

Cyclic Preference Scheduling for Nurses Using Branch and Price

Hadi W. Purnomo,¹ Jonathan F. Bard²

¹ American Airlines, AMR Corp. Headquarters HDQ1, Mail Drop 5358, Fort Worth, Texas 76155

² Graduate Program in Operations Research & Industrial Engineering, 1 University Station C2200, University of Texas, Austin, Texas 78712-0292

Received 7 December 2005; revised 25 May 2006; accepted 2 September 2006

DOI 10.1002/nav.20201

Published online in Wiley InterScience (www.interscience.wiley.com).

Abstract: This paper presents a new methodology to solve the cyclic preference scheduling problem for hourly workers. The focus is on nurse rostering but is applicable to any organization in which the midterm scheduling decision must take into account a complex of legal, institutional, and preferential constraints. The objective is to strike a balance between satisfying individual preferences and minimizing personnel costs. The common practice is to consider each planning period independently and to generate new rosters at the beginning of each. To reduce some of the instability in the process, there is a growing trend toward cyclic schedules, which are easier to manage and are generally perceived to be more equitable. To address this problem, a new integer programming model is presented that combines the elements of both cyclic and preference scheduling. To find solutions, a branch-and-price algorithm is developed that makes use of several branching rules and an extremely effective rounding heuristic. A unique feature of the formulation is that the master problem contains integer rather than binary variables. Computational results are reported for problem instances with up to 200 nurses. Most were solved within 10 minutes and many within 3 minutes when a double aggregation approach was applicable. © 2006 Wiley Periodicals, Inc. *Naval Research Logistics* 54: 000–000, 2007.

Keywords: cyclic scheduling; branch and price; column generation; nurse scheduling; feasibility heuristics

1. INTRODUCTION

Personnel scheduling in the service industry presents challenges that are absent in most manufacturing environments where 8-hour shifts are the norm. Organizations such as hospitals, call centers, airlines, and retail outlets typically operate up to 24 hours a day, 7 days a week, and face widely fluctuating demand during the week. The consequences of poor planning are often evidenced by low job satisfaction and morale accompanied by high turnover rates and increased recruitment and training costs (e.g., see [36]). Overstaffing results in excessive idle time and inflated payrolls, while understaffing leads to poor customer service and a potential loss of revenue. To cope with persistent variations in demand, it is common for service organizations to use multiple starting times, a combination of full- and part-time workers, and a range of shift lengths (e.g., see [35]).

The objective of the staff scheduling problem addressed here is to find a feasible set of rosters that minimizes the

weighted sum of labor costs and individual preference violations. A roster is defined as a combination of shifts and days-off assignments over a fixed period of time. The problem is complicated by the need to comply with government regulations, contractual agreements, and local policies related to weekend assignments, days-on and days-off patterns, maximum workstretches, and undesirable shift patterns, to name a few. In this paper, we focus on midterm nurse scheduling and develop a new model and solution algorithm for constructing cyclic schedules, i.e., schedules (rosters) that can be repeated periodically as long as the demand per shift remains relatively constant from one planning period to the next [16, 20]. The algorithm is now being implemented as part of a suite of optimization tools in the nurse management system, CareWare, and represents an order-of-magnitude improvement over current practice.

Problems related to the planning and scheduling of nurse resources can be viewed hierarchically and decomposed along the time axis. In the long run, the goal is to specify the composition of the permanent workforce by skill and to construct shift profiles that meet some measure of demand per planning cycle. The nursing staff at most hospitals

Correspondence to: J.F. Bard (jbard@mail.utexas.edu)

includes registered nurses, licensed practical nurses, technicians, and nurse aides. Each category may be further divided by specialty and experience.

Midterm scheduling fixes the work assignments for up to 6 weeks at a time. Each department or ward generates its own rosters independently, using average demand as input. Given the midterm rosters, further adjustments and decisions still need to be made on a short-term basis to deal with unplanned leave and spikes in demand. This is the daily adjustment problem [3, 40]. When supply exceeds demand in a unit, a nurse may be asked to float to another unit or may be sent home. When a unit is short of staff, the hospital has several options such as requesting overtime, calling in casuals, asking a nurse to work on his or her day off, or requesting outside resources. The latter include float pool and agency nurses. Each option imposes a different set of costs.

In the next section, we review the literature on staff scheduling with an emphasis on nurse rostering. In Section 3, we define the cyclic preference scheduling problem and present our constraint-based integer programming model, which is then transformed into a set-covering-type model using Dantzig–Wolfe decomposition. What we obtain is a column-oriented master problem in integer variables, a feature unique in the literature, and a set of independent 0–1 subproblems. This is followed in Section 4 with a discussion of our branch-and-price algorithm. Computational results for problem sets with up to 200 nurses are highlighted in Section 5 where it is seen that most solutions were obtained within a few minutes. We close with some remarks on the effectiveness of the approach.

2. BACKGROUND AND LITERATURE REVIEW

Although there has been abundant research on personnel scheduling, most efforts have focused on either shift scheduling or days-off scheduling with the objective of minimizing total cost (e.g., see [9, 14]). These problems are the central components of midterm or tour scheduling (e.g., see [11, 26]). In the past 2 decades, most of the research on nurse scheduling has concentrated on rostering with the aim of accommodating individual preferences, such as requests for specific shifts or days off. A high proportion of hospital staffing costs are associated with nursing resources, so generating schedules that better match supply with demand can have a significant impact on the operating budget [34].

In recent years, many hospitals have gone to self-scheduling where nurses create their own rosters based on eligibility and contractual agreements. When conflicts occur, adjustments are made through consensus [24]. Although it is easy to implement, self-scheduling may not be practical in all situations. As a consequence, more systematic approaches have been developed. The most popular is prefer-

ence scheduling, which embodies two key ideas: work rules and the quantification of individual preferences. Work rules are sometimes referred to as hard constraints and derive from labor contracts and regulatory statutes. Because they are non-negotiable, all rosters must comply with them. To quantify preferences, desirable and undesirable patterns (soft constraints) are identified and assigned bonuses or penalties depending on the severity of a violation. The assignment may be done by asking nurses to freely distribute points over a list of possible violations [40] or by using an automated process that has various assumptions built in about penalty levels [27].

One disadvantage of the preference scheduling approach is its inherent inconsistency. Due to the implicit assumption of independence of consecutive planning horizons, nurses may have noticeably different shift assignments from week to week and month to month. This can be unsettling for those who need to make personal plans well in advance. An alternative approach that has not yet been widely adopted due to the inherent difficulty in finding solutions is to include a cyclic feature into the problem along with the preferences.

For an in-depth review of the nurse rostering literature, see [13]. A brief summary follows.

Heuristics

Early research on nurse scheduling was primarily aimed at developing efficient heuristics. Miller et al. [31] were the first to formally address the preference scheduling problem. Starting with an initial solution, they developed a greedy neighborhood search procedure to find local optima (also see [37]). Howell [25] solved the cyclic scheduling problem by combining intuitive information of what constitutes a good schedule with greedy exchanges. More recently, metaheuristics, such as tabu search and genetic algorithms (GAs), have been designed for various midterm scheduling problems. The distinguishing feature of these methods is their ability to escape local optimality.

Tabu search, as designed for combinatorial optimization problems, keeps a list of the most recently visited solutions and prohibits the search function from returning to them for several iterations. To enhance its performance, additional features such as strategic oscillation, intensification–diversification, and multiple neighborhood types were adopted by Dowland [18] for the preference scheduling problem. Working independently, Burke et al. [12] implemented a tabu search algorithm with diversification and greedy shuffling in the Plane software system to schedule nurses at several Belgium hospitals. Similar implementations for constructing nurse rosters can be found in [32].

Starting with several initial solutions, GAs use crossover and mutation operators to generate pairs of improved solu-

tions. In the context of preference scheduling, Aickelin and Dowsland [1] transformed each roster into a string. This representation allowed the original problem to be viewed as a permutation problem, which is more suitable for GAs due to the nature of the exchange procedures. Heuristics were used to reconstruct solutions from the strings. Related approaches are discussed by Aickelin and White [1] and Kawanaka et al. [28].

Exact Methods

Based on the definition of the decision variables, there are two principal ways of formulating staff scheduling problems as integer programs (IPs). The first is the pattern-view formulation and leads to a set-covering-type problem with a large number of columns. In these models, each column represents a unique scheduling pattern or roster consisting of a sequence of shifts and days-off assignments that span the planning horizon. The second formulation is based on the shift view of the problem and often contains a large number of rows. The rows can generally be partitioned into system-wide constraints and individual employee constraints. Each formulation has its particular advantages, but the underlying problem remains NP-hard [29].

Exact algorithms typically involve some form of decomposition or the use of cutting planes derived from polyhedral theory. The column generation approach, an example of decomposition, uses the pattern-view formulation as a master problem and either heuristics or the shift-view formulation as subproblems to generate candidate rosters. In a call center application, Caprara, Monaci, and Toth [15] simplified the subproblems into network problems that were easily solved.

In the vast majority of cases, the pattern-view formulation only includes demand constraints, which gives it the advantage of simplicity, although the number of columns grows exponentially with the length of the planning horizon. In addition, when using a heuristic to generate columns, a separate code must be included to check for violations of the hard constraints and to calculate violations of soft constraints. These can be time-consuming activities [4]. The shift-view formulation eliminates these drawbacks, but gives a weaker formulation. Moreover, it requires an explicit representation of all the rules, which may require multiple sets of constraints and a proportionate number of logic variables.

An easy way to overcome the formidable size of the set-covering formulation is to generate only a subset of columns at a time. Warner [40] selected 50 columns with the help of a greedy method to set up a problem with 20 nurses. He used a block pivoting strategy to find feasible solutions. Taking this a step further, Jaumard, Semet, and Vovor [27] developed a branch-and-price (B&P) algorithm

for the preference scheduling problem. In this approach, a master problem is created that contains the system level constraints only. The hard and soft constraints are contained in a series of subproblems, one for each nurse, that are solved iteratively to generate columns for the master problem. Preliminary testing showed that instances with up to 41 nurses could be solved for 2-week blocks in about 16.5 hours on a Sparc Sun 5 workstation.

The cutting plane approach is based on the shift-view formulation and aims to generate valid inequalities to tighten the linear programming (LP) relaxation of the IP model. Felici and Gentile [22], for example, were able to generate strong cuts by exploiting the weekend constraints associated with a staff rostering problem at the Italian airline, Alitalia. Beginning with the shift-view formulation, Valouxis and Housos [38] developed a hybrid approach that solved a simplified version of the original preference scheduling problem that was constructed by ignoring several difficult-to-model constraints. After finding a solution to the reduced IP, a local search heuristic was used to achieve feasibility. Alternatively, goal programming is a common methodology for dealing with soft constraints. In this approach, rules are prioritized and treated as goals or objectives to be satisfied. The optimization is carried out sequentially so to ensure that goal achievement is preserved, e.g., see [10, 23]. Our model makes use of this idea.

Artificial Intelligence Methods

Approaches originating in computer science have also been devised to solve nurse scheduling problems. Okada [33] adapted an artificial intelligence (AI) procedure called memetic learning to construct rosters. Constraint programming (CP) is also an AI technique that, unlike integer programming, does not make use of an explicit mathematical representation of the hard and soft constraints. Instead, the constraints are used to formulate propagation rules that reduce the search space. When constraint satisfaction is difficult to achieve, each constraint is assigned a different priority that is used to drive the search process. Solution quality is judged on the basis of priority satisfaction. Cheng, Lee, and Wu [17] worked with 20 different constraints to construct rosters for up to 30 nurses. In their approach, a redundant constraint was employed to enhance the propagation effect and further reduce the search space. For a bibliography on nurse rostering, see the survey by Cheang et al. [16]; for a general bibliography on staff scheduling, see Ernst et al. [21].

3. CYCLIC PREFERENCE SCHEDULING AND MODEL DEVELOPMENT

When the nursing staff is fixed, the objective of cyclic scheduling is to generate a set of rosters that minimizes the

number of uncovered shifts. Nurses are then either rotated among rosters in consecutive planning periods or they are assigned the same roster until a change is called for. Alternatively, when the objective is to minimize a “dissatisfaction” cost associated with violations of soft constraints, the problem is called preference scheduling. The resulting rosters are not cyclic and so must be generated anew for each planning period. The problem addressed in this paper, which we call *cyclic preference scheduling*, combines elements of both approaches and includes multiple, overlapping shifts, restricted work hours, individual preferences, and rotational profiles.

