A Harmony Search Algorithm for Nurse Rostering Problems

Mohammed Hadwan[1], Masri Ayob[1], Nasser R. Sabar[1] and Roug Qu[2]

Data Mining and Optimisation Research Group (DMO), Centre for Artificial Intelligent (CAIT),

Universiti Kebangsaan Malaysia, 43600 UKM Bangi Selangor, Malaysia.

hadwan79@yahoo.com, masri@ftsm.ukm.my and naserdolayme@yahoo.com

[2]ASAP Research Group, School of Computer Science The University of Nottingham, Nottingham NG8 1BB, UK.

rxq@cs.nott.ac.uk

ABSTRACT. Harmony Search Algorithm (HSA) is a relatively new nature-inspired algorithm. It evolves solutions in the problem search space by mimicking the musical improvisation process in seeking agreeable harmony measured by aesthetic standards. The Nurse Rostering Problem (NRP) is a well-known NP-hard scheduling problem that aims at allocating the required workload to the available staff nurses at healthcare organizations to meet the operational requirements and a range of preferences. This work investigates research issues of the parameter settings in HSA and application of HSA to effectively solve complex NRPs. Due to the well-known fact that most NRPs algorithms are highly problem (or even instance) dependent, the performance of our proposed HSA is evaluated on two sets of very different nurse rostering problems. The first set represents a real world dataset obtained from a large hospital in Malaysia. Experimental results show that our proposed HSA produces better quality rosters for all considered instances than a genetic algorithm (implemented herein). The second is a set of well-known benchmark NRPs which are widely used by researchers in the literature. The proposed HSA obtains good results (and new lower bound for a few instances) when compared to the current state of the art of meta-heuristic algorithms in recent literature.

Keywords: Harmony search, Meta-heuristic, Timetabling and Personnel scheduling

1. INTRODUCTION

Nurse rostering problems (NRPs) are highly constrained combinatorial problems which are difficult to be solved to optimality [1]. Scheduling the nurses at healthcare organizations is a challenging task. Extra care needs to be taken as personnel of healthcare organizations consume about 40% of hospital budgets [2]. Bad or inflexible duty roster can affect the personal life of staff nurses, increase job dissatisfaction, and thus result in high staff turnover. These have a direct impact on the nursing services provided to patients [3, 2]. Based on the literature in NRPs, a lot of big healthcare organizations around the world still construct nurses' duty roster manually [3-5]. This raises the challenge for researchers to propose and investigate automated solution methodologies to solve this problem.

Various methods have been proposed to solve NRPs, including exact methods, meta-heuristics and others [3]. Although exact methods (mathematical based models) can obtain optimal solutions, they showed to be highly inflexible in solving large scale optimization problems due to the computational time required. Meta-heuristic approaches which are able to produce solutions of good quality for difficult combinatorial optimization problems within a reasonable computational time are thus usually preferable as the solution methodology [3]. Meta-heuristics refer to the approaches that mimic some of the natural and artificial

phenomena that combine “rules of thumb” and randomness to solve difficult problems [6, 7]. Well-known examples of meta-heuristic algorithms include genetic algorithms (GAs), simulated annealing (SA), tabu search (TS), to name a few [7] . During the last few decades, meta-heuristic approaches and their hybrids have been successfully applied to solve NRPs [3].

NRPs have gained the interest of many researchers for more than four decades [3]. Several comprehensive surveys [3, 8, 9] have reviewed a large number of published papers on NRPs. NRPs are well known over-constrained problems [9, 10]. They are very difficult to solve manually due to their nature of large number of often conflicting objectives that must be taken into consideration while constructing the duty roster, i.e. different types of nurses, different shift types, different coverage demand of shifts each day, popular/unpopular shifts and many more issues [3, 11]. Many optimization algorithms have been proposed to solve NRPs with different constraints. Some showed to be superior in solving some of the instances but not the others. Various population based and local search based meta-heuristic algorithms developed to solve NRPs include GAs [12, 13], TS [14, 15], SA [16, 17] and many others. In [18], the performance of scatter search algorithm and memetic algorithm (population-based) were better compared to variable neighborhood search and tabu search algorithms (local search-based) when tested against the presented benchmark datasets. This encouraged us to investigate the use of other population-based methods for NRPs.

