical Programming," Operations Research, Vol. 24(5), pp. 857-870, 1976.

[14]. J. M. Lazaro and P. Aristondo: "Using SOLVER for Nurse Scheduling," In Proceedings of ILOG SOLVER & ILOG SCHEDULE First International Users Conference, July 1995.

[15]. B. M. W. Cheng, J. H. M. Lee and J. C. K. Wu: "A Constraint-Based Nurse Rostering System Using a Redundant Modeling Approach," In Proceedings of the Eighth IEEE International Conference on Tools with Artificial Intelligence, November 1996.

[16]. V. Dhar and N. Ranganathan: "Integer Programming vs. Expert Systems: An Experimental Comparison," Communications of the ACM, Vol. 33(3), pp. 323-336, 1990.

[17]. R. A. Blau and A. M. Sear: "Nurse Scheduling with a Microcomputer," Journal of Ambulatory Care Management, Vol. 6, pp. 1-13, 1983.

[18]. J. R. Thornton: "An Enhanced Cyclic Descent Algorithm for Nurse Rostering," Honours Thesis, Faculty of Engineering and Applied Science, Griffith University Gold Coast, 1995.

[19]. C. K. Hui: "Knowledge-Based Approach to Roster Scheduling Problems," Master's Thesis, University of Hong Kong, 1988.

[20]. M. Kostreva, M. Leszcynski and F. Passini: "The Nurse Scheduling Decision via Mixed-Integer Programming," In Proceedings of the American Hospital Association Forum on Nurse Scheduling, pp. 281-305, 1978.

[21]. M. Kostreva and K. Jennings: "Nurse Scheduling on a Microcomputer," Computers and Operations Research, Vol. 18(8), pp. 731-739, 1991.

[22]. H. A. Taha: Operations Research: An Introduction, 5th edition, Maxwell Macmillan International, Singapore, 1992.

[23]. M. R. Garey and D. S. Johnson: Computers and Intractability, Freeman, New York, 1979.

[24]. G. W. Graves, R. D. McBride, D. A. Gershkoff and D. Mahidhara: "Flight Crew Scheduling," Management Science Vol. 39(6), pp. 736-745, 1993.

[25]. K. L. Hoffman and M. Padberg: "Solving Airline Crew Scheduling Problems by Branch-and-Cut," Management Science, Vol. 39(6), pp. 657-682, 1993.

[26]. S. U. Randhawa and D. Sitompul: "A Heuristic-Based Computerized Nurse Scheduling System," Computers and Operations Research, Vol. 20(8), pp. 837-844, 1993.