In the model, five different shift types are used. The first three divide the day evenly into non-overlapping periods of 8 hours each and are referred to as day (D), evening (E), and night (N). The remaining two divide the day into 12-hour, non-overlapping periods and are called AM and PM. The starting time of an AM shift coincides with the starting time of a D shift, while the end time of a PM shift coincides with the end time of an N shift. Demand is specified as a lower and upper bound on the number of nurses needed per shift per unit. Because of nation-wide shortages in the United States and most western European countries, it is unusual to be able to cover all demand with only the in-house staff. It is assumed, therefore, that gaps in the schedule will later be filled with agency or per diem nurses. Short-term scheduling addresses this issue.

To complete the description of the problem we define a rotational profile by the triplet (eligible shifts, ratio, total hours). Employment rules in most hospitals limit a nurse to work at most two different shift types, such as D and E, over a 2-week planning horizon. More than this number is undesirable from an ergonomic point of view because it may lead to poor performance and higher incidences of unintentional patient injury. The *ratio* indicates the minimum number of shifts of each type that must be assigned and is often expressed as a percentage. For example, a D/E nurse with a 40% ratio whose contract calls for 80 hours of work in 2 weeks (D/E, 40%, 80) must be assigned 10 shifts over this period. At least 4 of those shifts must be D and 4 E. An E/PM nurse with a 25% ratio who is contracted for 72 hours of work in 2 weeks (E/PM, 25%, 72) must be assigned at least 2 E shifts and 2 PM shifts. The possibilities are (3E, 4PM) and (6E, 2PM). Smaller ratios afford more scheduling options so they are naturally preferred by management. For completeness, nurses who work straight shifts are said to have a degenerate rotational profile.

The hard constraints considered in the model include the following.

- a. All full-time nurses must be assigned either 72 or 80 hours within a 2-week planning period, depending

on their contract. When a nurse is assigned fewer hours than specified in the contract, she is still paid her full weekly salary. Cancellations and overtime are taken into account on a daily basis and are not part of our model (see [3] for a discussion of the daily adjustment problem).

- b. A nurse can only be assigned shifts that are associated with her rotational profile.
- c. The number of consecutive working days, also commonly called the *workstretch*, cannot exceed some value, call it D^{\max} . In most cases, this parameter is set to 5.
- d. A nurse can work for at most 12 hours in a day, which means that at most one shift can be assigned in a day. Also, there needs to be at least an 8-hour break between consecutive assignments. Compliance is generally automatic because of the 12-hour rule; however, additional restrictions are required for those profiles in which back-to-back shifts are possible. In particular, the following sequences are not permitted: N/D, PM/D, N/AM, and PM/AM.
- e. Nurses must work two weekend shifts in the same weekend every 2 weeks. For our purposes, the first weekend shift starts at 7:00 p.m. on Friday and the last weekend shifts starts at 3 p.m. on Sunday.

Two soft constraints are also considered:

- f. Days-on and days-off patterns. There are two undesirable working patterns. The first involves 1 day off between 2 working days and is denoted by on-off-on or 1-0-1 [40]. The second involves a day on between 2 days off and is denoted by off-on-off or 0-1-0. It is more desirable to have at least 2 consecutive days off. In our implementation, nurses who only work 12-hour shifts (AM, PM or both) are not subject to the 1-0-1 and 0-1-0 soft constraints because most hospitals view them as too restrictive for rotational profiles with 12-hour shifts only.
- g. Different shift assignments on consecutive working days. This situation may occur when a nurse is assigned to work a sequence such as D/E/D without an intervening day off. This type of pattern is highly undesirable because it disrupts the body’s circadian rhythm.

3.1. Shift-View Model

In midterm scheduling for nurses, the planning horizon is generally 4 weeks so individual rosters must be constructed for this length of time. When preference scheduling is used, rosters are constructed for the full planning horizon at once; when cyclic scheduling is used, 14-day rosters are con-

structured and then repeated every 2 weeks. To be consistent with the literature, we define the planning horizon, D_{\max} , to be a multiple of 2 weeks, with the idea that we are really going to solve a 14-day problem and then repeat the solution in 2-week blocks. The model that we start with takes a shift view and includes both the hard and the soft constraints. The following notation is used in the developments.

Sets and Indices

D	set of days for which the model is to be solved; $d \in D = \{1, 2, \dots, 14\}$
D_W	set of weekend days in a 2-week period
N	set of nurses to be scheduled; $i \in N$
N_R	set of nurses with rotation patterns (two possible shifts); $N_R \subseteq N$
N_{BB}	set of nurses with back-to-back rotational profiles (N/D, PM/D, N/AM, PM/AM); $N_{BB} \subseteq N_R \subseteq N$
T	set of all possible shift types considered, $t \in T = \{D, E, N, AM, PM\}$
T_i	set of shift types that nurse i is hired to work; $T = \cup_{i \in N} T_i$
W	set of weeks under consideration; $m \in W$
a	index for the number of preference violations; $a = 1, \dots, V_{\max}$
t_1 (t_2)	first (second) shift in T_i for nurse i when $ T_i = 2$.

Parameters

r_a	penalty assigned to a roster that has a violations
h_t	length of shift t (hours)
M_t	large number representing the cost of an outside nurse (undercoverage) for shift t
H_i	number of hours nurse i is contracted to work every 2 weeks
LD_{dt} (UD_{dt})	lower (upper) demand requirement for shift t on day d
D_t^{\max}	maximum number of consecutive days (workstretch) that nurse i is permitted to work
P_{it}	minimum number of shifts of type t that nurse i must work every 2 weeks
W_i^{\max}	number of weekend shifts nurse i must work every 2 weeks
V_{\max}	maximum number of violations allowed for each nurse
TR_{\max}	maximum number of transitions from one shift type to another on consecutive days allowed in 14 days
O_{dt}^{\max}	maximum number of outside nurses that can be assigned to shift t on day d

Decision Variables

x_{idt}	(binary) 1 if nurse i works shift t on day d , 0 otherwise (note that x_{idt} are circulation variables; $x_{i,14+l,t} \equiv x_{ilt}$, $l = 1, \dots, D_i^{\max}$)
w_{im}	(binary) 1 if nurse i works on weekend m , 0 otherwise
V_{ia}	(binary) 1 if nurse i has a violations, 0 otherwise
b_{id}	(accounting) 1 if nurse $i \in N_R$ works shift t_1 on day d and shift t_2 on day $d + 1$, 0 otherwise; $t_1 \neq t_2$
p_{id}	(accounting) 1 when nurse i has a 0-1-0 pattern that starts on day d , 0 otherwise
q_{id}	(accounting) 1 when nurse i has a 1-0-1 pattern that starts on day d , 0 otherwise
y_{dt}	number of outside nurses assigned to shift t on day d
s_{dt}	excess number of nurses assigned to shift t on day d

$$\theta_{IP} = \text{Minimize} \sum_{i \in N} \sum_{a=1}^{V_{\max}} r_a V_{ia} + \sum_{d \in D} \sum_{t \in T} M_t y_{dt} \quad (1a)$$

$$\text{subject to} \sum_{i \in N} x_{idt} - s_{dt} + y_{dt} = LD_{dt}, \quad d \in D, \quad t \in T \quad (1b)$$

$$\sum_{d \in D} x_{idt} \geq P_{it}, \quad i \in N_R, \quad t \in T_i \quad (1c)$$

$$\sum_{d \in D} \sum_{t \in T_i} h_t x_{idt} = H_i, \quad i \in N \quad (1d)$$

$$\sum_{t \in T_i} x_{idt} \leq 1, \quad i \in N, \quad d \in D \quad (1e)$$

$$x_{idt_2} + x_{i,d+1,t_1} \leq 1, \quad i \in N_{BB}, \quad d \in D \quad (1f)$$

$$\sum_{l=d}^{d+D_i^{\max}} \sum_{t \in T_i} x_{ilt} \leq D_i^{\max}, \quad i \in N, \quad d \in D \quad (1g)$$

$$\sum_{d \in D_w} \sum_{t \in T_i} x_{idt} = W_i^{\max} w_{im}, \quad i \in N, \quad m \in W \quad (1h)$$

$$\sum_{m \in W} w_{im} = 1, \quad i \in N \quad (1i)$$

$$\sum_{t \in T_i} x_{idt} + \left(1 - \sum_{t \in T_i} x_{i,d+1,t} \right) + \sum_{t \in T_i} x_{i,d+2,t} + p_{id} \geq 1, \quad i \in N, \quad d \in D \quad (1j)$$

$$\left(1 - \sum_{i \in T_1} x_{idt}\right) + \sum_{i \in T_1} x_{i,d+1,t} + \left(1 - \sum_{i \in T_1} x_{i,d+2,t}\right) + q_{id} \geq 1, \quad i \in N, \quad d \in D \quad (1k)$$

$$1 - x_{id\alpha} + 1 - x_{i,d+1,\beta} + b_{id} \geq 1, \quad i \in N_R, \quad d \in D, \quad \alpha \neq \beta \in \{1,2\} \quad (1l)$$

$$\sum_{d \in D} b_{id} \leq TR_{\max}, \quad i \in N_R \quad (1m)$$

$$\sum_{d \in D} (p_{id} + q_{id} + b_{id}) = \sum_{a=1}^{V_{\max}} a v_{ia}, \quad i \in N \quad (1n)$$

$$\sum_{a=1}^{V_{\max}} v_{ia} \leq 1, \quad i \in N \quad (1o)$$

$$0 \leq s_{dt} \leq UD_{dt} - LD_{dt}, \quad 0 \leq y_{dt} \leq O_{dt}^{\max}, \quad \forall t, d \quad (1p)$$

$$b_{id}, p_{id}, q_{id} \geq 0, \quad \forall i, t, d; \quad v_{ia} \in \{0,1\}, \quad \forall i, a; \quad w_{im} \in \{0,1\}, \quad \forall i, m \quad (1q)$$

$$x_{idt} \in \{0,1\}, \quad \forall i, t, d, \quad \text{where } x_{i,14+l,t} \equiv x_{ilt}, \quad l = 1, \dots, D_i^{\max} \quad (1r)$$

The objective (1a) is to minimize the weighted sum of preference violations and the cost of covering gaps with outside nurses. The choice of the parameters M_t implicitly defines the tradeoff between satisfying the collective preferences of the nurses and incurring additional costs by allowing for shortages. In general, we want each $M_t \gg$ the penalty coefficient $r_{v_{\max}}$ to ensure that costs are minimized before preferences, although other options are possible. In our application, r_a is the exponential function 2^{a-1} , where $a \in [1, V_{\max}]$ is the total number of violations associated with a roster (see [4] for more discussion of this issue).

Constraints (1b) correspond to the demand requirement for each shift t on day d and represent a transformation from a two-sided inequality into a single equality constraint with an upper bound on the slack variable s_{dt} , as indicated in (1p). With some algebra, it can be shown that the number of nurses assigned to shift t must be at least LD_{dt} , and no more than UD_{dt} ; that is, $LD_{dt} \leq \sum_{i \in N} x_{idt} \leq UD_{dt}$, for all $d \in D, t \in T$. An optimal solution will exist with s_{dt} and y_{dt} integral so they can be treated as continuous variables. The complementary conditions $s_{dt} \times y_{dt} = 0$ will also hold for all d and t . Note that some hospitals express demand in terms of

periods rather than shifts, but one can be easily converted to the other (see [27]).

Hard Constraints

Constraints (1c) are applicable for all $i \in N_R$ and guarantee that at least P_{it} shifts of type t are assigned every 2 weeks, where P_{it} is determined from the *ratio*. For nurses with a single shift profile, i.e., $i \in \mathcal{M}N_R$, (1c) can be removed. Equation (1d) states that the total number of hours assigned to nurse i must be equal to the number of hours H_i that she is contractually obligated to work every 2 weeks.

Constraints (1e) restrict a nurse to at most one shift assignment within 24 hours. Because the length of a shift is at most 12 hours, constraints (1c)–(1e) automatically ensure an 8-hour break between shifts for nurses with nondegenerate rotational profiles except for the back-to-back cases mentioned in rule (d). For $j \in N_{BB}$, constraint (1f) rules out back-to-back shifts, i.e., it permits only one assignment of either an N or a PM shift (t_2) on day d , or a D or an AM shift (t_1) on day $d + 1$.

