A Hybrid AI Approach for Nurse Rostering Problem

Haibing Li[1], Andrew Lim[2] and Brian Rodrigues[3]
[1] Dept of Computer Science, NJIT, University Heights, Newark, NJ 07102
[2] Dept of IEEM, HKUST, Clear Water Bay, Kowloon, Hong Kong
[3] School of Business, Singapore Management University, 469 Bukit Timah Road, S259756

ABSTRACT

This paper presents a hybrid AI approach for a class of over-constrained Nurse Rostering Problems. Our approach comes in two phases. The first phase solves a relaxed version of problem which only includes hard rules and part of nurses' requests for shifts. This involves using a forward checking algorithm with non-binary constraint propagation, variable ordering, random value ordering and compulsory backjumping. In the second phase, adjustments with descend local search and tabu search are applied to improve the solution. This is to satisfy the preference rules as far as possible. Experiments show that our approach is able to solve this class of problems well.

1. INTRODUCTION

Duty rosters are quintessential towards the proper functioning of any hospital. The proper scheduling of nursing staff thus has a great impact on health care quality. Duty rosters can be generated manually by nursing officers for each hospital unit. However, this is a difficult and tedious job. The main reason lies in that hospitals need to be staffed around the clock each day. In addition, in many hospitals some nurses are allowed to ask for pre-set shifts, while other nurses are scheduled around these pre-set shifts. Usually, nursing officers have to spend a substantial amount of time on producing these rosters. This is especially so when there are a lot of staff requests; Also, more time may be spent on handling ad hoc changes to the current duty roster. These tedious and time-consuming manual scheduling operations, or what we call the "*Nurse Rostering Problems*" (NRPs) have attracted much research attention during the recent decades.

The NRPs involve producing a periodic duty roster for the nursing staff, subject to a variety of hard/soft constraints such as legal regulations, personnel policies, nurses' preferences, and other requirements. These constraints are different from one hospital to another. For this reason, various models and algorithms have been proposed to represent and consequently solve the NRPs. A recent survey on modelling issues and algorithms for the NRPs is referred to Li *et al.* [1].

This paper proposes a hybrid AI approach, which combines constraint-based techniques and local search techniques, for a class of NRPs defined in the next section.

SAC 2003, Melbourne, Florida

2. DEFINITION OF THE PROBLEM

We are interested in a class of NRPs which are abstracted from real-life hospital problems. The domains, planning rules, and objectives of the NRPs are described as follows.

2.1 Domain Description

2.1.1 Nurse –

In the NRPs, the total number of nurses to be scheduled is N. Usually, nurses in a clinical unit are classified into several levels. These include: nursing officers, senior nurses, staff nurses, and assistant nurses. To ensure the health care quality, real-life problems have minimum requirements of each staffing level for each shift. For each group of nurses within the same staffing level, nurses can be scheduled independently subject to the same planning rules. Therefore, it is convenient for us to assume that the N nurses have the same staffing level.

2.1.2 Shift –

Nurses work in shifts. There are three basic on-duty shifts a day: morning shift ('A'), afternoon shift ('P'), and night shift ('N'). The off-duty shifts include: day-off, compensation-off, public holiday, vacation leave, study day, sick leave, maternity leave, unpaid leave, and so on. Thus, the decision variables for shifts usually take 11 or more values. This increases computational efforts for algorithms. For this reason, Heus & Weil [2] proposed a reduction for variables' domains: All of the off-duty shifts can be regarded as a free shift ('F') so that each variable takes only 4 possible values: 'A', 'P', 'N', and 'F'.

2.2 Planning Rules

Planning rules include hard rules and preference rules. These are constraints for the NRPs. Hard rules are usually legal regulations, union rules, and personnel requirements that must be strictly satisfied. Preference rules are usually nurses' wishes that can be violated but should be satisfied as far as possible. In further detail, hard rules include:

Rule 1: Each nurse is assigned to one and only one shift ('A', 'P','N', or 'F') per day.

