A Hybrid Constraint Programming Approach for Nurse Rostering Problems

Rong Qu and Fang He

Abstract: Due to the complexity of nurse rostering problems (NRPs), Constraint Programming (CP) approaches on their own have shown to be ineffective in solving these highly constrained problems. We investigate a two-stage hybrid CP approach on real world benchmark NRPs. In the first stage, a constraint satisfaction model is used to generate weekly rosters consist of high quality shift sequences satisfying a subset of constraints. An iterative forward search is then adapted to extend them to build complete feasible solutions. Variable and value selection heuristics are employed to improve the efficiency. In the second stage, a simple Variable Neighborhood Search is used to quickly improve the solution obtained. The basic idea of the hybrid approach is based on the observations that high quality nurse rosters consist of high quality shift sequences. By decomposing the problems into solvable sub-problems for CP, the search space of the original problems are significantly reduced. The results on benchmark problems demonstrate the efficiency of this hybrid CP approach when compared to the state-of-the-art approaches in the literature.

1. Introduction

Due to their complexity and importance in real world modern hospitals, nurse rostering problems (NRPs) have been extensively studied in both Operational Research and Artificial Intelligence societies for more than 40 years [6,10,13]. Most NRPs in real world are NP-hard [16] and are particularly challenging as a large set of different rules and specific nurse preferences need to be satisfied to warrant high quality rosters for nurses in practice. Other wide range of heterogeneous and specific constraints makes the problem over-constrained and hard to solve efficiently.

NRPs consist of generating rosters where required shifts are assigned to nurses over a scheduling period satisfying a number of constraints [6,10]. These constraints are usually defined by regulations, working practices and preferences of nurses in different countries. They are usually categorised into two groups: hard constraints and soft constraints, as defined below:

- *Hard constraints* must be satisfied in order to obtain feasible solutions for use in practice. A roster satisfying all hard constraints is usually termed *feasible*. A common hard constraint is to assign all shifts required to the limited number of nurses.

Rong Qu and Fang He, School of Computer Science, University of Nottingham, Jubilee Campus, Nottingham, NG8 1BB, UK, email: {rxq|fxh}@cs.nott.ac.uk

- *Soft constraints* are not obligatory but are desired to be satisfied as much as possible. In real life, a roster which satisfies all hard and soft constraints usually does not exist. The violations of soft constraints in the roster can thus be used to evaluate the quality of solutions. A common soft constraint in NRPs is to generate rosters with a balanced workload so that human resources are used efficiently.

A wide range of techniques have been investigated in nurse rostering literature. Current approaches include meta-heuristics [3,4,5,12], which are shown to be effective for large scale and complicated real-world problems. These include Tabu search [3,12] and evolutionary algorithms [1,2,4], etc. However, the major drawback of meta-heuristics is that they neither provably produce optimal solutions nor reduce the search space.

AI techniques such as CP [11,17,24,25,26] also form an important subject in nurse rostering research. CP, originated from AI research, is an exact method which guarantees to find feasible solutions for constraint satisfaction problems or optimal solutions for constraint optimization problems. Furthermore, CP has shown its flexibility in expressing heterogeneous and detailed constraints in many of the real life applications such as job-shop problems and vehicle routing problems [15]. As a special type of scheduling problems, the NRP involves assigning resources (nurses) to activities (shifts) periodically while satisfying many side constraints, which are typically handled well by CP techniques [24]. The CP approach is effective in solving small scale problems [15], but is computationally expensive for large scale problems due to the exponential size of the search space when the size of the problem is increased.

Decomposition techniques have been investigated recently in NRPs. In [13], only the pre-defined shift sequences named 'stints' are used to build solutions. In [2], shift patterns with their corresponding penalties are pre-defined, and a Genetic algorithm is employed to build complete solutions. Brucker et al. in [8] presented a decomposition approach where predefined blocks of shifts are cyclically assigned to groups of nurses. The rest of shifts are assigned manually and the resulting roster is further improved by a quick local search. The underlying idea of these decomposition approaches is based on some common features of the high quality rosters - they consist of high quality shift sequences satisfying a set of constraints in the problems.

This paper presents our new attempts to an effective integration between CP and meta-heuristic approaches. The aim is to investigate efficient decomposition on real world NRPs into solvable weekly sub-problems for exact methods, without losing much optimality. In the first stage of this two-stage hybrid approach, the decomposed sub-problem is modeled as a constraint satisfaction problem for which a large number of weekly rosters consist of high quality shift sequences are obtained in seconds by using the CP technique. Then an iterative forward search is used to extend these sub-solutions to complete solutions. In the second stage, a variable neighborhood search is used to further improve the solution.