Constraints (1g) limit the workstretch of nurse i to no more than D_i^{\max} days in any time window of $D_i^{\max} + 1$ consecutive days. This corresponds to rule (c). The parameter D_i^{\max} is set to 5 for nurses who work for 8-hour shifts only and 4 for nurses who work both 8- and 12-hour shifts. Because the problem is cyclic, day 14 is followed by day 1. This is indicated in (1r). The weekend rule (e) is modeled by constraints (1h)–(1i). Weekends are defined by N and PM shifts for Friday and Saturday and by D, E, and AM shifts for Saturday and Sunday. Together, these constraints require that nurse i work exactly W_i^{\max} weekend days every 2 weeks. Although the days must fall on the same weekend, it is an easy matter to allow split weekends. Note that the value of W_i^{\max} is a function of the rotational profile. In our implementation, if nurse i works only 12-hour shifts, then she will be assigned just 1 weekend day ($W_i^{\max} = 1$) every 2 weeks; otherwise, $W_i^{\max} = 2$.

Soft Constraints

Constraints (1j)–(1o) determine the quality of the schedules. The undesirable patterns are counted in the model by the variables p_{id} , q_{id} and b_{id} . A 0-1-0 pattern starting on day d implies that $\sum_{i \in T_1} x_{idt} = 0$, $\sum_{i \in T_1} x_{i,d+1,t} = 1$, and $\sum_{i \in T_1} x_{i,d+2,t} = 0$. Constraints (1j) are implication constraints that set p_{id} to 1 when such a pattern exists. Because all of the other variables in these individual constraints are binary and all of the data are integral, p_{id} will always be integral in an optimal solution so it can be treated as a continuous variable. Constraints (1k) are the corresponding implication constraints for 1-0-1 patterns, which detect the existence of

$\sum_{t \in T_i} x_{idt} = 1$, $\sum_{t \in T_i} x_{i,d+1,t} = 0$, and $\sum_{t \in T_i} x_{i,d+2,t} = 1$ starting on day d . The total number of these patterns for nurse i is given by the summation $\sum_{d \in D} (p_{id} + q_{id})$. Implicit in the formulation is that a schedule is a circulation so that in (1j) and (1k), day $14 + d = \text{day } d$. This eliminates the need to introduce initial conditions. Constraint (1l) detects a shift transition during consecutive days and must be included for all possible combinations of shift transitions that nurse i may have. The maximum number permitted is given by the parameter TR_{\max} , as indicated in (1m). The next two constraints, (1n) and (1o), count the number of preference violations and determine which penalty coefficient r_a will be in effect, respectively. If it were desirable to account for the severity of each violation, we could do this by multiplying each variable in (1n) by the appropriate weight. The bounds in (1p) on the y_{dt} and s_{dt} variables limit the amount of under- and over-coverage, respectively, for each shift on each day. If O_{dt}^{\max} and $UD_{dt} - LD_{dt}$ are too tight, then problem (1) may be infeasible.

Additional Factors

Although personal requests and other managerial prerogatives are not included in the model, enough flexibility exists to permit many such options. For example, when nurse 1 does not want to work on day d , we can add the constraint $\sum_{t \in T_1} x_{1dt} = 0$, or when nurses 3 and 4 share a common shift type, say t_1 , and we do not want to schedule them together, we would include the constraint $x_{3dt_1} + x_{4dt_1} \leq 1$ for all $d \in D$. It is also an easy matter to include a variety of additional rules, such as the requirement that each nurse be given at least two days off in a row. This requirement can be imposed by removing q_{id} from constraint (1k). The more general case in which nurse i must have at least $D_i^{\min\text{off}}$ consecutive days off cannot be implemented efficiently.

Also not included in the model are grade considerations and the use of higher skilled workers to fill in for lower skilled workers when there is idle time in their schedules. Not accounting for seniority and individual skills opens up the possibility that the model might produce schedules in which some shifts are staffed by inexperienced nurses only, an undesirable situation. Our approach to dealing with this imbalance is called *downgrading* and was addressed in an earlier paper (see [5]). The downgrading component of model (1) was omitted to avoid unnecessary notation.

The size of model (1) is determined largely by the constraints (1e)–(1g), (1j)–(1l). While the other constraints grow linearly with the number of nurses in a unit, these constraints grow at a rate proportional to $O(|N| \cdot |D|)$. The number of variables grows at a rate proportional to $O(|N| \cdot |D| \cdot |T|)$. A small problem with 20 nurses and a 2-week

planning horizon contains roughly 1500 variables and 700 constraints. A medium size problem for the same planning horizon with 50 nurses requires about 5000 variables and 3500 constraints.

3.2. Reformulation as Set-Covering-Type Model

Initial attempts to solve 20-nurse instances of (1a)–(1r) with CPLEX 7.5 on a PC were not very encouraging. In most cases, it was not even possible to find feasible solutions within several hours. This experience led to the development of a B&P algorithm, a combination of (i) Dantzig–Wolfe (D–W) decomposition extended to accommodate integer variables and (ii) standard branch and bound (B&B) [30, 39, 41]. In midterm nurse scheduling, it is natural to separate the demand constraints (1b) from the other constraints and to create a master problem (\mathcal{MP}) that resembles the pattern-view formulation. Columns in \mathcal{MP} correspond to feasible rosters for the various rotational profiles and the objective function is the same as (1a). Having removed the demand constraint, we are left with (1c)–(1r), which conveniently decompose by nurse, giving $|N|$ subproblems. Points in the feasible region of subproblem i correspond to all feasible rosters for nurse i .

Informally speaking, the idea of B&P is to have the subproblems act as pattern generators guided by the values of the dual variables associated with the LP solution to a restricted \mathcal{MP} , i.e., one that contains only a subset of feasible rosters. At each major iteration, an optimality check is made by pricing out \mathcal{MP} . This is done by solving the subproblems to see whether one or more rosters (\mathcal{MP} columns) can be identified that have a negative reduced cost. When added to the restricted \mathcal{MP} , such columns will improve the current LP solution, which serves as a lower bound on the solution to the original IP. If all the reduced costs are nonnegative and the \mathcal{MP} variables are nonintegral, B&B is performed. The approach is most effective when the LP solution to the restricted \mathcal{MP} is found in the early stages of column generation.

To construct the restricted \mathcal{MP} , it is necessary to translate the demand constraints (1b) from the shift-view formulation to a set-covering-type formulation. This requires some additional notation. Let $K(i)$ be a subset of feasible rosters for nurse i , let ζ_{ik} be a binary decision variable equal to 1 if roster k is selected for nurse i and 0 otherwise, and let X_{idt}^k be a parameter equal to 1 if roster k for nurse i covers shift t on day d and 0 otherwise. The restricted \mathcal{MP} is as follows:

$$\theta_{\text{IP}} = \text{Minimize} \sum_{i \in N} \sum_{k \in K} c_{ik} \zeta_{jk} + \sum_{d \in D} \sum_{t \in T} M_t y_{dt} \quad (2a)$$

$$\text{subject to} \sum_{i \in N} \sum_{k \in K(i)} X_{idt}^k \zeta_{ik} - s_{dt} + y_{dt} = LD_{dt},$$

$$\forall d \in D, \quad t \in T \quad (2b)$$

$$\sum_{k \in K(i)} \zeta_{ik} = 1, \quad \forall i \in N \quad (2c)$$

$$0 \leq s_{dt} \leq UD_{dt} - LD_{dt}, \quad 0 \leq y_{dt} \leq O_{dt}^{\max}, \\ \forall d, t; \quad \zeta_{ik} \in \{0,1\}, \quad \forall j, k. \quad (2d)$$

The parameter X_{idt}^k is a specific value of the decision variable x_{idt} in the original model (1a)–(1r) and hence links the shift-view formulation to the set-covering formulation. Next, we have to specify how the penalty associated with roster k for nurse i (denoted by c_{ik}) is calculated. In general, the relationship is $c_{ik} = \sum_{a=1}^{V_{\max}} r_a v_{ia}$. However, the violation variable v_{ia} is not present in \mathcal{MP} so the value of c_{ik} must be computed independently for each column. To complete the formulation, we have the assignment constraints (2c), which ensure that exactly one roster is selected per nurse. When $K(i)$ contains the full set of feasible rosters for each nurse i , model (2) is called the *full master problem* and is equivalent to (1).

In the decomposition, the subproblem constraints are identical to (1c)–(1p) in the shift-view model, where the values of the parameters H_i , P_{it} , W_i^{\max} , and D_i^{\max} depend on the individual rotational profiles. The objective function of each subproblem is designed to identify “promising” columns for \mathcal{MP} and is formulated as a generic reduced cost. If the dual variables for the demand constraints (2b) and the assignment constraints (2c) are denoted by μ_{dt} and σ_i , respectively, then for each nurse i , the reduced cost for column k in \mathcal{MP} is

$$\bar{c}_{ik} = c_{ik} - \boldsymbol{\pi} \mathbf{A}_{ik} = \sum_{a=1}^{V_{\max}} r_a v_{ia} - \sum_{d \in D} \sum_{t \in T} \mu_{dt} X_{idt}^k - \sigma_i, \quad (3)$$

where $\boldsymbol{\pi} = (\boldsymbol{\mu}, 0, 0, \dots, \sigma_i, 0, \dots, 0)$ and \mathbf{A}_{ik} is the ik th column associated with (2b) and (2c). When the restricted \mathcal{MP} is optimized, all the reduced costs should be nonnegative, indicating that dual feasibility of model (2) has been attained. To determine whether this condition holds, we drop the symbol k and minimize (3) subject to (1c)–(1r) for each nurse $i \in N$ with X_{idt}^k replaced with x_{idt} . If the solution, call it (\bar{x}_{idt}) , gives a value $\bar{c}_i < 0$, then the existing columns in \mathcal{MP} do not contain the optimal set of rosters. The column associated with (\bar{x}_{idt}) is then added to \mathcal{MP} , which is resolved to get a new set of dual variables. The process is repeated until $\bar{c}_i = 0$ for all i .

Single Aggregation

Before writing out the full model, it is advantageous take into account subproblem symmetry. Because nurses are differentiated by their rotational profiles only in (1), those

with identical profiles will have identical subproblems. This means that if nurse i_1 and nurse i_2 have the same profile, then their solutions will be interchangeable so there is no need to solve both subproblems. To deal with this situation, we aggregate identical nurses into a single subproblem and redefine the decision variables in \mathcal{MP} to be integer rather than binary. The subproblem variables remain binary. For \mathcal{MP} , let j be the index for rotational profiles and define a new integer variables z_{jk} such that $z_{jk} = \sum_{i \in N_j} \zeta_{ik}$, where N_j is the set of nurses with profile j .

Recall that a rotational profile includes both the shift types and the ratio (minimum percentage of days that must be assigned to each shift), so a D/E profile with a 40% ratio is different than a D/E profile with a 20% ratio. With this in mind, we now present the single aggregation master problem and the subproblems.