Shift Types	'A'	'P'	'N'
Minimum Staff	n_1	n_2	n_3

Table 1: Minimum Required Staff Per Shift

Nurse	Mon	Tue	Wed	Thu	Fri	Sat	Sun
N-01	A	P	N	A	F	F	P
N-02	P	A	N	P	F	F	A
N-03	N	N	N	F	F	F	A

Table 2: Nurse-Day View

Rule 2: For each nurse, the total work days within a scheduling period of P days must not exceed a bound wl. Any excess will be considered as overtime.

Rule 3: The minimum rest period between shifts is 8 hours. This implies that 'N' followed by 'A' is not permitted.

Rule 4: Each nurse works at most 1 night shift per week.

Rule 5: The minimum required staff per shift must be met. The staff requirement for each shift is shown in Table 1.

Preference rules include:

Rule 6: Nurses' requests for shifts.

Rule 7: the nurses prefer longer off-duty periods as provided by the following patterns:

Rule 7.1: 'A' → 'F'

Rule 7.2: 'F' → 'N'

Rule 7.3: 'F' → 'P'

Rule 8: The following patterns should be avoided as far as possible: 'P' → 'A'

2.3 Objective

The objective in solving the problem is to produce a weekly (or fortnightly, monthly) duty roster which satisfies all hard rules and satisfies the preference rules as far as possible.

3. PROBLEM MODELLING

The duty rosters of NRPs can be conveniently described as a nurse-day view. Table 2 shows part of a weekly roster which indicates the shifts allocated to a group of nurses. This view is a direct depiction of the two-dimensional layout of duty rosters. Accordingly, the decision variable can be defined for each nurse on each day as $v_{ij} = \{0, 1, 2, 3\}$, with 0, 1, 2, 3 corresponding to 'A', 'P', 'N', and 'F' respectively, here, $1 \leq i \leq N$ indexes the nurses, and $1 \leq j \leq P$ indexes the days within a scheduling period P. Suppose $\mathcal{X} = [V_1, V_2, \ldots, V_i, \ldots, V_N]^T$, where $V_i = [v_{i1}, v_{i2}, \ldots, v_{iP}], 1 \leq i \leq N$. We define $\mathcal{F}(\mathcal{X})$ as a function mapping $\mathcal{X}$ to a positive value which reflects the satisfaction level of the preference rules. The problem can be formulated as:

$$maximize\ \mathcal{F}(\mathcal{X}) \tag{1}$$

strictly subject to the following hard constraints corresponding to the hard rules respectively:

$$v_{ij} \in \{0, 1, 2, 3\}, 1 \leq i \leq N, 1 \leq j \leq P \tag{2}$$

$$\begin{gathered}\sum_{j=1}^{P} f_1(v_{ij}) \leq wl, 1 \leq i \leq N, where,\\ f_1(v_{ij}) = \begin{cases} 0 & \text{when } v_{ij} = 0, \\ 1 & \text{otherwise.} \end{cases}\end{gathered} \tag{3}$$

$$v_{ij} = 2 \rightarrow v_{i,j+1} \neq 0, 1 \leq i \leq N, 1 \leq j \leq P-1 \tag{4}$$

$$\begin{gathered}\sum_{j=1}^{P} f_2(v_{ij}) \leq \lceil P/7 \rceil, 1 \leq i \leq N, where,\\ f_2(v_{ij}) = \begin{cases} 1 & \text{when } v_{ij} = 3, \\ 0 & \text{otherwise.} \end{cases}\end{gathered} \tag{5}$$

$$\begin{gathered}\sum_{i=1}^{N} f_3(v_{ij}, s) = Table_1(s), 1 \leq j \leq P, where,\\ f_3(v_{ij}, s) = \begin{cases} 1 & \text{nurse } i \text{ is assigned shift } s \text{ on day } j, \\ 0 & \text{otherwise.} \end{cases}\end{gathered} \tag{6}$$

Here, $Table_1(s)$ refers to the minimum required staff for shift type s, as shown in Table 1.

If any one of the above hard constraints is violated, $\mathcal{F}(\mathcal{X}) = -\infty$, otherwise, $\mathcal{F}(\mathcal{X}) = 0$.