In this work, we adapt harmony search algorithm (HSA), which was developed by Geem et al [19] to solve combinatorial optimization problems, to solve NRPs. It mimics the analogy of the natural process of musical improvisation that searches for suitable musical notes. The idea is adapted in the search process for solving optimization problems [19]. HSA can be categorized as a recent evolutionary algorithm and showed to be efficient in solving difficult optimization problems such as university course timetabling [20], vehicle routing [21], Sudoku Puzzle [22], and many others [23-27].

Compared to the traditional optimization methods, HSA has some features that motivate our investigation to solve NRPs. These include [28]: (i) HSA has few parameters that need to be tuned and thus could be easily adapted to different NPRs or instances. These parameters do not require too much tuning effort to obtain high quality solutions [29]. (ii) HSA is a stochastic random search method. (iii) HSA overcomes the drawback of building block theory of GAs by considering all existing solutions, whilst GAs consider only two solutions (parents) in reproduction [29]. To our knowledge, there is only limited published work on HSA to solve NRPs [30], where a basic HSA has been evaluated on only the small instances established by the International Nurse Rostering Competition 2010 (INRC2010). How HSA will perform on large or complex NPRs instances has not been investigated. In addition, the parameter values of HSA in [30] were arbitrary chosen. It is well known that the performance of many population based methods is highly dependent on its parameter values [31]. Therefore, the aim of this work is to address research issues in applying HSA to NRPs by intensive investigations on suitable parameter values and performance evaluation on a large number of instances of problems with different size and complexity.

In order to reduce the gap between practical problems and academic theories, a real world dataset obtained from a large hospital in Malaysia (UKMMC) is used to assess the performance of the proposed HSA. The performance of HSA is also evaluated on the nurse rostering benchmark problems (www.cs.nott.ac.uk/~tec/NRP/) [32]. Computational results on both problems demonstrate the efficiency and effectiveness of the proposed HSA in

producing high quality solutions in a shorter time compared to a basic genetic algorithm for the UKMMC problem, and obtaining good results compared to some existing meta-heuristic algorithms for the same benchmark NRPs. For more resources and application areas of HSA, please refer to [33, 34].

The rest of the paper is organized as follows. Section 2 presents the problem description. The proposed harmony search algorithm is presented in Section 3. Section 4 presents the computational results and analysis. Finally, concluding remarks are presented in Section 5.

2. PROBLEM DESCRIPTION

NRPs can be defined as allocating the workload to the available staff nurses at healthcare organizations to meet the operational requirements [3]. More precisely, given a set of nurses of specific categories, a set of pre-defined periods (shifts) on a working day, and a set of working days; the aim is to assign each nurse to specific planning periods satisfying some constrains (known as hard and soft constraints). Hard constraints are mandatory whereas soft constraints can be violated if necessary [3, 9]. The quality of the generated roster can therefore be assessed based on how many soft constraints have been satisfied. Due to the variety in hard and soft constraints, which are different from one organization to another, the modeling and implementation process present to be challenging tasks, as a unique general mathematical model to accommodate all related constraints does not exist [35] [1,3,9]. As a result, different instances require different implementations and configurations in designing the algorithms. There are, however, a group of problems represented with a common XML structure but associated with different constraints and objectives [26]. Some effort has also been made in the literature to construct relatively general frameworks [31, 32]. Therefore, in this work we have attempted to solve real world and benchmark NRPs because we believe that HSA can be applied to solve a wide range of NRPs. The descriptions of the two problems concerned are provided as follows.

2.1 The UKMMC Nurse Rostering Problem

To bridge the gap between academic research and real world practice, this work considers eight different rostering problem scenarios faced by UKMMC, see Table 6 in Section 4.1. UKMMC is a Malaysian public hospital with more than 1400 nurses working around the clock [36]. In this section, the list of hard and soft constraints, the evaluation function and constraint weightings are discussed. For more details about the problem at UKMMC, we refer readers to our previous work in [16, 36].

(i) Hard Constraints

For the UKMMC NRP, we have identified the following hard constraints:

(HC1) The coverage demand for each shift type must be fulfilled. Under staffing is not allowed.
(HC2) All nurses work at most one shift per day.
(HC3) One senior nurse must be allocated for every shift type.
(HC4) An isolated working day is prohibited. That is a working day with a day off before and after the day is not allowed.