Notation

j	index for rotational profiles
k	index for rosters
c_{jk}	penalty coefficient for roster k and rotational profile j
$K(j)$	set of alternative rosters for rotational profile j
N^P	set of all rotational profiles; $n^P = N^P $
n_j^R	total number of nurses with rotational profile j
X_{jdt}^k	parameter equal to 1 if roster k associated with rotational profile j covers shift t on day d , 0 otherwise
Z_{jk}	(decision variable) number of nurses with profile j that are assigned roster k

Master Problem

$$\theta_{IP} = \text{Minimize} \sum_{j \in N^P} \sum_{k \in K(j)} c_{jk} z_{jk} + \sum_{d \in D} \sum_{t \in T} M_t y_{dt} \quad (4a)$$

$$\text{subject to} \sum_{j \in N^P} \sum_{k \in K(j)} X_{jdt}^k z_{jk} - s_{dt} + y_{dt} = LD_{dt}, \\ d \in D, \quad t \in T \quad (4b)$$

$$\sum_{k \in K(j)} z_{jk} = n_j^R, \quad j \in N^P \quad (4c)$$

$$0 \leq s_{dt} \leq UD_{dt} - LD_{dt}, \quad 0 \leq y_{dt} \leq O_{dt}^{\max}, \\ \forall d, t; \quad z_{jk} \geq 0 \text{ and integer}, \quad \forall j, k \quad (4d)$$

Subproblem j

$$\theta_j^{\text{SP}} = \text{Minimize} \sum_{a=1}^{V_{\max}} r_a v_{ja} - \sum_{d \in D} \sum_{t \in T} \mu_{dt} x_{jdt} - \sigma_j \quad (5a)$$

$$\text{subject to} \sum_{d \in D} x_{jdt} \geq P_{jt}, \quad t \in T_j \quad (5b)$$

$$\sum_{d \in D} \sum_{t \in T_j} h_r x_{jdt} = H_j \quad (5c)$$

$$\sum_{t \in T_j} x_{jdt} \leq 1, \quad d \in D \quad (5d)$$

$$x_{jdt_2} + x_{j,d+1,t_1} \leq 1, \quad d \in D \quad (5e)$$

$$\sum_{k=d}^{d+D_j^{\maxon}} \sum_{t \in T_j} x_{jdt} \leq D_j^{\maxon}, \quad d \in D \quad (5f)$$

$$\sum_{d \in D_w} \sum_{t \in T_j} x_{jdt} \geq W_j^{\max} w_{jm}, \quad m \in W \quad (5g)$$

$$\sum_{m \in W} w_{jm} = 1 \quad (5h)$$

$$\sum_{t \in T_j} x_{jdt} + \left(1 - \sum_{t \in T_j} x_{j,d+1,t}\right) + \sum_{t \in T_j} x_{j,d+2,t} + P_{jd} \geq 1, \quad d \in D \quad (5i)$$

$$\left(1 - \sum_{t \in T_j} x_{jdt}\right) + \sum_{t \in T_j} x_{j,d+1,t} + \left(1 - \sum_{t \in T_j} x_{j,d+2,t}\right) + q_{jd} \geq 1, \quad d \in D \quad (5j)$$

$$1 - x_{jdt_\alpha} + 1 - x_{j,d+1,t_\beta} + b_{jd} \geq 1, \quad d \in D, \quad \alpha \neq \beta \in \{1,2\} \quad (5k)$$

$$\sum_{d \in D} b_{jd} \leq TR_{\max} \quad (5l)$$

$$\sum_{d \in D} (p_{jd} + q_{jd} + b_{jd}) = \sum_{a=1}^{V_{\max}} a v_{ja} \quad (5m)$$

$$\sum_{a=1}^{V_{\max}} v_{ja} \leq 1 \quad (5n)$$

$$b_{jd}, p_{jd}, q_{jd} \geq 0 \quad \forall d, t; \quad v_{ja} \in \{0,1\}, \quad \forall i, a; \quad w_{jm} \in \{0,1\}, \quad \forall m \quad (5o)$$

$$x_{jdt} \in \{0,1\}, \quad \forall t, d, \quad \text{where } x_{j,14+l,t} \equiv x_{jlt}, \quad l = 1, \dots, D_j^{\maxon} \quad (5p)$$

In (4), the modified demand constraints are now based on the rotational profile j instead of the individual nurse i , and

constraints (4c) are a generalization of the assignment constraints (2c). To complete the formulation, bounds on the slack variables and gap variables are introduced in (4d). Finally, model (5b)–(5p) is the same as (1c)–(1r) except that the variables and parameters are now based on rotational profiles (j) instead of individual nurses (i).

One disadvantage of the aggregation approach is that we lose the ability to impose limits on soft constraint violations for individual nurses, such as those specified in (1m). The best that we can do, in general, is to limit the number of profiles that incur these violations by adjusting TR_{\max} in (5l), placing upper bounds on p_{jd} and q_{jd} in (5o), or with respect to \mathcal{MP} , limit the number of profiles of a certain type by placing bounds on the variables z_{jk} in (4d). It is also possible to single out those nurses with unique constraint sets and define an individual rotational profile for each.

4. SOLUTION METHODOLOGY

B&P is similar to B&B when lower bounds are computed by solving an LP relaxation. A search tree is set up and at each node κ , the LP relaxation of the original problem, as modified by the branching constraints, is solved. The optimal solution value, call it θ_{LP}^κ , is then compared with the incumbent IP solution $\bar{\theta}_{IP}$ (usually obtained with a heuristic), and when the percentage difference between the two is within some tolerance $\epsilon\%$, the corresponding node is fathomed. Fathoming also occurs when the current LP is infeasible or when its solution is integral. In depth-first search, unfathomed nodes are partitioned by identifying branching constraints, usually in the form of variable bounds, and creating two successor nodes. In the context of B&P, the relaxed solution is obtained at every node by solving the restricted \mathcal{MP} with D-W decomposition. The branching strategies may be different than those common to B&B, but they share the same goal of trying to explore the feasible region by solving a series of more restricted problems. In the worst case, it would be necessary to generate all feasible columns to (4a)–(4d) as the search tree is enumerated and the algorithm iterates between \mathcal{MP} and the subproblems.

In our experience, the success of B&P depends on two critical features: the branching strategy and the use of an efficient heuristic to find high quality feasible solutions. Each is discussed below.

4.1. Branching Strategy

Upon termination of D-W at the root node, the optimal solution $(\theta_{LP}^0, \mathbf{z}^0, \mathbf{s}^0, \mathbf{y}^0)$ obtained by solving the LP relaxation of \mathcal{MP} is not likely to be integral even though the subproblems (5a)–(5p) are treated as IPs. Therefore, some amount of branching is inevitable. It is well known that the sole application of standard B&B to the restricted \mathcal{MP} is not

sufficient to obtain the optimal solution to the original problem because columns not in the restricted \mathcal{MP} may in fact be part of the optimal IP solution [7].

The widely applied branching rule in B&P is to create a search tree by fixing one of the subproblem variables at each node. Jaumard, Semet and vovor [27] applied this rule to a midterm preference scheduling problem in which there was one subproblem per nurse. Because of the relationship between the subproblem variables x_{idt} and the parameters X_{idt}^k in their set-covering formulation, the rule only required a simple modification of \mathcal{MP} . An \mathcal{MP} with binary decision variables means that fixing subproblem variables is equivalent to selecting columns with the implied properties. The assignment constraints (2c) in \mathcal{MP} are then altered so that columns whose properties do not conform to the fixed variables in the subproblem are removed.

The difficulty that we face with such an approach is that (4a)–(4d) contains integer rather than binary variables, so fixing a subproblem variable x_{jdt} at 1 or 0 does not translate into fixing or removing a column. For rotational profile j , for example, removing column k would mean that the associated roster could not be assigned to any nurse with that profile, a much too restrictive condition.

To develop appropriate rules, we need to revisit the physical interpretation of the integer variables z_{jk} in \mathcal{MP} . Because z_{jk} is an aggregation of several binary variables, i.e., $z_{jk} = \sum_{i \in N_j} \zeta_{ik}$, a fractional value $z_{jk} = 2.5$ means that 2 nurses with profile j can be assigned the same roster k and at least 1 nurse with the same profile does not have an assignment. The idea then is to partition the feasible region so that all nurses get an assignment. Two different rules are proposed. The first is a modified version of subproblem variable branching while the second is based on the master problem variables.

Branching Based on Subproblem Variables

To describe this rule, we start with the relationship between the subproblem variables x_{jdt} and the master problem parameters X_{jdt}^k . Given a fractional solution z_{jk} to \mathcal{MP} , there exists at least 1 day and shift, say day d_1 and shift t_1 , and corresponding constraint (4b) with a fractional value, call it $\beta_{jd_1t_1}$, for rotational profile j ; that is,

$$\beta_{jd_1t_1} \equiv \sum_{k \in K(j)} X_{jd_1t_1}^k z_{jk} \notin \mathbb{Z}^1. \quad (6)$$

Because a fractional z_{jk} can be greater than 1, a generalized branching constraint is to force β_{jdt} to be integral. In extending the search tree from the current node, we follow the common depth-first approach of creating two successor nodes defined by the constraints

$$\sum_{k \in K(j)} X_{jd_1t_1}^k z_{jk} \leq \lfloor \beta_{jd_1t_1} \rfloor, \quad \sum_{k \in K(j)} X_{jd_1t_1}^k z_{jk} \geq \lceil \beta_{jd_1t_1} \rceil, \quad (7)$$

where $\lfloor \beta \rfloor$ denotes the greatest integer less than or equal to β and $\lceil \beta \rceil$ denotes the smallest integer value greater than or equal to β .

In the first case, we need to prevent subproblem j from selecting any rosters that would result in $X_{jd_1t_1}^k = 1$ in \mathcal{MP} . To implement this restriction, we set the variable $x_{jd_1t_1} = 0$. With respect to \mathcal{MP} , the left-hand constraint in (7) means that at most $\lfloor \beta_{jd_1t_1} \rfloor$ nurses with profile j can be assigned a roster with $X_{jd_1t_1}^k = 1$. For the complementary constraint in (7), we set $x_{jd_1t_1} = 1$ in subproblem j .

The branching constraints that are added to \mathcal{MP} introduce new dual variables that must be taken into account in the pricing subproblem objective function (5a). Let us denote these nonnegative variables by τ_{jl}^L and τ_{jl}^R depending on whether we are referring to the left (L) or right (R) branch, where l is the level of the search tree. By noting that the summation term in Eq. (6) can be rewritten as $\sum_{k \in K(j)} X_{jd_1t_1}^k z_{jk} = \sum_{k: X_{jd_1t_1}^k = 1} z_{jk}$, the inequalities in (7) become $\sum_{k: X_{jd_1t_1}^k = 1} z_{jk} \leq \lfloor \beta_{jd_1t_1} \rfloor$ and $\sum_{k: X_{jd_1t_1}^k = 1} z_{jk} \geq \lceil \beta_{jd_1t_1} \rceil$, so the objective function (5a) of the updated pricing subproblem is

$$\sum_{a=1}^{V_{\max}} r_a v_{ja} - \sum_{d \in D} \sum_{t \in T} \mu_{dt} x_{jdt} + \sum_{l \in L(j)} (\tau_{jl}^L - \tau_{jl}^R) - \sigma_j. \quad (8)$$

In (8), $L(j)$ is the set of levels in the search tree at which branching constraints are imposed on profile j . The constraints in (7) can be viewed as master problem cuts and are similar to the cover cuts used by Barnhart, Hane, and Vance [6].

Example. To illustrate the proposed branching strategy, suppose that rotational profile 5 consists of 10 nurses who are to be assigned ten 8-hour evening shifts over the next 14 days, i.e., $n_5 = 10$ and rotational profile 5 = (E, 100%, 80 hours). Assume that after running D-W at the root node, the optimal LP solution contains 13 columns for profile 5 with the following five fractional values:

- $z_{5,1} = 3.753$, with assignments on days 2, 3, 4, 5, 8, 9, 10, 11, 13, 14
- $z_{5,4} = 2.032$, with assignments on days 1, 2, 3, 5, 6, 7, 8, 10, 11, 12
- $z_{5,6} = 1.903$, with assignments on days 2, 3, 4, 5, 8, 9, 10, 12, 13, 14
- $z_{5,12} = 2.188$ with assignments on days 1, 3, 4, 5, 8, 9, 11, 12, 13, 14
- $z_{5,13} = 0.124$ with assignments on days 1, 2, 3, 4, 6, 7, 8, 10, 11, 12.

Thus, $K(5) = \{1, \dots, 13\}$ but only the z_{5k} variables associated with the columns 1, 4, 6, 12, and 13 have nonzero

values. The physical interpretation is that $(\lfloor 3.753 \rfloor + \lfloor 2.032 \rfloor + \lfloor 1.903 \rfloor + \lfloor 2.188 \rfloor + \lfloor 0.124 \rfloor) = 8$ nurses have been assigned rosters, leaving 2 unassigned. If we arbitrarily pick $x_{5,1,1}$ as the branching variable, then by setting $x_{5,1,1} = 0$ in subproblem 5, we create the first successor node in the search tree. The implication for \mathcal{MP} is that any rosters now selected for the 2 unscheduled nurses with profile 5 should not include shift 1 on day 1. This is implemented by adding the constraint $z_{5,4} + z_{5,12} + z_{5,13} \leq 4$ to \mathcal{MP} . Also, the new dual variable $\tau_{5,1}^L$ is now included in the objective function (8) of subproblem 5. To create the second successor node, we set $x_{5,1,1} = 1$ and modify subproblem 5 accordingly. We then add the branching constraint $z_{5,4} + z_{5,12} + z_{5,13} \geq 5$ to \mathcal{MP} . Its dual variable is denoted by $\tau_{5,1}^R$.