The paper is organized as follows. In Section 2 we present the problem formulation. The CP model for the problem is presented in Section 3. We give details of the two stage hybrid CP approach in Section 4 and evaluate its efficiency and effectiveness by experiments on benchmark problems in Section 5. Finally we present our conclusions and future work.

2. Problem Formulation

The benchmark NRPs we are tackling are derived from real-world problems in intensive care units at a Dutch hospital. The problem consists of assigning a predefined number of shifts of four types (i.e. early, day, late and night shifts) within a scheduling period of 5 weeks to 16 nurses of different working contracts in a ward. Twelve of the full-time nurses work 36 hours per week. One and other three part-time nurses work maximally 32 and 20 hours per week, respectively. The problems can have a number of variants with respect to the number of nurses, number of shift types, number of skill levels and length of scheduling period, etc., but the main constraints are similar. We define the main problem here and test a number of its variants in the experiments (see Section 5). More details can be found in [9] and at http://www.cs.nott.ac.uk/~tec/NRP/. Table 1 presents the definitions and the daily coverage demand of the four shift types in the problems.

Table 1 Shift types and demand during a week. Each shift covers 9 hours including one hour of resting time, except that the night shift contains no resting time. So there are 8 actual working hours for each of these shift types.

Shift type	Start time	End time	Demand						
			Mon	Tue	Wed	Thu	Fri	Sat	Sun
Early	**07:00**	**16:00**	**3**	**3**	**3**	**3**	**3**	**2**	**2**
Day	**08:00**	**17:00**	**3**	**3**	**3**	**3**	**3**	**2**	**2**
Late	**14:00**	**23:00**	**3**	**3**	**3**	**3**	**3**	**2**	**2**
Night	**23:00**	**07:00**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

The hard constraints listed in Appendix A must be satisfied under any circumstances; otherwise the roster is considered to be unacceptable.

In most nurse rostering literature, hard and soft constraints are considered together when generating and evaluating solutions using different algorithms. In our previous work [9], constraints were categorized into three groups. A large pool of shift sequences of length up to 5 was built offline considering only sequence related constraints. Then complete solutions were composed based on these sequences by using heuristics considering the rest of two groups of constraints.

In this work, we categorize the constraints into two groups: *sequence* and *schedule* constraints, which are considered separately at different steps of the first stage of the hybrid CP approach. In the first step (see Section 4.1.1), only sequence constraints are considered in the constraint satisfaction problem (CSP) model to generate weekly rosters with high quality shift sequences. In the second step (see Section 4.1.2), both sequence and schedule constraints are included in another constraint optimization problem (COP) model to extend the weekly rosters to build complete roster. The two groups of constraints are described follows:

- *Sequence* constraints are applied when generating shift sequences for each nurse within weekly rosters, and
- *Schedule* constraints are applied when the weekly rosters are extended to complete rosters for all nurses.

We also define the following terms that are frequently used in the rest of paper:

- *Shift sequence* is the sequence of shifts assigned to each nurse within weekly rosters;
- *Weekly roster* is the one week roster consists of shift sequences for all nurses;
- *Roster* is the complete assignment of shifts within the scheduling period to all nurses, i.e. the complete solution to the problem.

3. Constraint Programming Model

Different instances of the problem can be defined by the following parameters:

I: set of nurses;

I_t, $t \in \{1,2,3\}$: subset of nurses of different working contracts, i.e. 20, 32 and 36 hours per week, respectively, $I = I_1+I_2+I_3$;

J: number of days within the scheduling period, $J = \{1, 2, \ldots, 35\}$;

W: number of weeks within the scheduling period, $W = \{1, 2, 3, 4, 5\}$;

K: set of shift types $K = \{1(day), 2(early), 3(late), 4(night), 0(off)\}$;

D_{jk}: coverage demand of shift type k on day j (see Section 2), $j \in J$, $k \in K$;

m_i: maximum number of working days for nurse i in the scheduling period;

n_1: maximum number of consecutive *night* shifts in the scheduling period;

n_2: maximum number of consecutive working days in the scheduling period;

n_3: maximum number of working days peer week in the scheduling period;

n_4: minimum number of weekends off in the scheduling period;

g_i / h_i: upper and lower bounds of weekly working days for nurse i;

p_i / q_i: upper and lower bounds of consecutive working days in the scheduling period for nurse i;

3.1 Constraints

CP is a very flexible technique to model a rich set of constraints due to its powerful declarative ability. In this work we use ILOG Solver, a C++ library to model the detailed constraints in NRP. Two types of constraints, *cardinality* and *stretch*, are used in the model.

cardinality constraint is also named as *distribute* (*gcc*, *generalized cardinality*) (see [15] pages 420-450). It bounds the number of variables, each taking of a given set of domain values. It is written as:

cardinality(x/v, l, u) where x is a set of variables (x_1, …, x_m); v is an m-tuple of domain values of the variables x; l and u are m-tuples of nonnegative integers defining the lower and upper bounds of x, respectively. The constraint defines that, for j = 1, …, m, at least l_j and at most u_j of the variables take the value v_j.