In addition to the hard constraints, satisfaction of preference rules is incorporated into $\mathcal{F}(\mathcal{X})$ by weights which reflect the preference order. Thus, the nurses' wishes are to be satisfied as far as possible by maximizing $\mathcal{F}(\mathcal{X})$. Suppose $\mathcal{X}'$ is a pre-set duty roster for a requested shift, and $|\mathcal{X} \simeq \mathcal{X}'|$ is the number of satisfied requests for shifts, then Rule 6 can be incorporated into $\mathcal{F}(\mathcal{X})$ as,

$$\mathcal{F}(\mathcal{X}) = \mathcal{F}(\mathcal{X}) + w_1|\mathcal{X} \simeq \mathcal{X}'| \tag{7}$$

Furthermore, Rule 7 and Rule 8 can be incorporated as,

$$\begin{gathered}\forall i, j | 1 \leq i \leq N, 1 \leq j \leq P-1,\\ v_{ij} = 0 \wedge v_{i,j+1} = 3 \rightarrow \mathcal{F}(\mathcal{X}) = \mathcal{F}(\mathcal{X}) + w_2\\ v_{ij} = 3 \wedge v_{i,j+1} = 2 \rightarrow \mathcal{F}(\mathcal{X}) = \mathcal{F}(\mathcal{X}) + w_3\\ v_{ij} = 3 \wedge v_{i,j+1} = 1 \rightarrow \mathcal{F}(\mathcal{X}) = \mathcal{F}(\mathcal{X}) + w_4\end{gathered} \tag{8}$$

$$\begin{gathered}\forall i, j | 1 \leq i \leq N, 1 \leq j \leq P-1,\\ v_{ij} = 1 \wedge v_{i,j+1} = 0 \rightarrow \mathcal{F}(\mathcal{X}) = \mathcal{F}(\mathcal{X}) - w_5\end{gathered} \tag{9}$$

When different weights are assigned to w_i(i=1, 2, ..., 5), the above formulation can be customized to be various constraint models.

Firstly, if $\forall i, w_i = 0$, the objective function (1) is trivial. In such a case, since no objective function is presented, the NRPs become Constraint Satisfaction Problems(CSPs) [3]. Otherwise, one can define $\mathcal{F}(\mathcal{X})$ as other functions which map $\mathcal{X}$ to reflect preferences of the decision makers, such as minimum staffing requirements, best health care quality, and so on. Therefore, the problem can be modelled as Mathematical Programs or Goal Programs. However, the formulations (2 to 6) introduce too many constraints to be dealt with by the mathematical programs.

Secondly, for other settings of w_i, the NRPs can be modelled as variants of CSPs which involve optimization of objectives. Naturally, over-constrained CSPs can be modelled as Partial Constraint Satisfaction Problems (PCSPs) [4], Constraint Hierarchies (CHs) [5] or Hierarchical Constraint Satisfaction Problems (HCSPs) [6]. These models weigh the importance of the constraints using partial orders or hierarchical structures.

In our research, we model the NRP as a Weighted Constraint Satisfaction Problem (WCSP), under the condition that all the hard rules are strictly satisfied. For the preference rules, the overall preference ordering is: Rule 6 $\succ$ Rule 8 $\succ$ Rule 7 by setting values for w_i. However, instead of strictly sticking to this preference order, satisfaction level of the preference rules is determined by the objective function $\mathcal{F}(\mathcal{X})$. Therefore, partial or hierarchical ordering which is emphasized in PCSPs, CHs, and HCSPs is relaxed in WCSPs. The reason for this is that human's preference for choice is quite fuzzy, it is thus not natural to make it fixed that one preference is definitely more favorable than another. For example, when two nurses are satisfied with Rule 7 at the expense of dissatisfaction of another nurse's request for a shift, the health care quality may be better than the case that the two nurses are not satisfied with Rule 7 while the other nurse's request for shift is satisfied. Therefore, the objective function plays an important role for the satisfaction level of the soft constraints. A WCSP is defined as,