Branching Based on Master Problem Variables

Given a fractional solution \hat{z}_{jk} to \mathcal{MP} at node κ , an obvious branching strategy is to enforce $z_{jk} \leq \lfloor \hat{z}_{jk} \rfloor$ and $z_{jk} \geq \lceil \hat{z}_{jk} \rceil$ at successor nodes $\kappa + 1$ and $\kappa + 2$, respectively. The difficulty with this strategy is in imposing the constraints implied by the first bound on the corresponding subproblem. In particular, when node $\kappa + 1$ is created by adding the constraint $z_{jk} \leq \lfloor \hat{z}_{jk} \rfloor$ to \mathcal{MP} , the feasible region of subproblem j must be modified to ensure that the point (X_{jdt}^k) associated with column k is infeasible. To do this, let $S(k)$ be the set containing the day–shift combinations associated with column k and add the following “schedule elimination constraint” to subproblem (5a)–(5p):

$$\sum_{(d,t) \in S(k)} x_{jdt} \leq \alpha_{jk} - 1, \quad \text{where } \alpha_{jk} = \sum_{d \in D} \sum_{t \in T} X_{jdt}^k. \quad (9)$$

To see why (9) only cuts off the roster associated with column k , observe that the working hours constraint (5c) is written as an equality, implying that all feasible rosters contain exactly α_{jk} shifts regardless of the ratio in (5b). Therefore, those shifts cannot be a subset of any other feasible roster. For the right branch $z_{jk} \geq \lceil \hat{z}_{jk} \rceil$, no modification of subproblem j is needed because column k will not be a solution. This follows from linear programming theory, as discussed in the proof below.

Proposition 1. Branching on fractional values of z_{jk} in \mathcal{MP} leads to a complete enumeration of the feasible region of the original problem (1) when constraint (9) is added to subproblem j when the search tree is extended to include the branching constraint $z_{jk} \leq \lfloor \hat{z}_{jk} \rfloor$.

Proof. If we are at node κ and create the branch associated with $z_{jk} \leq \lfloor \hat{z}_{jk} \rfloor$, it is well known that this constraint will be binding at node $\kappa + 1$ and that its dual variable will be negative. This means that the objective function would

decrease if the solution corresponding to (X_{jdt}^k) were feasible; hence, the need to exclude it. Because (9) removes only one roster at a time, all remaining rosters are feasible to subproblem j .

On the second branch leading to node $\kappa + 2$, the constraint $z_{jk} \geq \lceil \hat{z}_{jk} \rceil$ will be binding in the LP solution and its dual variable will be positive. Therefore, there is no need to exclude the solution associated with (X_{jdt}^k) . The strategy of creating two successor nodes with constraints $z_{jk} \leq \lfloor \hat{z}_{jk} \rfloor$ and $z_{jk} \geq \lceil \hat{z}_{jk} \rceil$, respectively, does not exclude any integer feasible solutions to (4). As long as solutions exist with fractional z_{jk} , though, branching will continue and in the worst case, all feasible columns and all values of z_{jk} will be enumerated. \square

With master problem variable branching, more and more schedule elimination constraints must be added to the subproblems as the search tree grows. This increases their difficulty and thus limits the suitability of the approach to situations in which good feasible solutions emerge early on. Nevertheless, fixing \mathcal{MP} variables complements the IP rounding heuristic that we have developed to find upper bounds (see next subsection). The heuristic is based on an IP obtained from \mathcal{MP} and becomes easier to solve as the bounds in (7) are added to \mathcal{MP} .

Example (continued). To illustrate the master problem variable branching strategy, suppose that in the above example $z_{5,1} = 3.753$ is selected as the branching variable. Two nodes are created, the first with $z_{5,1} \leq 3$ and the second with $z_{5,1} \geq 4$. In the first case, constraint (9) must be added to subproblem 5. From the solution obtained for $z_{5,1}$, we get $\alpha_{5,1} = 10$, so the corresponding schedule elimination constraint is $x_{5,2,1} + x_{5,3,1} + x_{5,4,1} + x_{5,5,1} + x_{5,8,1} + x_{5,9,1} + x_{5,10,1} + x_{5,11,1} + x_{5,13,1} + x_{5,14,1} \leq 9$. In the second case, no modifications to subproblem 5 are necessary.

4.2. Heuristics

An essential component of B&B is a complementary algorithm for finding good feasible solutions. Not only do such solutions increase the frequency with which unexplored nodes are fathomed, thus reducing the size of the tree, but should it be necessary to terminate the enumeration process before convergence, they also provide an implementable course of action.

Any heuristic strategy selected, however, must be computationally efficient and make intelligent use of fractional solutions if it is to be an effective component of B&B. Metaheuristics have proven to be quite powerful when applied to a wide range of relatively pure combinatorial optimization problems, but in our experience, their neighborhood search logic is difficult to adapt to problems with unstructured technological constraints.

With set-covering-type formulations, a practical approach is to use rounding heuristics [8]. Given the relaxed \mathcal{MP} solution $(\theta_{LP}^k, \hat{z}^k, s^k, y^k)$ at node κ , one possible approach is to try to solve \mathcal{MP} as an IP. By including only a subset of “good” columns in the IP rather than all columns and fixing those fractional variables that are within some tolerance γ of integrality, a high-quality feasible solution to (1) may be obtained quickly. The key to success is the variable fixing process. A tight fixing rule may render the IP infeasible, while a loose rule may lead to excessive computation times.

In our application, we use a parameter γ to guide the rounding procedure. The complete algorithm is given below.

Algorithm_IP_Rounding

Input: \mathcal{MP} and relaxed solution $\{\hat{z}_{jk}: \forall j \in N^p, k \in K(j)\}$ at some node in the search tree; fractional tolerance γ

Output: Feasible integer solution σ^{feas} to cyclic preference scheduling problem

Step 1: For all variables $z_{jk}, j \in N^p, k \in K(j)$

If $(\hat{z}_{jk} - \lfloor \hat{z}_{jk} \rfloor) \leq \gamma$, then fix $z_{jk} = \lfloor \hat{z}_{jk} \rfloor$ in \mathcal{MP}

If $(\lceil \hat{z}_{jk} \rceil - \hat{z}_{jk}) \leq \gamma$, then fix $z_{jk} = \lceil \hat{z}_{jk} \rceil$ in \mathcal{MP}

If $\hat{z}_{jk} > 1$, then fix $z_{jk} = \lfloor \hat{z}_{jk} \rfloor$ in \mathcal{MP}

Step 2: Solve the modified \mathcal{MP} (4a)–(4d) as an IP with the remaining non-fixed variables to get \hat{z}_{jk}^* .

Step 3: Convert solution to a set of rosters σ^{feas} as follows. For all $\hat{z}_{jk}^* > 0, j \in N^p, k \in K(j)$ put $\sigma_{jk}^{\text{feas}} \leftarrow \{X_{jdt}^k, \forall d \in D, t \in T\}$, where $\sigma^{\text{feas}} = (\sigma_{jk}^{\text{feas}}, j \in N^p, k \in K(j))$.

In the first two tests at Step 1, the \mathcal{MP} variables that are within γ of integrality are rounded to the closest integer and fixed. In the third test, all fractional variables not yet fixed but greater than 1 (that is, all rosters with more than one assigned nurse) are rounded down and fixed. This is consistent with the physical interpretation of \hat{z}_{jk} in the definition of the branching rule. An Step 2, the resultant set-covering-type problem (4) is solved as an IP. The non-fixed variables in this problem include those in the range $\gamma < \hat{z}_{jk} \leq 1 - \gamma$ and those whose values were 0 in the LP solution of \mathcal{MP} . Although it was never the case in our experiments, if columns that are included in the IP are not sufficiently varied, or, as mentioned in Section 3.1, if the bounds in (4d) on s_{dt} and y_{dt} are too tight, then (4a)–(4d) may be infeasible. In Step 3, the IP solution is converted to a feasible schedule, σ^{feas} .

Example (continued). Applying the rounding heuristic to the master problem solution obtained with $\gamma = 0.2$ in the above example, we get $z_{5,1} = 3, z_{5,4} = 2, z_{5,6} = 2, z_{5,12} = 2$ and $z_{5,13} = 0$ at Step 1. Thus, a total of nine rosters have been assigned to profile 5. The remaining roster will be determined at step 2 and will be associated with one of the following non-fixed variables: $z_{5,2}, z_{5,3}, z_{5,5}, z_{5,7}, z_{5,8}, z_{5,9}, z_{5,10}, z_{5,11}$.

4.3 Enhancement Options

Early Termination

In B&P implementations, it is common for a tailing off effect to occur when solving \mathcal{MP} . The effect becomes evident when more and more columns are added but no significant improvement in the objective function is realized. One way to combat this problem is to compute a dual lower bound on the LP solution and use it to terminate column generation when an acceptable gap is reached. Wolsey [41] gives a dual feasible solution to \mathcal{MP} that can be used for this purpose. For our problem, the bound is

$$\theta_{LP} \geq \sum_{d \in D} \sum_{t \in T} \mu_{dt} LD_{dt} + \sum_{j \in N^p} \sigma_j - \sum_{d \in D} \sum_{t \in T} (UD_{dt} - LD_{dt}) \eta_{dt}^1 - \sum_{d \in D} \sum_{t \in T} O_{dt}^{\max} \eta_{dt}^2 + \sum_{j \in N^p} \theta_j^{\text{SP}} = \underline{\theta}_{LP}, \quad (10)$$

which is valid at any D–W iteration and at any node in the search tree.

The first four terms on the right-hand side of the inequality in (10) represent the objective function of the dual of \mathcal{MP} , where $\eta_{dt}^1 \geq 0$ and $\eta_{dt}^2 \geq 0$ are the dual variables associated with the bound constraints on s_{dt} and y_{dt} , respectively. The last term is the sum of the subproblem objective functions (5a). At each D–W iteration, the LP relaxation of \mathcal{MP} is solved to optimality for the current set of columns and the dual variables are passed to the subproblems. Assuming that \mathcal{MP} is bounded, strong duality implies that the dual objective function value in (10) can be replaced with the corresponding LP solution, so we don’t actually have to calculate it, i.e., (10) can be written as $\theta_{LP} \geq \bar{\theta}_{LP} + \sum_{j \in N^p} \theta_j^{\text{SP}} = \underline{\theta}_{LP}$, where $\bar{\theta}_{LP}$ is the LP solution of \mathcal{MP} at any D–W iteration.

With a known lower bound, $\underline{\theta}_{LP}$, we can now establish a procedure for terminating D–W before all subproblems price out nonnegatively. The approach that we take is based on the percentage difference between $\bar{\theta}_{LP}$ and $\underline{\theta}_{LP}$, where, for a particular percentage parameter ρ , the subproblem computations are terminated when $[(\bar{\theta}_{LP} - \underline{\theta}_{LP})/\underline{\theta}_{LP}] \times 100\% \leq \rho$, that is, when $(\sum_{j \in N^p} \theta_j^{\text{SP}}/\underline{\theta}_{LP}) \times 100\% \leq \rho$. Using this test, one can expect a smaller search tree and a faster solution to \mathcal{MP} , but at a cost in overall solution

quality. In addition, when $\underline{\theta}_{LP}$ (rather than $\bar{\theta}_{LP}$) is used for fathoming, the solution obtained when the B&P algorithm terminates may differ from true IP optimum, θ_{IP} , by ρ or more, depending on the stopping criteria.

Because \mathcal{MP} is not optimized with this approach, the value of the LP solution at early termination may be greater than θ_{IP} . Moreover, had D–W been allowed to continue, the current node may have been fathomed by bounds. To compensate for this loss in precision, one can adaptively set ρ , with smaller values being used at lower levels in the search tree.

Double Aggregation

Models (4) and (5) treat each rotational profile separately. When two such profiles include the same shifts but different ratios, the corresponding subproblems (5) only differ by the value of P_{jt} in constraint (5b). For example, consider a 20% D/E profile and a 40% D/E profile indexed by j_1 and j_2 , respectively, and assume that nurses working these profiles must be assigned a total of 80 hours or 10 shifts over a 2-week period. In the first case, $P_{j_1t} = 2$ for $t \in \{D, E\}$ and in the second case, $P_{j_2t} = 4$ for $t \in \{D, E\}$. In our original formulation, these two profiles are viewed as two separate subproblems. The fact that subproblem j_1 with a 20% ratio may give a solution that satisfies subproblem j_2 with a 40% ratio argues for combining the two into a single subproblem. Potential benefits include reducing the overall computation time and eliminating symmetry.