So constraint H1 can be easily written as:

$$cardinality(s_{ij}, K, D_{jk}, D_{jk}),\ \forall i \in I, j \in J, k \in K$$

This restricts decision variable s_{ij} taking only values in set K within the bounds of D_{jk}.

stretch constraint (see [15] pages 420-450) is written as:

stretch(x/v, l, u, P) where x is a set of variables (x_1, …, x_m); v is an m-tuple of possible domain values of the variables; l and u are m-tuples of lower and upper bounds for x, respectively. P is a set of patterns, which are pairs of values (v_j, v_j'), requiring that when a stretch of value v_j immediately precedes a stretch of value v_j', the pair (v_j, v_j') must be in P.

A *stretch* is a sequence of consecutive variables that take the same value. That is, $x_j \dots x_k$ is a stretch if for a value v, $x_{j-1} \neq v$, $x_j, \dots, x_k = v$ and $x_{k+1} \neq v$. This constraint also restricts that any stretch of value v_j in x, $j \in \{1, \dots, m\}$, has a length within the range [l_j, u_j]. Thus the *stretch* constraint puts bounds on how many consecutive days a nurse can work each shift, and which shifts can immediately follow another.

For example, constraint H7 can be defined as follows:

$$stretch(s_{ij}, Night, 2, 3, P), P = \{(Night, Off)\}$$

This restricts a nurse having consecutive night shifts within length [2, 3], and the only shift type allowed following the night shift is *off* (as given in P).

Based on the *cardinality* and *stretch* constraints presented above, the other constraints listed in Appendices A and B can be modeled in the same way. For more relevant literature see [15,19,21,22,23].

3.2 Constraint Programming Models

We decompose the problems into weekly sub-problems, and then extend the weekly rosters obtained to complete solutions. Two CP models are thus defined, where different variables and their corresponding domains are given with respect to shift sequences in weekly rosters and complete solutions. The first model is CSP model (subjects to a subset of constraints). It models the decomposed problems where weekly rosters are concerned. The second model is a COP model representing the complete problem, subjects to the complete list of constraints and takes penalty as the objective to minimize.

Model 1

Decision variable s_{ij}: represents the shift assigned to nurse i on day j,

$$s_{ij} \in K, i \in I, j \in \{1, 2, 3, 4, 5, 6, 7\} \quad (1)$$

All the sequence constraints listed in Appendix A and B are concerned:

H1	$cardinality(s_{ij}, K, D_{jk,} D_{jk}),\ \forall\, i \in I, j \in J,\ k \in K$	(2)
H2	this constraint is implicitly satisfied by requiring each constrained variable to take exactly one value.	
H4	$f_1(s_{ij}) \le n_3,\ \forall\, i \in I, j \in J, f_1$ is a counting function	(3)
H5	$cardinality(s_{ij}, Night, 0, n_1),\ \forall\, i \in I, j \in J$	(4)
H7	$stretch(s_{ij}, Night, 2, 3, P), P = \{(Night, Off)\},\ \forall\, i \in I, j \in J$	(5)
H8	$stretch(s_{ij}, Night, 0, n_1, P), P = \{(Night, Off)\},\ \forall\, i \in I, j \in J$	(6)
H9	$stretch(s_{ij}, K, 0, n_2, P)$, $P = \{(Night, Off)\}, (Day, Off)\}, (Early, Off)\}, (Late, Off)\},\ \forall\, i \in I, j \in J$	(7)
S1	$s_{ij_1} = s_{ij_2}, \forall i \in I, j_1 = 6, j_2 = 7, j_1, j_2 \in J$	(8)
S2	$stretch(s_{ij}, K, 2, n_3, P), P = \{(Day, Night), (Day, Early), \ldots\},\ \forall\, i \in I, j \in J$	(9)
S3	this constraint is implicitly satisfied by constraint H7;	
S4	$stretch(s_{ij}, Off, 2, 5, P)$, $P=\{(Night,Off)\},(Day,Off)\},(Early,Off)\},(Late,Off)\}, \forall\, i \in I, j \in J$	(10)
S5	$h_i \le\ f_1\,(s_{ij}) \le g_i,\ \forall\, i \in I, j \in J$	(11)
S9	this constraint can be expressed using a boolean implication constraint, which reflects a boolean logical relation between two variables as follows:	
	$s_{ij} = Day\ \longrightarrow s_{i(j+1)} \ne Early,\ \forall\, i \in I, j \in J\text{-}1$	(12)
	$s_{ij} = Late\ \longrightarrow s_{i(j+1)} \ne Early,\ \forall\, i \in I, j \in J\text{-}1$	(13)
	$s_{ij} = Late\ \longrightarrow s_{i(j+1)} \ne Day,\ \forall\, i \in I, j \in\ J\text{-}1$	(14)
S10	$s_{ij} = Early\ \longrightarrow s_{i(j+1)} \ne Night,\ \forall\, i \in I, j \in J\ \text{-}1$	(15)