DEFINITION 1. *A WCSP is a tuple:*

$$\mathcal{T} =< \mathcal{X}, \mathcal{D}, \mathcal{C}, \mathcal{W}, f, \succ>$$

Where, $\mathcal{X}$ is a set of decision variables, $\mathcal{D}$ is a set of domains specifying the values taken by the decision variables, $\mathcal{C}$ is a set of constraints, $\mathcal{W}$ is a set of values weighting the importance of the constraints. $\succ$ denotes preference order. Let $\mathcal{X} \downarrow \mathcal{X}'$ denote labelling the variables in $\mathcal{X}$ as $\mathcal{X}'$,

$$f(c, \mathcal{X} \downarrow \mathcal{X}') = \begin{cases} 0 & \textit{if } \mathcal{X}' \textit{ satisfies } c \in \mathcal{C}, \\ 1 & \textit{otherwise.} \end{cases}$$

Further, $\mathcal{F}(\mathcal{X})$ denotes a function evaluating the satisfaction level of $\mathcal{C}$, and define the set of satisfied constraints $\mathcal{S}(\mathcal{X}')$ for $(X) \downarrow (X')$ as $\mathcal{S}(\mathcal{X}') = \{c \in \mathcal{C} | f(c, \mathcal{X} \downarrow \mathcal{X}') = 0\}$, then $\mathcal{S}(\mathcal{X}') \succ \mathcal{S}(\mathcal{X}'')$ iff $\mathcal{F}(\mathcal{X}') > \mathcal{F}(\mathcal{X}'')$.

To follow this definition, we make the following assumptions for the NRP: $(w_1) \succ (w_5) \succ (w_2 = w_3 > w_4)$. If $\succ$ represents $\gg$, then a WCSP becomes a PCSP/HCSP. However, we weigh w_i with real values, so that the NRP becomes a WCSP. Like the PCSP/HCSP, the algorithms for the CSPs can be customized to solve the WCSPs. In particular, combination of Branch & Bound (B&B) with Constraint Propagation, Variable Ordering, and Value Ordering is helpful for speeding up the B&B algorithm for search of an optimal solution. However, for problems with large numbers of decision variables, much computational effort is required. Because of this reason, we seek a different approach which combines CSP algorithms with local search techniques. In the next section, we shall discuss this approach in detail.

4. THE ALGORITHM

Our approach for solving the NRP as a WCSP is a two-phase algorithm as follows:

Phase 1: Obtain an initial solution for a subproblem which incorporates all hard rules and part of nurses' requests for shifts. The subproblem was constraint- relaxed so that it's not over-constrained and can be solved as a CSP.

Phase 2: Start local searches to improve the solution, so that the soft constraints can be satisfied as far as possible.

4.1 Solving the Subproblem as a CSP

In Phase 1, to check if a CSP is over-constrained is $\mathcal{NP}$-Complete. However, for a subproblem, Rule 5 is the direct constraint affecting if it is over- constrained. With regards to the nurse-day view in Table 2 and the constraints (2 to 6) formulated on this view, we use the following heuristic to obtain a relaxed subproblem by over-softening part of the nurses' requests for shifts to initially satisfy Rule 5.

procedure SOFTEN-CONSTRAINTS

for each day $d \leftarrow 1$ to P; shift $s \leftarrow 1$ to 3 **do**

calculate $r(s)$: the number of s requested by the nurses on day d

while $r(s) + \tau > Table_1(s)$ **do**

disable the request for s from the nurse with the maximum number of requests; $r(s) \leftarrow r(s) - 1$

In this procedure, τ, being less than $Table_1(s)$, is an integer which helps the subproblem be over-softened. Since the disabled requests are of those who have the maximum number of requests; those nurses will still be satisfied because their other requests can be fulfilled.

A number of techniques have been proposed for solving CSPs [3, 7, 8]. Among systematic algorithms, Backtracking algorithms, such as basic Backtracking (BT), Backjumping(BJ), Backmarking(BM), Forward Checking(FC), and Maintaining Arc Consistency(MAC), are still the most popular. All of these algorithms do consistency checking backward or forward to speed up the backtracking procedure. Experimental results indicate that the algorithms which do some level of forward constraint propagation are the most successful. For example, FC and MAC. However, most of the literature about these algorithms have only dealt with arc-consistency(AC) in Constraint Propagation (CP) for solving binary CSP.