(HC5) Within each 14 days, the maximum working days are 12 days whilst the minimum are 10 days.
(HC6) Each nurse works no more than 4 consecutive working days.
(HC7) Night shift must be in the form of 4 consecutive night shifts followed by two days off.

(ii) Soft Constraints

Soft constraints reflect the general preferences of staff nurses and hospital's requirements at UKMMC. The quality of the constructed duty roster is dependent on how far we can satisfy those soft constraints. The weights (importance) of soft constraints in the UKMMC NRP are described in Table 1.

Table 1 The weights of soft constraints

	Soft Constraints	**Penalty**
(SC1)	Assigns equal number of working days and days off to all staff nurses during the rostering period, i.e. fairness	100
(SC2)	Assigns each nurse at least one day off in the weekends during the rostering period	100
(SC3)	Assigns four consecutive morning shifts followed by one day off	10
(SC4)	Assigns four consecutive evening shifts followed by one day off	10
(SC5)	Assigns either a day off (O) or an evening (E) shift after two days off that follow the night (N) shift pattern, i.e. (NNNNOOE) or (NNNNOOO)	1

For the UKMMC dataset we have four desirable patterns (DPs). These patterns are the patterns that are not violating any of the hard constraints or soft constraints. The more the DPs are given to the nurses the more the nurses' preferences will be satisfied. These desirable patterns are MMMMo, EEEEo, NNNNooo and NNNNooE.

(iii) The Objective Function and Constraint Weightings

Mostly, the main goal of solving NRPs is to efficiently utilize the available staff nurses by producing a well-balanced duty roster that also satisfies nurses' preferences (in general) [37]. The objective function of this work is to minimize the total penalty of soft constraint violations while making sure all hard constraints are satisfied. Each soft constraint is associated with a weight that represents its importance. Constraints with higher weights are more important to be satisfied, thus cause higher penalty if violated. One of the issues regarding setting the weights of constraints in NRPs is that, there are no standard weights to be given for each soft constraint. This is due to the wide range of constraints that are different from one hospital to another. Therefore, like in [12, 38], we have assigned weights for our soft constraints based on the consultation with the head nurses in different wards at UKMMC. Table 1 presents the weight of each soft constraint.

2.2 Benchmark NRP Problems

The second set of NRPs are taken from the NRP benchmark web site [32] to validate our results against the existing literature. Since there are too many NRP datasets from different universities and different hospitals around the world, testing all these datasets become a tedious and difficult task. The reasons behind choosing the datasets in this paper are as follows: (i) in the literature these datasets have been widely studied using other meta-heuristic algorithms, (ii) the best known results are publicly available, and (iii) the various

difficulties of these benchmark datasets with different number of nurses, days and shift types, etc., provide an excellent test bed for evaluating algorithm performance.

Each problem of the selected dataset can be seen as a new and different optimization problem rather than different instances with different size, see Table 13 in Section 4.2. This is mainly because each problem is derived from a different organization, with different constraints, rules and requirements. The common feature between these problems is that some constraints appear in most of the problems. Due to the variety of problem constraints, standard mathematical model for all instances does not exist [3]. Therefore, the problem description, mathematical formation and the objective function of the considered problems are taken from [39] and the implementation was based on the framework introduced in [40]. Please note that in the literature, no work has tested all problems at [32]. Most of the algorithms are evaluated on a single or several problems at the web site.

3. THE HARMONY SEARCH ALGORITHM FOR NRPS

HSA is one of the population-based meta-heuristic algorithms, inspired by musical improvisation [41]. In music performance, each musician plays one musical note at a time. Those musical notes are combined together to form a harmony, measured by aesthetic standards. In optimization, each variable during the optimization process is assigned a value at a time; those values all together form a solution for the considered problem, evaluated by the objective function [33].

Similar to a group of musicians developing their harmonies iteratively, HSA improves solutions iteratively based on good candidate solutions from the initial population, i.e. the *harmony memory* [19]. It carries out a stochastic random search on the solution, i.e. a vector of decision variables, via a number of improvisations. The harmony memory is updated between improvisations. At each improvisation, stored values of decision variables in the harmony memory are adapted according to a *considering rate*. Variable values in the solution are adjusted according to a *pitch adjusting rate*. HSA does not require any starting values of the decision variables nor does it require complex derivatives to adjust the variable values for the new generated solutions [29]. Motivated by these features, this work adapts HSA for NRPs. The procedure of HSA has the following five main steps:

Step 1: Initialize the parameters of HSA.