Model 2

Decision variable s_{iw}: represents the shift sequence of one week length assigned to nurse i in week w. The domain of variables is the permutations of the shift sequences generated by the first model, i.e. $\{(0011444), (4400022), \ldots\}$. All the sequence constraints and schedule constraints are concerned in this model subject to decision variables s_{ij} in Model 1 and s_{iw}. The model is presented as follows:

Objective:

$$Minimize\ \sum w_i P(x_i)$$

where $P(x_i)$ is a function representing if soft constraint i is violated in the roster. w_i is the weight of soft constraint i.

Subject to: all the sequence constraints in Model 1 and schedule constraints listed in Appendix A and B as follows:

H3	$f_1\,(s_{ij}) \le m_i,\ \forall\, i \in I$	(16)
H6	$cardinality(s_{ij'}, Off, n_4, W),\ \forall\, i \in I, j' = \lceil j\,/\,6 \rceil, j' \in J$	(17)
S6	$q_i \le f_1\,(\,s_{ij}) \le p_i,\ \forall\, i \in I, j \in J$	(18)
S7	$stretch(s_{ij}, Early, 2, 3, P), P = \{(Early, Off)\}, (Early, Day)\}, \ldots\ \},\ \forall\, i \in I, j \in J$	(19)
S8	$stretch(s_{ij}, Late, 2, 3, P), P = \{(Late, Off)\}, (Late, Day)\}, \ldots\ \},\ \forall\, i \in I, j \in J$	(20)

4. A Hybrid CP Approach to NRP

The problem we are solving has a search space of $4^{16\times5}$ (i.e. 1.424E337), for which a systematically tree search is computationally expensive and cannot provide a solution even after one day. We thus investigate a two-stage hybrid CP approach:

- Stage I: Weekly rosters are built by using the CSP model 1. The iterative forward search is used to extend the weekly rosters to complete solutions using the COP model 2.
- Stage II: A variable neighborhood search is then used to improve the solution built from stage I.

4.1 Stage I: Constraint Programming and Iterative Forward Search

4.1.1 Weekly roster construction

Weekly rosters which consist of high quality shift sequences are firstly generated by CSP Model 1 defined in Section 3.2. The algorithm used is a systematic backtracking depth first search. The *first-fail* principle is used as the variable order heuristic. One illustrative example of weekly roster (of overall penalty 0) generated by CSP Model 1 is given in Table 2. These shift sequences for each nurse satisfy all the sequence constraints, so are of high quality and are desired to be preserved in the final complete solution. By using CSP Model 1, thousands of weekly rosters can be generated in seconds (8.7E5 approximately, see experiments in Section 5). We randomly select 50 initial weekly rosters to build complete solutions by using the iterative forward search.

Table 2 An illustrative example of weekly (partial) roster ("O": no shift assigned; N: night; E: early; D: day)

	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Cost
Nurse 1	O	O	D	D	N	N	N	0
Nurse 2	N	N	O	O	O	E	E	0
…	…							

4.1.2 Roster construction by Iterative Forward Search

Iterative forward search [18] works upon feasible incomplete solution (weekly rosters generated by the above step). It iteratively extends these blocks into a complete solution. Figure 1 presents the pseudo code of the search algorithm.

The algorithm extends the current partial solutions by assigning values to variables until all the variables have been assigned values. If succeed, the one-week roster will be extended to two-week roster and continue in the same way. The number of outside iterations corresponds to the number of weeks in the whole roster (4 iterations to build 5 weeks' roster in the problem here). The inside iterations of the procedure assign values to variables iteratively. When a conflict occurs after a value has been assigned to a variable, the latest variable is un-assign and another value is tried (backtracking). If all the values have been tried and the search cannot continue consistently, the search starts from the outside iteration and attempts another initial weekly roster block to continue.

```
Procedure IFS (initial weekly roster block i = 1)
  outside iteration repeat
    iteration = 0;
    current solution = initial weekly roster i;
    inside iteration repeat
      select variable and value;           //with or without heuristic selection
      assign value to variable;
      current solution = initial weekly roster i + assigned variable;
      un-assign conflict variable;
    until(allWeeklyVariableAssigned)
    if(canContinue(initial weekly roster i))
      iteration = iteration + 1;
    else
      initial weekly roster block i = i + 1;
  until(allVariableAssigned)
  complete solution = current solution
end procedure
```