Combining two or more rotational profiles with the same shifts and total hours but different ratios requires modifications to both the master problem and the associated subproblem. The new subproblem will use the least restrictive ratio, which, in the previous example, is 20%. To guarantee that rosters with more restrictive ratios are generated as well, it is necessary to augment the assignment constraints (4c) in the master problem with one assignment-type constraint for each ratio above the least restrictive. To develop the new constraints, let $R(j)$ be the set of ratios to be aggregated into a single subproblem j , let P_{jt}^r be the corresponding lower bound on the number of shifts of type t that must be assigned for each $r \in R(j)$, let $K_r(j)$ be the subset of alternative rosters k for profile j that satisfy the ratio constraints (5b) with a given parameter P_{jt}^r , $r = 1, \dots, |R(j)| - 1$ [i.e., $K_r(j) = \{k: \sum_{d \in D} X_{jdt}^k \geq P_{jt}^r, \forall t \in T_j\} \subset K(j)$], and let n_r be the number of nurses whose ratio corresponds to index r . With this notation, we now redefine a rotational profile by the triplet (eligible shifts, $R(j)$, total hours).

Assuming that the elements of $R(j)$ are sorted from the most restrictive to the least restrictive ratio, then the following constraints, when added to \mathcal{MP} , ensure that all ratio requirements are met for aggregate profile j .

$$\sum_{k \in K_r(j)} z_{jk} \geq \sum_{p=1}^r n_p, \quad r = 1, \dots, |R(j)| - 1 \quad (11a)$$

$$\sum_{k \in K(j)} z_{jk} = \sum_{r \in R(j)} n_r \quad (11b)$$

When $r = 1$, constraint (11a) guarantees that a solution to \mathcal{MP} includes at least n_1 rosters that contain P_{jt}^1 shifts of type t for $t \in T_j$. When $r = 2$, this constraint ensures that a solution contains at least P_{jt}^1 and P_{jt}^2 shifts of type t for profile j . For the final element of $R(j)$, we have (11b), which replaces the original assignment constraints (4c) and ensures that exactly one roster is selected for each nurse with profile j .

The next step is to modify the pricing subproblem j to take into account (11a) and (11b). Let σ_{jr} , $r = 1, \dots, |R(j)| - 1$, be the dual variables for the $|R(j)|$ new constraints, where all but $\sigma_{j,|R(j)|}$ are nonnegative, and let δ_{jr} be a binary variable equal to 1 if the ratio associated with index r is selected when subproblem j is solved and 0 otherwise. We now replace the objective function in (5a) with the following:

$$\sum_{a=1}^{V_{\max}} r_a v_{ja} - \sum_{d \in D} \sum_{t \in T} \mu_{dt} x_{jdt} - \sum_{r=1}^{|R(j)|-1} \sigma_{jr} \delta_{jr} - \sigma_{j,|R(j)|}. \quad (5a')$$

To understand how the term associated with the new variables in (5a') arises, consider what happens when subproblem j is solved under double aggregation. Because all the ratios are now aggregated in the subproblem, the objective function must identify which ratio is being priced out. If we wish to price out the case that corresponds to the least restrictive ratio (i.e., $r = |R(j)|$), then only the dual variable $\sigma_{j,|R(j)|}$ associated with constraint (11b) in \mathcal{MP} is relevant. Therefore, all the other dual variables σ_{jr} , $r = 1, \dots, |R(j)| - 1$, should be removed from the subproblem objective function. At the other extreme, if we wish to price out the case that corresponds to the most restrictive ratio (i.e., $r = 1$), then all the dual variables σ_{jr} , $r = 1, \dots, |R(j)|$, should be included in the subproblem objective function. In actuality, we want the model, not us, to decide which case is being priced out. This can be achieved by adding the following constraints to subproblem j .

$$\sum_{d \in D} x_{jdt} \geq P_{jt}^r \zeta_{rt}, \quad \forall t \in T_j, \quad r = 1, \dots, |R(j)| - 1 \quad (12a)$$

$$\sum_{d \in D} x_{jdt} \leq P_{jt}^r - 1 + M \zeta_{rt}, \quad \forall t \in T_j, \quad r = 1, \dots, |R(j)| - 1 \quad (12b)$$

$$\sum_{t \in T_j} \zeta_{rt} - \delta_{jr} \leq 1, \quad r = 1, \dots, |R(j)| - 1 \quad (12c)$$

$$\delta_{jr} \leq \zeta_{rt}, \quad \forall t \in T_j, \quad r = 1, \dots, |R(j)| - 1 \quad (12d)$$

$$\delta_{jr}, \zeta_{rt} \in \{0, 1\}, \quad \forall t \in T_j, \quad r = 1, \dots, |R(j)| - 1 \quad (12e)$$

The binary variables δ_{jr} and ζ_{rt} together determine which case is active. When $\zeta_{rt} = 1$ for both $t \in T_j$, constraint (12a) requires $\sum_{d \in D} x_{jdt} \geq P_{jt}^r$ so all feasible points must satisfy the ratio associated with index r . For this situation to occur, δ_{jr} must also be 1, as seen in constraints (12c) and (12d). If $\delta_{jr} = 0$, then at most one ζ_{rt} can be 1 so at most one of the two shift types will satisfy the ratio requirement. When $\zeta_{rt} = 0$ for all $t \in T_j$, constraint (12b) is active and $\delta_{jr} = 0$ so at most $P_{jt}^r - 1$ shifts of type t can be included in a solution. Thus, when $\zeta_{rt} = 0$ for $r = 1, \dots, |R(j)| - 1$, we have the least restrictive case so only (5b) applies and only the dual variable $\sigma_{j,|R(j)|}$ will appear in the subproblem objective function (5a'). Note that in Eq. (12b), M is a large number.

The new formulation, which includes (11) in \mathcal{MP} and (12) in subproblem j , represents an increase in model complexity, but may give better bounds at each node in the search tree. More specifically, we have the following result.

Proposition 2. Let $\hat{\theta}_{LP}$ be the objective function value obtained by solving the double aggregation \mathcal{MP} as an LP. Then $\hat{\theta}_{LP} \geq \theta_{LP}$.

Proof. In the single aggregation (SA) model (4), there is one assignment-type constraint (4c) for each rotational profile. In the double aggregation (DA) model, there are $|R(j)|$ assignment-type constraints (11) for each rotational profile but the totals are the same, i.e., $|N^p| = \sum_j |R(j)|$. Thus, SA and DA have the same number of constraints, but the former has more columns due to the fact that the same rosters may be feasible to multiple subproblems.

To show that DA gives at least as good a bound as SA, we will compare the feasible regions of their duals. Consider two rotational profiles j_1 and j_2 that differ only by their ratios r_1 and r_2 , where the first is the more restrictive one. Let k_1 and k_2 be the column indices in \mathcal{MP} for these profiles, and let the vectors $\mathbf{X}_{j_1}^{k_1}$ and $\mathbf{X}_{j_2}^{k_2}$ be the corresponding columns associated with the demand constraints (4b). Noting that $\mathbf{X}_{j_1}^{k_1} = \mathbf{X}_{j_2}^{k_2} \equiv \mathbf{X}_j^k$ and that the associated objective function coefficients are equal, i.e., $c_{j_1 k_1} = c_{j_2 k_2}$, we can drop the index k and write dual constraints and objective function terms for these columns as

$$\boldsymbol{\mu} \mathbf{X}_j + \sigma_{j_1} \leq c_j, \quad n_{j_1}^R \sigma_{j_1} \quad (13a)$$

$$\boldsymbol{\mu} \mathbf{X}_j + \sigma_{j_2} \leq c_j, \quad n_{j_2}^R \sigma_{j_2} \quad (13b)$$

where σ_{j_1} and σ_{j_2} are unrestricted. For DA, the dual constraint and objective function term are

$$\boldsymbol{\mu} \mathbf{X}_j + \sigma_{j_1} + \sigma_{j_2} \leq c_j, \quad n_{j_1}^R \sigma_{j_1} + (n_{j_1}^R + n_{j_2}^R) \sigma_{j_2}, \quad (14)$$

where $\sigma_{j_1} \geq 0$, σ_{j_2} is unrestricted, and n_{r_1} and n_{r_2} in (11) have been replaced with their equivalents in (4c).

Because we are maximizing the dual objective function, we want σ_{j_1} and σ_{j_2} to be as large as possible in both SA and DA. From the constraints in (13), this implies that at optimality $\sigma_{j_1}^{SA} = \sigma_{j_2}^{SA}$, which allows us to write the objective function terms in (13) as $(n_{j_1}^R + n_{j_2}^R) \sigma_{j_2}^{SA}$. By noting that $\boldsymbol{\mu} = \boldsymbol{\mu}^{SA}$, $\sigma_{j_1} = 0$, $\sigma_{j_2} = \sigma_{j_2}^{SA}$ is feasible to (14) and that $\sigma_{j_1} \geq 0$ for DA, we have $n_{j_1}^R \sigma_{j_1}^{DA} + (n_{j_1}^R + n_{j_2}^R) \sigma_{j_2}^{DA} \geq n_{j_1}^R \sigma_{j_1}^{SA} + n_{j_2}^R \sigma_{j_2}^{SA}$ at optimality. By further noting that (i) all constraints in the duals of SA and DA that are not associated with aggregated variables in the primal are identical and (ii) the relationship between (13) and (14) can be extended to the case in which more than two profiles are aggregated, we can conclude that $\hat{\theta}_{LP} \geq \theta_{LP}$. \square

5. COMPUTATIONAL RESULTS

The B&P algorithm was implemented in Visual C++ and linked to CPLEX 7.5, which was used to solve the master problem LPs and the subproblem IPs at each D–W iteration. Two versions of the algorithm were created, one with single aggregation and the other with double aggregation. Both included the IP rounding heuristic, which was called in accordance with the strategy discussed below. The early termination procedure was included in the single aggregation version only and was activated with a user-supplied parameter.

To gauge performance, 25 problem instances of various sizes were generated based on data obtained from an intensive care unit in a 400-bed U.S. hospital. The unit has 80 full-time nurses. To create smaller instances, we randomly selected nurses from the original 80; to create large instances, we duplicated a random subset of the nurses and then increased the demand proportionately.

The computational experiments were aimed at determining the effectiveness of the two proposed branching strategies as well as the quality of the solutions provided by the rounding heuristic. In addition, we were interested in learning what the costs and benefits were of using the early termination procedure and whether the double aggregation scheme improved overall performance. A PC with a 1.1-GHz processor running Windows XP was used for the computations.

Table 1. Input characteristics for all problems instances.

Problem No.	No. of nurses	No. of subproblems	Demand (h)	Supply (h)	Allowed slack (h)	Allowed gap (h)
1	20	5	1344	1600	464	336
2	20	8	1504	1536	672	672
3	30	5	2206	2400	368	336
4	30	8	2240	2312	712	672
5	50	8	3276	3792	1024	672
6	50	12	3504	3840	896	672
7	50	15	4556	3840	1144	1354
8	80	10	6152	6112	1192	672
9	80	12	6264	6144	1192	1354
10	80	15	6248	6128	1192	672
11	80	20	6248	6128	1192	672
12	100	12	7484	7672	912	1354
13	100	15	7588	7680	672	672
14	100	18	7588	7696	784	672
15	100	20	7572	7672	882	672
16	100	25	7616	7720	806	672
17	150	15	11672	11520	784	1354
18	150	18	11672	11688	784	1354
19	150	20	11568	11704	784	672
20	150	25	11600	11664	784	672
21	150	30	11600	11680	784	672
22	200	15	14672	15480	1424	1354
23	200	20	14384	15424	1544	672
24	200	25	14640	15488	1424	1354
25	200	30	14832	15552	1256	672

Before presenting the results, several implementation issues need to be clarified. The first is the requirement to provide \mathcal{MP} with an initial feasible solution in the form of columns. We took a “big- M ” approach and started with artificial variables for the demand constraints (4b), one for each (d, t) pair. Simply setting $y_{dt} = LD_{dt}$ for all t and d may not work due to the upper bounds on y_{dt} in (4d). One artificial variable was also needed for each assignment constraint in (4c). Once initialized, D–W generates feasible columns as it iterates. Of course, if any artificial columns remain in the solution when \mathcal{MP} is optimized at node 0, then the original IP is infeasible.

The second issue concerns the rounding heuristic. At the root node of the search tree, after the restricted master problem is solved as an LP, an attempt is made to solve it as an IP using CPLEX. If the optimality gap between the best solution found within 60 seconds, call it θ_{IP}^{60} , and θ_{LP} is greater than 0.5%, then branching starts. The IP rounding heuristic is called every five nodes with the parameter γ adaptively set as follows: $\gamma = 0.3$ for nodes at levels ≤ 20 . For nodes between levels 21 and 60, $\gamma = 0.15$, and for levels > 60 , $\gamma = 0.05$. The smaller the value of γ , the fewer variables that are fixed in the IP so the better the solution is likely to be. Fewer fixed variables, though, means a more difficult problem so the choice of parameter settings represents a compromise.