The constraints in the NRPs are more complicated than binary constraints. From the nurse-day view in Table 2, constraints span the decision variables horizontally over each day within the planning period P and vertically over each nurse to be scheduled. This fact is also reflected by the constraint formulations (3 to 6). For simplicity, we name these constraints *Horizontal Constraints* and *Vertical Constraints* with regards to the nurse-day view. Each horizontal constraint involves up to P decision variables, while each vertical constraint involves up to N decision variables. Therefore, for the NRPs, AC helps very little on constraint propagation. Although a non-binary CSP can be converted to a binary CSP [9, 10, 11], we propose a forward checking algorithm with non-binary constraint propagation, for solving the CSP.

4.1.1 Forward Checking with Non-Binary Constraint Propagation –

The instantiation order of v_{ij} in the nurse-day view is nurse by nurse and then day by day for each nurse. When v_{ij} is instantiated as a shift $s \in \mathcal{D}_{ij}$ ($\mathcal{D}_{ij}$ is the domain of v_{ij}), constraint propagation, as the following function NONBINARY-CP, can be done both horizontally and vertically to reduce

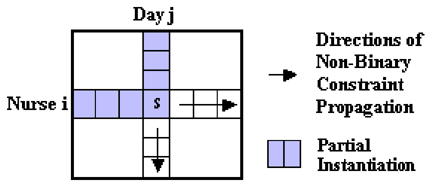


Figure 1: Non-Binary Constraint Propagation

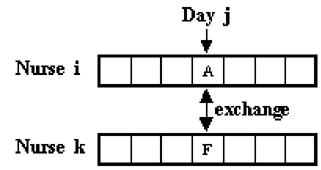


Figure 2: Vertical Exchange

the domains of related decision variables which have the constrained relationship with v_{ij}, as shown in Figure 1.

function NONBINARY-CP(i, j)

$dwo \leftarrow$ **false**
for $k \leftarrow j+1$ to P **do** ◁ *horizontal propagation*
  for each $s' \in \mathcal{D}'_{ik}$ **do** ◁ $\mathcal{D}'_{ik}$ *is a copy of* $\mathcal{D}_{ik}$
    if $(v_{i1}, v_{i2}, \ldots, v_{ij}, v_{ik} = s')$ is not consistent **then**
      remove s' from $\mathcal{D}'_{ik}$
      if $cardinality(\mathcal{D}'_{ik}) = 0$ **then** $dwo \leftarrow$ **true**
for $k \leftarrow i+1$ to N **do** ◁ *vertical propagation*
  for each $s' \in \mathcal{D}'_{kj}$ **do**
    if $(v_{1j}, v_{2j}, \ldots, v_{ij}, v_{kj} = s')$ is not consistent **then**
      remove s' from $\mathcal{D}'_{kj}$
      if $cardinality(\mathcal{D}'_{kj}) = 0$ **then** $dwo \leftarrow$ **true**
return dwo

Apart from returning the status whether the domain of a variable is wiped out by constraint propagation, this function revises domains in: $\bigcup_{k=j+1}^{P} \mathcal{D}'_{ik}$ and $\bigcup_{k=i+1}^{N} \mathcal{D}'_{kj}$.

The forward checking algorithm with non-binary constraint propagation, certain degree of Variable Ordering (VRO), random Value Ordering (VLO), and compulsory Backjumping (CBJ) are as follows:

procedure FC-CBJ-NONBINARY-CP

$backtracks \leftarrow 0$
reorder V_i in terms of the number of shifts requested by the nurses ◁ *certain degree of VRO*
for $i \leftarrow 1$ to N; $j \leftarrow 1$ to P **do** $\mathcal{D}'_{ij} \leftarrow \mathcal{D}_{ij}$ ◁ *copy*
while stop condition is not met yet **do**
  $r \leftarrow$ CHECK-FORWARD(1, 1)
  if $r = 2$ **then**
    for $i \leftarrow 1$ to N; $j \leftarrow 1$ to P **do** $\mathcal{D}'_{ij} \leftarrow \mathcal{D}_{ij}$ ◁ *CBJ*
  else if $r = 1$ **then return** ◁ *a solution is found*
output notification of failure