Figure 1 Pseudo-code of the iterative forward search algorithm

The above algorithm is parameterized by two heuristics, variable selection and value selection heuristics. In this work we compare these two heuristics with a random rule and evaluate their effects within our hybrid CP approach:

1. Randomly select variables and values during the search
2. Select variables and values by heuristics:
   a) Variable selection heuristic: first-fail principle, by which nurses with heavier workload from previous iteration is selected first;
   b) Value selection heuristic: night shift sequences first.

The variable selection heuristic chooses next variable in the search based on the information collected in the previous iterations of the search. Shift sequences assigned to each nurse are recorded and the nurses are ranked by their workloads. The heavier workload the nurses have received, the more likely a conflict will occur later with respect to the workload constraint. Therefore we follow the first-fail principle to consider the heavier workload nurses first in the next step search.

The night shift is the most important and complicated shift in the problems, due to the fact that it is involved in a number of hard constraints (H5, H7, and H9) and soft constraints (S2, S3) with high costs of 1000. Therefore we assign night shift sequences first. The rest of the sequences are of the same importance and are randomly selected and assigned to the nurses.

4.2 Stage II: Variable Neighborhood Search

A simple Variable Neighborhood Descent [14] is applied to further improve the solution built from Stage I. Two neighborhoods structures are employed in the algorithm; both have been widely used in meta-heuristics in the nurse rostering literature [7]. Note that this work concerns mainly a hybrid CP approach rather than designing elaborated meta-heuristics. The two neighborhoods are defined by the following moves upon a complete roster:

- Neighborhood structure 1: re-assign a shift to a different nurse working on the same day.
- Neighborhood structure 2: swap shifts assigned to two nurses on the same day.

Initialization select neighborhood structures N_k, $k = 1,2...k_{max}$; construct an initial solution x;
Repeat until no improvement is obtained:
(1) Select k = 1;
(2) Repeat the following steps until $k = k_{max}$:
(a) Explore to find the best neighbor x' of x ($x' \in N_k(x)$);
(b) Move or not. If the solution thus obtained x' is better than x, set x = x' and k = 1; otherwise, set k = k + 1;

Figure 2 Pseudo-code of the Variable neighborhood Search algorithm (see [14])

The pseudo-code of the variable neighbourhood search is presented in Fig. 2. The neighborhoods (by the smaller neighborhood structure 1) are repeatedly examined for possible improving moves. When there are no improving moves by using neighborhood structure 1, neighborhoods by larger neighborhood structure 2 are examined. Then the search switches back to neighborhood structure 1 again. This process is repeated until there are no improving moves left by using both neighborhood structures 1 and 2.

The variable neighborhood search searches upon feasible solutions built from the first stage. The feasibility of the solutions is preserved during the search by considering all the constraints in the problem.

5. Experimental Results

We evaluate our hybrid CP approach upon a set of benchmark nurse rostering problem instances, pubic available at http://www.cs.nott.ac.uk/~tec/NRP. They are monocyclic problems and different from each other with respect to parameters such as the number of nurses, number of shift types and the length of scheduling period. Table 3 presents the characteristics of the problems we use in this paper. For all problems, 6 runs are carried out on an Intel Core 1.86GHz machine with 1.97GB memory, from which average results are presented.

Table 3 Characteristics of the benchmark nurse rostering problems (instances ORTEC#1 to #4 are similar instances with some differences on some constraints)

	A	B	C	GPOST	ORTEC#1-#4	ORTEC#Jan-#Dec
Number of Shift types	2	2	2	2	4	4
Number of Nurses	8	8	8	8	16	16
Period of Schedule(day)	7	28	28	28	35	35
Number of Skill Levels	1	2	2	1	1	1

Experiment I. Direct CP and Hybrid CP Approaches. We first evaluate the hybrid CP approach compared to the direct CP approach on the 5 benchmark problems presented in Table 3. Here the termed direct CP approach uses a complete COP model where all constraints in Appendices A and B are included to solve this set of problems of the original size without decomposition. The depth-first Branch-and-Bound search is used as the search algorithm. Table 4 presents the results (i.e. violations of soft constraints, see Objective in COP model 2) and demonstrates their abilities to handle constraints in different problems. The column "problem size" in the table gives the number of variables and number of constraints in the CP model. It is observed that the direct CP approach can handle only small scale instances (measured by the number of variables and constraints) but cannot produce solutions for large scale instances even after 24 hours of running time. The hybrid CP approach can obtain results for all these large scale instances within 1 hour.