The final issue concerns the branching rules. Initial testing showed that branching on the subproblem variables was superior to branching on the master problem variables for the smaller problem sets. Consequently, we adopted the former rule for all instances except those with 200 nurses, which could only be solved efficiently with the latter rule. Comparative results are provided for the 100- and 150-nurse problem sets. Depth-first search was used in the early nodes with the aim of finding good feasible solutions quickly. When the gap between the incumbent and the best LP solution dropped below 2%, we switched to a best-bound strategy. When this gap was $\leq 0.5\%$, the corresponding node was fathomed. In all cases, the search tree was limited to 100 nodes.

5.1. Problem Sets

Table 1 lists the basic characteristics of the problem sets used in the testing. The second column indicates the total number of nurses to be scheduled simultaneously. Recall that each unit in the hospital constructs its rosters separately. The total number of subproblems given in column 3 is equivalent to the total number of rotational profiles after single aggregation. The larger this number, the more difficult the problem is to solve. For profiles with only one or two shifts, the case here, 15 is the maximum number of

Table 2. Solution characteristic of all data sets.

Problem No.	No. violations	Surplus hours	Gap hours	Problem size ($m \times n$)	New columns	Initial LP soln	Initial gap (%)	IP soln	Time (s)	Total nodes
1	32	256	0	47×197	14	35	5.7	36	50	16
2	32	100	124	78×420	45	422	7.1	426	220	100
3	47	128	72	47×198	0	412	0	413	59	1
4	42	120	48	78×623	218	311	13.3	313	545	100
5	41	508	96	78×296	0	448	0	448	16	1
6	43	360	304	82×364	2	1498	7.3	1500	227	55
7	56	0	724	85×576	33	2549	6.3	2558	824	100
8	85	192	232	80×436	35	1016	3.9	1019	66	5
9	84	144	296	82×401	10	1338	3.5	1339	193	35
10	67	68	188	85×578	83	770	22.2	772	517	100
11	100	152	320	90×527	10	594	7.4	598	895	10
12	115	452	252	82×385	0	1180	0	1180	25	1
13	117	378	84	85×483	61	404	0.8	408	567	100
14	111	248	168	88×490	16	763	29.9	768	549	70
15	119	280	192	90×474	3	875	40.6	879	172	5
16	120	240	176	95×597	20	803	37.6	806	382	30
17	162	180	320	85×485	2	1391	22.7	1398	487	40
18	154	216	280	88×504	4	1241	36.8	1243	265	15
19	167	120	196	90×569	10	889	12.5	895	827	100
20	202	152	184	95×666	0	932	13.2	940	1166	100
21	182	112	128	100×739	15	927	24.6	932	1446	100
22	207	1024	408	85×631	146	1512	—	1514	1152	100
23	231	1028	176	90×649	61	878	55.2	883	1351	100
24	267	976	192	95×1069	393	990	19.9	999	1910	100
25	248	944	168	100×714	2	849	43.8	852	242	5

subproblems that are possible without considering ratios and total working hours. The eligible shifts for the subproblems in Table 1 were randomly generated from these 15. Ratios and total hours (either 72 or 80) were then assigned.

The demand, supply, allowed slacks (s_{dt}), and allowed gap (y_{dt}) are measured in hours and determine the tightness of a problem's feasible region. The total demand for all shifts over the 2-week planning horizon is given in column 4. The total supply is the sum of working hours for all nurses to be scheduled and is given in column 5. The allowed slack is the total number of surplus hours that can be assigned to a shift. In the demand constraints (4b), this is controlled by the upper bound on s_{dt} , which is $UD_{dt} - LD_{dt}$. The allowed gap is computed by summing the upper bound O_{dt}^{\max} on the gap variables over all shifts and days in the 2-week period.

For a fixed number of profiles, the problem instances in Table 1 are generally the most difficult that we could construct. Most instances with less tight combinations of demand, supply, allowed slack, and allowed gap were frequently solved at the root node and are not discussed in the paper.

5.2. Performance of the B&P Algorithm

The computational results for the 25 instances are presented in Table 2. Final roster quality is measured by the

total number of violations (column 2), the total number of surplus hours (column 3), and the total number of hours that must be covered with outside nurses (column 4). The total violations include the undesirable days-on and days-off patterns and the number of switches in shift assignments on consecutive days. Surplus hours occur when the number of nurses assigned to a shift exceeds the demand, giving rise to idle time. Similarly, the "gap hours" indicate the total amount of undercoverage that exists in the schedule for the upcoming 2 weeks. The number of surplus and gap hours should be no greater than the maximum hours shown in the last two columns of Table 1. Because of the complementary condition $s_{dt} \times y_{dt} = 0$, it is impossible to have a schedule in which the surplus and gap hours are both at their maximum values.

The remaining columns in Table 2 report on the efficiency of the algorithm as measured by the final problem size, the computation time, the initial percentage gap, and the IP solution. The problem size is given by the total number of rows and columns in \mathcal{MP} when the B&P algorithm terminates. The number of rows, m , is equal to (number of shift types) \times (number of days) + number of rotational profiles. Problems 1 and 3, for example, have three shift types (D, N, E) and five rotational profiles (D-only, N-only, E-only, D/E, and D/N), which leads to $3 \times 14 + 5 = 47$ constraints. Because no columns were deleted during the branching process, all columns generated during the

Table 3. Comparisons between master variable and subproblem variable branching.

Problem No.	Master variable branching				Subproblem variable branching			
	New columns generated	IP soln	Total nodes	Time (s)	New columns generated	IP soln	Total nodes	Time (s)
13	51	408	100	760	61	408	100	567
14	7	768	25	291	16	765	30	296
15	4	882	15	227	3	879	5	172
16	12	806	25	408	20	806	30	382
17	2	1395	25	338	2	1398	40	487
18	1	1243	5	93	5	1243	15	265
19	2	895	5	176	10	895	100	827
20	3	938	35	530	0	940	100	1166
21	5	930	5	272	15	932	100	1446

search are contained in these statistics. The number of columns generated beyond the root node is indicated under the “new columns” heading.

The value reported in the “initial gap” column is the percentage difference between the IP solution θ_{IP}^{60} obtained by trying to solve the restricted \mathcal{MP} as an IP at the root node with CPLEX and the initial LP solution θ_{LP} , i.e., $(\theta_{IP}^{60} - \theta_{LP}/\theta_{LP}) \times 100\%$. These values range from 0 to 55.2% with an average of 17.3%. In only instance 22 was CPLEX not able to find a feasible solution in 60 seconds. Column 9 labeled “IP soln” gives the objective function value of the best integer solution found at termination. The last two columns give the total computation time in seconds and the total number of nodes explored in the search tree.

Examining the output, we see that the average number of violations per nurse is less than 2 and relatively constant for all instances. In practice, this number depends on the tightness of the problem. The total slack and gap hours are significantly less than the maximum allowed. Another statistic that is relatively constant is size of the final \mathcal{MP} , which is about 500 columns. This is an extremely small number when compared to the total number of feasible rosters. The iterations at the root node appeared to have generated most of the “good” columns because very few new columns were added during the branching process. Only a few instances produced more than one column per node on average once branching began, with the largest being seven for instance 8. For the most part, only the fractional values of the master problem variables changed as branching progressed and not the number of variables in \mathcal{MP} .

Of the 25 instances, 11 reached the 100-node threshold, with the longest taking 1910 seconds, or nearly 32 minutes. Two were solved at the root node and only five required over 15 minutes. The majority of those were large, containing between 20 and 30 rotational profiles. Nevertheless, the best IP solutions found were always within 0.95% of their lower bound and most were within 0.1%.

The effectiveness of the rounding heuristic was evidenced by the quality of solutions it provided in the early

stages of the computations. In all instances that required branching, it found the best feasible solution reported while reducing the optimality gap from an average of 17.3% at the initial node to less than 1% at termination. In addition, an overwhelming number of nodes were fathomed by bounds, in part, due to the heuristic’s ability to uncover good feasible solutions after only a few B&P iterations.

It can also be seen from Table 2 that there is no strong correlation between problem size and difficulty. Although the trend is upward, many of the smaller instances took longer to solve than the larger ones. Problems with 50 nurses or less were easily solved within 10 minutes. Problems with more than 80 nurses were generally more difficult but there were some exceptions, such as instances 12 and 15, which took between 2 and 3 minutes.

As might be expected, the iterations at the root node took the longest time, averaging 40 seconds over all instances (results not shown). Somewhat surprisingly, there was little correlation between the initial gap and the degree of difficulty. For example, instance 7 had an initial gap of 6.2% but did not converge within the 100-node limit. In contrast, instance 25 had an initial gap of 43.8% but required only five nodes.

5.3. Comparative Analysis

Different Branching Strategies

With the exception of instances 22–25, the results in Table 2 were based on subproblem variable branching. To gauge the relative effectiveness of master problem variable branching, we resolved the 100- and 150-nurse instances 13–21 with this rule and collected the following data: number of new columns generated, IP solution, total number of nodes, and solution time.

The results are reported in Table 3 and demonstrate that subproblem variable branching is generally superior for the 100-nurse instances and master problem variable branching for the 150-nurse instances. In the case of the former, the results are comparable; in the case of the latter, master

Table 4. Lower bound performance for problem 13.

Iteration	LP solution	Lower bound	% difference
1	71120	-5675	1353.2
2	23624	-50770	146.5
3	9657.4	-52641	118.4
4	3713.5	-32200.2	111.5
5	1690.4	-19129.9	108.8
6	981.4	-8492.7	111.5
7	702.8	-2692.1	126.1
8	513.2	-2107.7	124.3
9	464.9	67.8	585.7
10	445.8	234.4	90.2
11	433.7	284.5	52.4
12	416.8	323.2	28.9
13	408.7	365.9	11.7
14	406.2	371.6	9.3
15	404.8	398.1	1.7
16	404.2	404.2	0

problem variable branching is able to find good feasible solutions quickly and hence limit the size of the search trees to at most 35 nodes. When subproblem variable branching was used, three of the five 150-nurse instances reached the 100-node limit.

Effectiveness of Dual Lower Bound

As the D-W procedure iterates, it is common to see a large initial gap between the LP objective function and the dual lower bound (10) that gradually converges to zero. For the problem sets that we investigated, the convergence rate was fast. This is illustrated by the computations reported in Table 4 for problem instance 13 and by the corresponding plots in Figure 1. The initial gap was about 1353% and remained above 100% for 9 iterations. It quickly shrank to 1.7% at iteration 15 and to 0 at iteration 16, with LP solutions of 404.8 and 404.2, respectively. Recall that the IP solution is 408. Using a 5% tolerance for terminating the computations would have only reduced the number of iterations by one; using a 1% tolerance would not have provided any reduction.

The early termination procedure was tested on the five 100-nurse instances using both a 5% and a 1% tolerance.

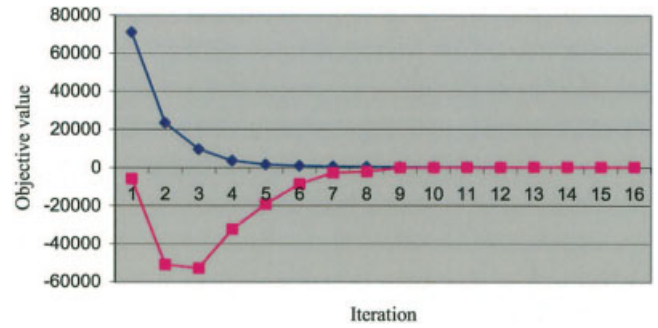


Figure 1. Comparison between LP solution and dual lower bound for problem 13. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

The results are summarized in Table 5, where it can be seen that the same IP solution was found for problem 12 and better IP solutions (406 and 407) for problem 13, compared to the best IP of 408 round with the two previous branching rules. The IP solutions for the other instances were slightly worse than those reported in Table 2 but the computation times were noticeably smaller, except for problem 13.

Using a 5% early termination criterion, solution values deteriorated by an average of 0.18% compared to the results in Table 2 while run times decreased by 46%. With a 1% early termination criterion, solution values deteriorated by 0.16% on average while the run times decreased by 36%. For large instances, truncating the D-W computations prematurely may be justified in light of the small reduction in overall solution quality observed. Using a 5% lower bound still gave good results.