where, the function CHECK-FORWARD is defined as,

function CHECK-FORWARD(i, j)

for each randomly-drawn $s \in \mathcal{D}'_{ij}$ **do** ◁ *random VLO*
  $v_{ij} \leftarrow s$
  if $i = N$ and $j = P$ **then**
    print instantiated $\mathcal{X}$ ◁ *a solution is found*
    return 1
  else
    if NONBINARY-CP(i, j) $\neq false$ **then**
      $j' \leftarrow j+1$ ◁ *move to the next variable*
      $i' \leftarrow i + \lfloor j' / P \rfloor$; $j' \leftarrow j' \% P$
      $r \leftarrow$ CHECK-FORWARD(i', j') ◁ *recursion*
      if $r \neq 0$ **then return** r
    reset domains in $\bigcup_{k=j+1}^{P} \mathcal{D}'_{ik}$ and $\bigcup_{k=i+1}^{N} \mathcal{D}'_{kj}$ to their values before v_{ij} was instantiated to be s
$backtracks \leftarrow backtracks + 1$
if $backtracks \geq \kappa$ **then** ◁ *for CBJ*
  return 2
else return 0

In this function, κ is a parameter for tracing the number of backtracks made so far. When $\kappa = \infty$, the function becomes a standard check forwarding algorithm with non-binary constraint propagation and random value-ordering. Otherwise, this function will be forced to jump back to search from v_{11} after every κ backtracks.

4.2 Local Adjustment

The result obtained by the above CSP algorithm is the solution only to a subproblem. To improve satisfaction of the preferences rules, we apply local adjustments to the initial solution, with regards to Rule 6 to Rule 8.

4.2.1 Local Adjustment Structure –

From the nurse-day view, it is more preferable to exchange the shifts vertically for two decision variables within the same column. The reason for this is: Rule 5 (as a vertical constraint) has been satisfied by the initial solution already. Therefore, vertical exchange of shifts between two variables within the same column will not violate this rule. Otherwise, if we consider horizontal exchange, we need to take care of this rule. The fact is that Rule 5 is always violated by exchanging the shifts between two variables horizontally, in case that the two shifts are not 'F'. However, exchanging two 'F' shifts within the same row makes no sense, thus we only consider vertical exchange as Figure 2. Local adjustments are carried out based on this exchanging operator, as follows:

function LOCAL-ADJUSTMENT($\mathcal{X}$)

$improved \leftarrow false$
for $j \leftarrow 1$ to P; $i \leftarrow 1$ to N; $k \leftarrow 1$ to N, $k \neq i$ **do**

if $v_{kj} \neq v_{ij}$ **then**
  if exchanging shift of v_{kj} with shift of v_{ij} is consistent and results in a better $\mathcal{F}(\mathcal{X})$ **then** do the exchange; $improved \leftarrow true$
record the improved $\mathcal{X}$; **return** $improved$

4.2.2 *Descent Local Search and Tabu Search –*

Based on LOCAL-ADJUSTMENT, descent local search (DLS) or metaheuristic algorithms can be implemented to improve the solution quality. First, we conduct a DLS on the initial solution, and then apply a tabu search (TS) from the solution obtained by DLS. DLS starts LOCAL-ADJUSTMENT from a seed solution, if a better $\mathcal{X}$ is found, then repeat LOCAL-ADJUSTMENT on $\mathcal{X}$, until no improvement can be found. TS labels along its exploring trail accepts unexploited local optima whether they are better or worse than the seed solutions. The TABU-LOCAL-ADJUSTMENT is used to avoid cycling searches,

function TABU-LOCAL-ADJUSTMENT($\mathcal{X}$)

$bestValue \leftarrow -\infty$
for each non-tabu vertical exchange within $\mathcal{X}$
  if the vertical exchange will be consistent and will result in $\mathcal{F}(\mathcal{X}) > bestValue$ **then**
    record this vertical exchange as the best candidate
    $bestValue \leftarrow \mathcal{F}(\mathcal{X})$
$\mathcal{X}' \leftarrow$ carry out the best vertical exchange
record the best vertical exchange in a short-term tabu list with a fixed length L; **return** $\mathcal{X}'$

function TABU-SEARCH($\mathcal{X}$)