Table 4 Results of direct CP and hybrid CP approaches on nurse rostering problems of different characteristics. "-" indicates that no solutions can be obtained within 24 hours.

Data	**Problem Size**		**Direct CP**	**Hybrid CP**
	Variables	**Constraints**	**(within 1 hour)**	**(within 1 hour)**
A	722	2109	8	8
B	3460	4600	0	0
C	3639	4612	10	10
GPOST	7897	5866	2	2
ORTEC#1	6672	22380	-	616
ORTEC#2	8208	28562	-	786
ORTEC#3	8624	29108	-	650
ORTEC#4	8720	29234	-	616

Experiment II. Variable and Value Selection in the Hybrid CP Approach. Another set of experiments is carried out to evaluate the effect of variable and value selection heuristics within the hybrid CP approach upon problem instances ORTEC#1-4 in Table 3. It is observed that random selection rule can easily cause a large number of violations to the high penalty constraints. The solutions produced by using this rule cannot be further improved in the second stage, mainly due to the bad assignments of night shifts. Table 5 presents the results of the hybrid CP approach by using different variable and value selection rules. Both of them can obtain results within 1 hour. The hybrid CP approach with variable and value heuristic selection heuristic within the iterative forward search produces better results for 3 out of 4 problems. For the other problem, the difference is small.

Table 5 Results with random and heuristic variable and value selection rules in the hybrid CP approach

Problem	**Random Selection**	**Heuristic Selection**
ORTEC#1	1686	**616**
ORTEC#2	1035	**786**
ORTEC#3	635	650
ORTEC#4	705	**616**

Most search algorithms for solving CSP search systematically through all the possible assignments of values to variables with backtracking when a dead-end is reached (no valid value can be assigned to the variable in consideration). The main drawback of such backtrack-based search is that they typically make mistakes in the early stage of search, i.e. a wrong early assignment can cause a whole sub tree to be explored with no success. With the guidance of variable and value heuristic selection, the hybrid CP approach is capable of producing better results within a limited time.

Experiment III. **The Hybrid CP Approach on Large Scale Benchmarks.** Table 6 presents the results from the hybrid CP approach compared with those from other current approaches on twelve large real-world NRP instances (ORTEC#Jan-#Dec in Table 3). The first approach is a hybrid Genetic Algorithm which has been developed by ORTEC, Netherlands in the commercialised software Harmony^{TM} [20]. The second approach is a hybrid Variable Neighbourhood Search with a heuristic ordering as the construction method [7].

Table 6 Results from our hybrid CP approach, compared to current approaches in the literature, best results in bold.

Problem instances ORTEC#Jan-#Dec	**Hybrid GA [20] (1 hour)**	**Hybrid VNS [7] (1 hour)**	**Hybrid CP approach (½ hour)**
Jan	775	735	**616**
Feb	1791	1866	**1736**
Mar	2030	**2010**	2766
Apr	612	**457**	956
May	2296	2161	**1786**
Jun	9466	9291	**8700**
Jul	781	**481**	650
Aug	4850	4880	**2171**
Sep	**615**	647	1300
Oct	736	665	**616**
Nov	2126	2030	**1620**
Dec	625	520	**496**
Average	2225	2145	**1951**

In our hybrid approach, CP in the first stage generates weekly rosters in a short time (on average of 370 seconds, depending on the number of constraints in the model). These blocks are permutations of high quality shift sequences (the total number is up to 8.7E5 approximately). The iterative forward search procedure with COP Model 2 terminates after a complete solution is found. Then the simple variable neighbourhood search obtains the improved solution within 1 minute. The overall process takes up to 30 minutes. We have also test the performance of our hybrid CP approach in longer running time either by extending the number of initial solutions or allowing extra running time in Stage II for improvement. It is observed that the extra number of initial solutions have no impact upon the final solution mainly due to that all the selected initial solutions are of the same quality (of overall penalty 0). The Variable Neighborhood Search usually improve the initial solutions within minutes; longer running time did not show significant improvement.

The results in Table 6 demonstrate the efficiency of our hybrid CP approach developed in this work. Within a much shorter computational time, our hybrid CP approach obtained the best results for 8 out of 12 problems compared to the current best approaches in the literature. The overall performance of the hybrid CP approach (average results across all 12 problems) ranked the best compared to the other approaches.