Effectiveness of Double Aggregation

The problem instances used to evaluate the relative effectiveness of the double aggregation scheme were those that had more than 15 subproblems and contained two or more similar rotational profiles that differed only in their ratio values. In model (5), these values are represented by the parameter P_{jr} . Table 6 lists the nine instances that met these criteria. They were all solved using master problem variable branching. The column headings are identical to

Table 5. Performance of B&P algorithm with early termination.

Problem No.	5% lower bound				1% lower bound			
	No. violations	IP soln	Total nodes	Time (s)	No. violations	IP soln	Total nodes	Time (s)
12	115	1180	5	36	115	1180	1	25
13	117	407	100	845	118	406	64	586
14	117	770	15	128	115	770	1	54
15	101	886	1	75	109	884	1	140
16	118	807	5	194	123	810	1	94

Table 6. Computational results for double aggregation.

Prob No.	No. subs	Total penalty	Total slack	Total gap	Problem size ($m \times n$)	New columns	Initial LP sol	Initial gap (%)	IP soln	Time (s)	Total nodes
15	15	120	284	192	95×498	0	879	0.2	881	134	1
16	15	120	240	176	95×499	0	806	0.5	810	108	1
18	15	171	136	272	87×447	0	1243	0.4	1248	46	1
19	15	184	192	184	90×491	0	893	0.5	898	45	1
20	15	201	152	184	95×550	2	937	1.8	944	268	15
21	15	182	184	192	100×525	0	930	0.3	933	196	1
23	15	232	1032	176	90×793	278	879	—	885	1280	100
24	15	267	976	192	90×557	0	997	0.2	999	141	1
25	15	248	944	168	100×542	0	849	1.4	851	431	25

those in Table 2 with the exception of the new column labeled “No. subs,” which gives the total number of sub-problems that are now included in each instance—15 in all cases.

The results in Table 6 indicate that double aggregation improves the lower bound at the root node and reduces the computation times in all instances except No. 25, where no improvement was found. This was to be expected in light of Proposition 2. In fact, the LP solutions coincide with the best IP solutions found in Table 2 for instances 15, 16, 18, and 24, which means that those solutions are optimal.

A related improvement of note is the performance of the IP heuristic at the root node. In all instances but No. 23, the initial gap was 1.8% or less. This compared favorably with the gaps reported in Table 2, which ranged from 12.5 to 55.2% for these instances. In fact, six of the nine problems were solved at the root node using the 0.5% stopping criterion and most within 200 seconds due to the increased quality of the LP and IP solutions.

A further explanation for the relatively good performance of the B&P algorithm relates to the use of slack (y_{dt}) and surplus (s_{dt}) variables in the original demand constraints (1b) and subsequently in the transformed demand constraints (4b). These variables provide natural bounds on the dual variables (μ_{dt}) and hence tighten the dual feasible region. It has been shown empirically and in a more general context that the inclusion of such variables in the master problem constraints both stabilizes and accelerates the solution process [19, 30]. This is especially true in the presence of primal degeneracy where the effort to prove optimality during column generation may be frustrated by the presence of multiple dual solutions. The introduction of bounded slack and surplus variables often eliminates this problem.

6. SUMMARY AND CONCLUSIONS

The purpose of this paper has been to describe a new B&P algorithm for solving the cyclic preference scheduling

problem for nurses. Its effectiveness was demonstrated on problem instances with up to 200 nurses and 30 rotational profiles. The ability of the algorithm to converge in a matter of minutes in most cases was due in part to the design of the IP-based neighborhood search heuristic, which always found the best solutions reported at termination.

Because the decomposition led to a master problem with integer variables, it was necessary to modify existing branching rules in the construction of the search tree. Two approaches were investigated, one based on the master problem variables and the other on subproblem variables. Empirically, the former was more suited for instances with a large number of nurses and profiles, while the latter was more efficient for small and medium-size problems.

In an attempt to improve computational efficiency, two enhancements to the basic algorithm were proposed. The first was to use a lower bound on the LP solution obtained from dual information, rather than the LP solution itself, to terminate the D-W iterations when a prespecified gap was reached. This approach reduced the run times but produced slightly worse solutions. The second was to aggregate the rotational profiles by modifying the assignment constraints in the master problem. This resulted in a stronger formulation and, hence, stronger lower bounds, smaller search trees, and faster convergence. Further aggregation might be possible for cases in which the rotational profiles differed not only by the ratio values but by the total number of working hours as well.

One criticism of models like (1) is that the objective function consists of two incommensurate “costs.” To deal with this issue in practice, we first solve the problem for values of the parameters M_t , $t \in T$, larger than the maximum possible preference penalty given by $|N| \times r_{v_{\max}}$. The solution gives the minimum number of outside nurses needed to meet the demand; call this value O_{\min} . Next, we remove the second term in the objective function (1a) and add to the model a constraint of the form $\sum_{d \in D} \sum_{t \in T} y_{dt} \leq n_{\text{outside}}$. This allows the user to perform “what if” analyses by varying

n_{outside} , where $O_{\min} < n_{\text{outside}}$. A second option being considered is the addition of a budget constraint, $\sum_{d \in D} \sum_{t \in T} M_{dt} y_{dt} \leq B$, where B is the current budget for outside nurses.

The composition of the rotational profiles restricts the coverage patterns that can be assigned over the week. An interesting area of future research would be to develop a procedure for determining the optimal set of profiles when the demand is given, or more robustly, when the demand follows a known probability distribution. A related problem centers on the augmentation of the staff. If it were possible to hire one or more nurses, we would like to be able to determine the best profiles to assign them. Although this problem sounds simple, it has the same complexity as the original.

REFERENCES

- [1] U. Aickelin, and K. Dowland, An indirect algorithm for a nurse-scheduling problem, *Comput Oper Res* 31 (2004), 761–778.
- [2] U. Aickelin, and P. White, Building better nurse scheduling algorithms, *Ann Oper Res* 128 (2004), 159–177.
- [3] J.F. Bard, and H.W. Purnomo, Hospital-wide reactive scheduling of nurses with preference considerations, *IIE Trans Oper Eng* 37(7) (2005), 589–608.
- [4] J.F. Bard, and H.W. Purnomo, Preference scheduling for nurses using column generation. *Eur J Oper Res* 164 (2005), 510–534.
- [5] J.F. Bard, and H.W. Purnomo, A column generation-based approach to solve the preference scheduling problem for nurses with downgrading. *Socio-Econ Plan Sci* 39(3) (2005), 193–213.
- [6] C. Barnhart, C.A. Hane, and P.H. Vance, Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Oper Res* 48(2) (2000), 318–326.
- [7] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance, Branch-and-price: Column generation for solving huge integer programs, *Oper Res* 46(3) (1998), 316–329.
- [8] J.J. Bartholdi III, A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering. *Oper Res* 29(3) (1981), 501–510.
- [9] S.E. Bechtold, and L.W. Jacobs, The equivalence of general set-covering and implicit integer programming formulations for shift scheduling. *Nav Res Logist* 43(2) (1996), 233–249.
- [10] I. Berrada, J.A. Ferland, and P. Michelon, A multi-objective approach to nurse scheduling with both hard and soft constraints. *Socio-Econ Plan Sci* 30(3) (1996), 183–193.
- [11] M.J. Brusco, Solving personnel tour scheduling problems using the dual all-integer cutting plane. *IIE Trans Oper Eng* 30(9) (1998), 835–844.
- [12] E.K. Burke, P.D. Causmaecker, and G.V. Berghe, “A hybrid tabu search algorithm for the nurse rostering problem,” *Simulated Evolution and Learning B. McKay et al. (Editors) Lecture Notes in Artificial Intelligence* 1585, Springer, Berlin, 1998, pp. 187–194.
- [13] E.K. Burke, P.D. Causmaecker, G.V. Berghe, and H. Van Landeghem, The state of the art of nurse rostering. *J Scheduling* 7(6) (2004), 441–499.
- [14] R.N. Burns, and M.W. Carter, Work force size and single shift schedules with variable demands. *Manage Sci* 31(5) (1985), 599–607.
- [15] A. Caprara, M. Monaci, and P. Toth, Models and algorithms for a staff scheduling problem, *Math Progr B* 98 (2003), 445–476.
- [16] B. Cheang, H. Li, A. Lim, and B. Rodrigues, Nurse rostering problems—A bibliographic survey. *Eur J Oper Res* 151 (2003), 447–460.
- [17] B.M.W. Cheng, J.M.L. Lee, and J.C.K. Wu, A nurse rostering system using constraint programming and redundant modeling, *IEEE Trans Inform Technol Biomed* 1(1) (1997), 44–54.
- [18] K.A. Dowland, Nurse scheduling with tabu search and strategic oscillation. *Eur J Oper Res* 106 (1998), 393–407.
- [19] Du Merle, O.D. Villeneuve, J. Desrosiers, and P. Hansen, Stabilized column generation. *Discrete Math* 194 (1999), 229–237.
- [20] H. Emmons, Work-force scheduling with cyclic requirements and constraints on days off, weekends off, and work stretch. *IIE Trans* 17(1) (1985), 8–15.
- [21] A.T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier, Staff scheduling and rostering: A review of applications, methods and models, *Eur J Oper Res* 153 (2003), 3–27.
- [22] G. Felici, and C. Gentile, A polyhedral approach for the staff rostering problem. *Manage Sci* 50(3) (2004), 381–393.
- [23] J.A. Ferland, I. Berrada, I. Nabli, B. Ahiod, P. Michelon, V. Gascon, and E. Gagné, Generalized assignment type goal programming problem: Application to nurse scheduling. *J Heurist* 7 (2001), 391–413.
- [24] H. Griesmer, Self-scheduling turned us into a winning team. *Manage Decisions* 56(12) (1993), 21–23.
- [25] J.P. Howell, Cyclical scheduling of nursing personnel. *Hosp JAHA* 40 (1998), 77–85.
- [26] M.W. Isken, An implicit tour scheduling model with applications in healthcare. *Ann Oper Res* 128 (2004), 91–109.
- [27] B. Jaumard, F. Semet, and T. Vovor, A generalized linear programming model for nurse scheduling. *Eur J Oper Res* 107 (1998), 1–18.
- [28] H. Kawanaka, K. Yamamoto, T. Yoshikawa, T. Shinogi, and S. Tsuruoka, “Genetic algorithm with constraints for the nurse scheduling problem,” *Proceedings of Congress on Evolutionary Computation*, IEEE Press 2, Seoul, South Korea, 2001, pp. 1123–1130.
- [29] H.C. Lau, On the complexity of manpower shift scheduling, *Comput Oper Res* 23(1) (1996), 93–102.
- [30] M.E. Lübbecke, and J. Desrosiers, Selected topics in column generation, *Oper Res* 53(6) (2005), 1007–1023.
- [31] H.E. Miller, W.P. Pierskalla, and G.J. Rath, Nurse scheduling using mathematical programming, *Oper Res* 24(5) (1976), 857–870.
- [32] K. Nonobe, and T. Ibaraki, A tabu search approach to the constraint satisfaction problem as a general problem solver, *Eur J Oper Res* 106 (1998), 599–623.
- [33] M. Okada, An approach to the generalized nurse scheduling problem—Generation of a declarative program to represent institution-specific knowledge. *Comput Biomed Res* 28 (1992), 417–434.
- [34] W. Pierskalla, and D.J. Brailer. “Applications of operations research in health care delivery,” *Handbooks in Operations*

- Research & Management Science* 6, Amsterdam, North Holland, 1994, pp. 469–505.
- [35] E.R. Pinker, and R.C Larson, Optimizing the use of contingent labor when demand is uncertain, *Eur J Oper Res* 144(1) (2003), 39–55.
- [36] V. Quan, Retail labor scheduling, *OR/MS Today* 31(6) (2005), 33–35.
- [37] S.U. Randhawa, and D. Sitompul, A heuristic-based computerized nurse scheduling system, *Comput Oper Res* 20(8) (1993), 837–844.
- [38] C. Valouxis, and E. Housos, Hybrid optimization techniques for the workshift and rest assignment of nursing personnel, *Artif Intell Med* 20 (2000) 155–175.
- [39] F. Vanderbeck, On Dantzig–Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm, *Oper Res* 48(1) (2000), 111–128.
- [40] D.M. Warner, Scheduling nursing personnel according to nursing preference: A mathematical programming approach, *Oper Res* 24(5) (1976), 842–856.
- [41] L.A. Wolsey, *Integer Programming*, Wiley, New York, 1998.