$\mathcal{X}_b \leftarrow \mathcal{X}$
while stop condition is not met yet **do**
  $\mathcal{X}' \leftarrow$ TABU-LOCAL-ADJUSTMENT($\mathcal{X}$)
  if $\mathcal{F}(\mathcal{X}') > \mathcal{F}(\mathcal{X}_b)$ **then** $\mathcal{X}_b \leftarrow \mathcal{X}'$
  $\mathcal{X} \leftarrow \mathcal{X}'$
return $\mathcal{X}_b$

5. EXPERIMENTAL RESULTS

The algorithm was coded in Visual C++ 6.0. For evaluation, we conducted the following experiments on PC P-545MHZ.

5.1 Experiments for the CSP Algorithm

For the subproblems with only hard rules, we also programmed in ILOG 4.4 for comparison purposes. The parameters are set as: $n_1 = n_2 = 2N/9$, $n_3 = N/9$, $wl = 5$; $\kappa = 200$. Table 3 shows the running time for obtaining feasible solutions by the two methods. From the trends of the running time versus N shown in Figure 3, the running time of ILOG increases very fast with N. In contrast, our algorithm is quite fast and stable, feasible solutions can be obtained within 1 second.

5.2 A Case Study

To test our algorithm for solving the over-constrained NRPs, we had a case study. Figure 4 shows a NRP with pre-set shift requests of a weekly duty roster ($P = 7$). In this problem, the paramters are set as: $N = 27, n_1 = n_2 = 6, n_3 = 3$,

N	ILOG (Sec)	Our Algorithm (Sec)
9	0.05	0.0
18	2.31	0.05
27	6597.43	0.11
36	out of time	0.16
45	out of time	0.28
54	out of time	0.33
108	out of time	0.77

Table 3: Running Time: Our Algorithms *vs* ILOG 4.4

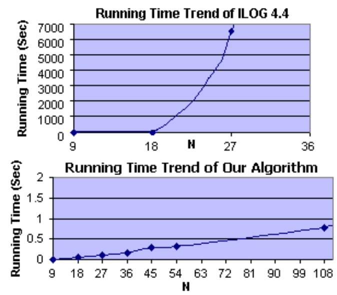
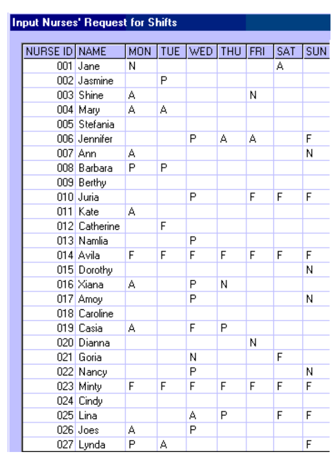


Figure 3: Running Time Trends

$wl = 5$; $w_1 = 10, w_2 = w_3 = 5, w_4 = 4, w_5 = 6$; $\tau = 2, \kappa = 200, L = N$. The tabu search procedure is terminated after 200 iterations without improvement. Figure 4 shows a duty roster obtained by the our algorithm. In the roster, requested shifts in the bold and italic fonts were satisfied, while requested shifts in the "underlined" font were dissatisfied.

The satisfaction level of preferences rules reached by the duty roster is shown as Table 4. In the table, the maximum patterns for Rule 6 are requested by the nurses (59). For Rule 7.1, 7.2, and 7.3, the number of patterns are dominated by the total number of shifts 'A', 'N' and 'P', i.e., $n_1 \times P = 42$, $n_3 \times P = 21$ and $n_2 \times P = 42$, respectively. The items subtracted from these numbers in the table mean that some requested shifts make it impossible to satisfy related patterns. For Rule 8, the maximum number of patterns is