6. Conclusion and Future Work

In this paper we developed an efficient hybrid approach integrating constraint programming and meta-heuristic methods applied to the large scale benchmark problems of nurse rostering. Constraint programming is used to efficiently build weekly rosters consisting of high quality shift sequences. An iterative forward search is adopted to extend the partial rosters to complete solutions. Based on these initial solutions, a simple variable neighbourhood search is used to quickly improve the solutions obtained. The effective integration between constraint programming techniques and meta-heuristic algorithm leads to an algorithm that overcomes the inherent weaknesses of each approach. The numerical results on both small scale problems and a set of large scale real world benchmark problems of different problem characteristics demonstrated the efficiency and effectiveness of the hybrid CP approach in comparison to the direct constraint programming approach and other two current hybrid algorithms with respect to both solution quality and computational time. The hybrid CP approach produced overall better results than the best results in the literature on the benchmarks.

Future work will investigate other efficient ways of integrating constraint programming and meta-heuristics in tackling highly constrained nurse rostering problems. Hybridising exact methods such as constraint programming and integer programming techniques is another interesting and challenging direction on other highly constrained optimization problems.

References

[1] U. Aickelin and K. Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem, *Journal of Scheduling*, 3(3): 139-153, 2000

[2] U. Aickelin and K. Dowsland. An indirect genetic algorithm for a nurse scheduling problem, *Journal of Operations Research Society*, 31(5): 761-778, 2003

[3] E. K. Burke, P. De Causmaecker and G. Vanden Berghe. A hybrid tabu search algorithm for the nurse rostering problem. Lecture Notes in Artificial Intelligence, vol. 1585, 187-194, 1998

[4] E. K. Burke, P. Cowling, P. De Causmaecker and G. Vanden Berghe. A memetic approach to the nurse rostering problem, *Applied Intelligence*, 15,119-214, 2001

[5] E. K. Burke, P. De Causmaecker, S. Petrovic and G. Vanden Berghe. Variable neighborhood search for nurse rostering problems. In: M.G.C. Resende and J.P. de Sousa (eds.), *Metaheuristics: Computer Decision-Making* (Combinatorial Optimization Book Series), Kluwer, 153-172, 2003

[6] E .K. Burke, P. De Causmaecker, G. Vanden Berghe and H. Van Landeghem. The state of the art of nurse rostering, *Journal of Scheduling*, 7, 441-499, 2004

[7] E. K. Burke, T. E. Curtois, G. Post, R. Qu and B. Veltman. A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem, *European Journal of Operational Research*, 2, 330-341, 2008.

[8] P. Brucker, R. Qu, E. K. Burke and G. Post. A decomposition, construction and post-processing approach for a specific nurse rostering problem, MISTA'05, 397-406. New York, USA, Jul 2005

[9] P. Brucker, E.K. Burke, T. Curtois, R. Qu and G. Vanden Berghe. Adaptive construction of nurse schedules: A shift sequence based approach. Under review at *European Journal of Operational Research*. Technical Report NOTTCS-TR-2007-1, School of Computer Science, University of Nottingham

[10] B. Cheang, H. Li, A. Lim and B. Rodrigues. Nurse rostering problems - a bibliographic survey, *European Journal of Operational Research*, 151, 447-460, 2003

[11] B. M. W. Cheng, J. H. M. Lee, and J. A. C. K. Wu. A nurse rostering system using constraint programming and redundant modelling, *IEEE Transactions on information technology in biomedicine*, vol. 1(1), 44-54, 1997

[12] K. Dowsland. Nurse scheduling with tabu search and strategic oscillation, *European Journal of Operational Research*, 106, 393-407, 1998

[13] A.T. Ernst, H. Jiang, M. Krishnamoorthy and D. Sier. Staff scheduling and rostering: A review of applications, methods and models, *European Journal of Operational Research*, 153, 3-27, 2004

[14] P.Hansen, N. Mladenovic, Variable Neighborhood Search: Principles and Applications, *European Journal of Operational Research*, 130, 449-467, 2001

[15] J. Hooke, *Integrated Methods for Optimization*, Springer, 2006

[16] R. M. Karp. Reducibility among combinatorial problems, In: R.E. Miller and J.W.Thatcher (eds.), Complexity of Computer Computations, Plenum Press, New York, 85-103, 1972

[17] H. Meyer auf'm Hofe. Solving rostering tasks as constraint optimisation. In: Burke E.K. and W. Erben (eds.) Lecture Notes in Computer Science, vol. 2079, 280-297, 2000, Springer, Berlin

[18] T. Muller, R. Bartak and H. Rudova. Iterative forward search algorithm: combining local search with maintaining arc consistency and a conflict-based statistics. Lecture Notes in Computer Science, Vol 3258, 802-817, 2004, Springer, Berlin

[19] G. Pesant. A filtering algorithm for the stretch constraint. In: Principles and Practice of Constraint Programming- CP01: The Proceedings of the Seventh International Conference, Lecture Notes in Computer Science, Vol. 2239, 183-195, 2001, Springer, Berlin

[20] G. Post and B. Veltman. Harmonious personnel scheduling. In: E. Burke and M. Trick (eds.), Proceedings of the 5th International Conference on Practice and Automated Timetabling (PATAT'04), 557-559, 2004

[21] C. G. Quimper, A. Lopez-Ortiz, P. van Beek, and A. Golynski. Improved algorithms for the global cardinality constraint. In: M. Wallace (editor), Principles and Practice of Constraint Programming (CP 2004), Lecture Notes in Computer Science, Vol 3258, 542-556, 2004, Springer, Berlin

[22] J. C. Regin. Generalized arc consistency for global cardinality constraint. National Conference on Artificial Intelligence (AAAI 1996), AAAI Press, 209-215, 1996

[23] J. C. Regin and J. F. Puget. A filtering algorithm for global sequencing constraint, in: Principles and Practice of Constraint Programming -CP97: The Proceedings of the Third International Conference, Lecture Notes in Computer Science, Vol. 1330, 32-46, 1997, Springer, Berlin

[24] L. M. Rousseau, M. Gendreau and G. Pesant. A general approach to the physician rostering problems, *Annals of Operations Research*, 115, 193-205, 2002

[25] M. Sellmann. Crew assignment via constraint programming: integrating column generation and heuristic tree search, *Annals of Operations Research*, 115, 207-225, 2002
[26] G.Y.C. Wonga and A.H.W. Chun. Constraint-based rostering using meta-level reasoning and probability-based ordering, *Engineering Applications of Artificial Intelligence*, 17, 599-610, 2004

Appendix A. The list of hard constraints [9]

	Hard constraints	Type
1	Demand needs to be fulfilled (i.e. all the requested shifts in Table 1 must be covered).	sequence
2	For each day, one nurse may start only one shift.	sequence
3	Within a scheduling period, a nurse is allowed to exceed the number of hours for which he/she is available for his/her department by at most 4 hours.	schedule
4	The maximum labor time per week is on average 36 hours over a period of 13 consecutive weeks if this period does not include work during night shifts.	sequence
5	The maximum number of night shifts is 3 per period of 5 consecutive weeks.	sequence
6	A nurse must receive at least 2 weekends off duty per 5 week period. A weekend off duty lasts 60 hours including Saturday 00:00 to Monday 04:00.	schedule
7	Following a series of at least 2 consecutive night shifts, a 42 hours rest is required.	sequence
8	The number of consecutive night shifts is at most 3.	sequence
9	The number of consecutive shifts (workdays) is at most 6.	either*

Appendix B. The list of soft constraints [9]

	Soft constraints	Weights	Type
1	For the period of Friday 23:00 to Monday 0:00, a nurse should have either no shifts or at least 2 shifts (Complete Weekend).	1000	sequence
2	Avoid sequence of shifts with length of 1 for all nurses.	1000	sequence
3a	For nurses with availability of 30-36 hours per week, the length of a series of *night* shifts should be within the range [2, 3]. It could be part of, but not before, another sequence of shifts.	1000	sequence
3b	For nurses with availability of 0-30 hours per week, the length of a series of *night* shifts should be within the range [2, 3]. It could be part of, but not before, another sequence of shifts.	1000	sequence
4	The rest after a series of *day*, *early* or *late* shifts is at least 2 days.	100	sequence
5a	For nurses with availability of 30-36 hours per week, the number of shifts is within the range [4, 5] per week.	10	sequence
5b	For nurses with availability of 0-30 hours per week, the number of shifts is within the range [2, 3] per week.	10	sequence
6a	For nurses with availability of 30-36 hours per week, the length of a series of shifts should be within the range of [4, 6].	10	schedule
6b	For nurses with availability of 0-30 hours per week, the length of a series of shifts should be within the range [2, 3].	10	schedule
7	For all nurse, the length of a series of *early* shifts should be within the range [2, 3]. It could be within another series of shifts.	10	schedule
8	For all nurse the length of a series of *late* shifts should be within the range of [2, 3]. It could be within another series of shifts.	10	schedule
9a	An *early* shift after a *day* shift should be avoided.	5	either*
9b	An *early* shift after a *late* shift should be avoided.	5	either*
9c	A *day* shift after a *late* shift should be avoided.	5	either*
10	A *night* shift after an *early* shift should be avoided.	1	either*

'either' indicates that the corresponding constraints could be either sequence or schedule constraints.