Input Nurses' Request for Shifts

NURSE ID	NAME	MON	TUE	WED	THU	FRI	SAT	SUN
001	Jane	N					A	
002	Jasmine		P					
003	Shine	A				N		
004	Mary	A	A					
005	Stefania							
006	Jennifer			P	A	A		F
007	Ann	A						N
008	Barbara	P	P					
009	Berthy							
010	Juria			P		F	F	F
011	Kate	A						
012	Catherine		F					
013	Namlia			P				
014	Avila	F	F	F	F	F	F	F
015	Dorothy							N
016	Xiana	A		P	N			
017	Amoy			P				N
018	Caroline							
019	Casia	A		F	P			
020	Dianna					N		
021	Goria			N			F	
022	Nancy			P				N
023	Minty	F	F	F	F	F	F	F
024	Cindy							
025	Lina			A	P		F	F
026	Joes	A		P				
027	Lynda	P	A					F

Figure 4: A Nurse Rostering Problem

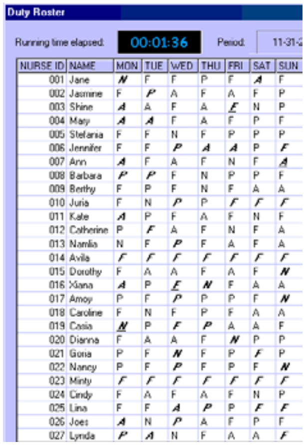


Figure 5: A Duty Roster for the Nurse Rostering Problem

Rules	Satisfied	Max Patterns	Percentage
Rule 6	55	59	93%
Rule 7.1	20	42-3	51%
Rule 7.2	16	21-2	84%
Rule 7.3	16	42-4	42%
Rule 8	159	162	96%

Table 4: Satisfaction Level of the Preference Rules

$N \times (P-1) = 162$, only 3 patterns (not including requested patterns) violate this Rule. The percentage shown in the table roughly reflects the preference order for the preference rules. Figure 5 also shows that the running time is 1 minute 36 seconds.

6. CONCLUSION

In this paper, we proposed a hybrid AI approach for solving a class of NRPs, which were modeled as weighted CSPs. Our approach consists of two phases. Our algorithm was able to return a feasible solution within 1 second for the subproblem with only hard rules, while ILOG run out of time for problems with more than 36 nurses. For the subproblem with hard rules and part of nurses' requests for shifts, our algorithm could also obtain an initial solution within several seconds. In the second phase, adjustments with descend local search and tabu search, which are based on a neighborhood structure for vertical exchange of shifts, were applied to improve the solution to satisfy the preference rules as far as possible. The case study showed that our approach was able to solve this class of NRPs quickly. In our future work, we will apply the ideas in this hybrid AI approach to solve other over-constrained CSPs.

7. REFERENCES

[1] H.; Li, A.; Lim, and B. Cheang. Algorithms for nurse rostering problems: A survey. *Submitted for publication*, 2001.

[2] K.; Heus and G. Weil. Constraint programming a nurse scheduling application. In *Proceedings of the Second International Conference on the Practical Application of Constraint Technology*, pages 115–127, 1996.

[3] V. Kumar. Algorithms for constraint satisfaction problems: A survey. *AI Magazine*, 13(1):32–44, 1992.

[4] E.C.; Freuder and R.J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58(1-3):21–70, 1992.

[5] A.; Borning, F-B.; Bjorn, and M. Wilson. Constraint hierarchies. *Lisp and Symbolic Computation*, 5(3):223–270, 1992.

[6] H. Meyer auf'm Hofe. Nurse rostering as constraint satisfaction with fuzzy constraints and inferred control strategies. In E.C.Freuder and R.J.Wallace, editors, *Constraint Programming and Large Scale Discrete Optimization*, volume 57 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 67–99. DIMACS, 2001.

[7] R.; Dechter and D. Frost. Backtracking algorithms for constraint satisfaction problems, 1999.

[8] F.; Bacchus and A. Grove. Looking forward in constraint satisfaction algorithms, 1999.

[9] R.; Dechter and J. Pearl. Tree clustering for constraint networks. *Artificial Intelligence*, 38:353–366, 1989.

[10] F.; Bacchus and P. van Beek. On the conversion between non-binary and binary constraint satisfaction problems. In *AAAI'1998*, pages 311–318, 1998.

[11] K.; Stergiou and T. Walsh. Encodings of non-binary constraint satisfaction problems. In *AAAI'1999*, 1999.