Step 2: Build the harmony memory (HM).

Step 3: Improvise new solutions.

Step 4: Update the harmony memory (HM).

Step 5: Repeat steps 3 and 4 until reaching the stopping criteria.

Each of these mentioned steps in our proposed HSA will be discussed in details in the following subsections.

3.1 Initialize the Parameters of HSA

In step 1, HSA parameters are initialized as follows:

1. Harmony memory size (HMS) is the number of solutions that are stored in the HM. HMS is similar to the population size in genetic algorithms.

2. Harmony Memory Considering Rate (HMCR) is used during the improvisation process to decide whether the variables of the solution should take the value of any one in HM. HMCR takes a value in the range [0, 1]. It is similar to the crossover rate in genetic algorithms. For example, if HMCR is 0.9 it means that the probability of choosing the value of the variable from HM is 90%; whilst the probability of choosing a value randomly from the domain of the variable is 10%, i.e. (1 – HMCR). In our case, the domain of the variable refers to all the possible shift patterns that HSA can choose from. Selecting a random value for the variable at probability of (1 – HMCR) is similar to the mutation operator in genetic algorithms.
3. Pitch Adjusting Rate (PAR) is also used during the improvisation process to decide whether the variable of the solution should be changed to a neighbor value. PAR takes a value in the range [0, 1]. The amount of change is determined by the bandwidth to move the solution from one neighbor to another. The value of the bandwidth is randomly chosen from its domain, and used to change one shift pattern for a nurse. For example, if PAR is 0.3 it means that the probability of changing the variable value is 30%; whilst 70%, i.e. (1 – PAR), is the probability of keeping the variable without any change. PAR is similar to a local search algorithm which accepts only improving solutions.
4. The maximum number of improvisations (NI) in the search represents the number of iterations in HSA.

In this work, we investigate the suitable values for HSA parameters including HMS, HMCR, PAR and NI.

3.2 Build the Harmony Memory (HM)

In Step 2, a set of initial solutions of size HMS are generated to construct the HM. HM is represented by a matrix of two dimensions. Rows compromise a set of solutions x^i (duty rosters), whilst columns compromise the variables of each solution (nurses). Each solution x^i can be seen as a one dimensional array. The size of the array is the maximum number of nurses in the considered problem instance. For highly constrained NP-hard problems the solution space is very huge and the possible solutions vary significantly [42]. One way to deal with this huge solution search space is to divide and conquer, i.e. decompose the problem into sub-problems and treat each part separately and then combine these parts in the final stage [37, 43, 44]. In this work, the initial solutions for the UKMMC instances are generated by decomposing the problem into sub-problems [29][34]. These sub-problems are generated and solved by the following steps:

I. Generate all valid 2-day and 3-day shift patterns for all shift types (morning (M), evening (E), night (N) and day off (o)). See Tables 2 and 3 for all valid 2-day and 3-day shift patterns, respectively.

Table 2 All of 2-day valid shift patterns

No.	Pattern	No.	Pattern	No.	Pattern	No.	Pattern
1.	oo	4.	oN	7.	ME	10.	No
2.	oM	5.	Mo	8.	Eo	11	NN
3.	oE	6.	MM	9.	EE		

Table 3 All of 3-day valid shift patterns

No.	Pattern	No.	Pattern	No.	Pattern	No	Pattern	No.	Pattern
1.	ooo	7.	MMM	13.	Eoo	19.	EoN	25.	EoE
2.	ooM	8.	oEE	14.	EoM	20.	Noo	26.	NNN
3.	oMM	9.	MoE	15.	EEE	21.	NNo		
4.	Moo	10.	MME	16.	ooN	22.	oME		
5.	MoM	11.	MEo	17.	oNN	23.	EEo		
6.	MMo	12.	MEE	18.	MoN	24.	ooE		

II. Combine the generated 2-day and 3-day shift patterns to form one-week valid patterns. This combination is based on the restriction list of one-week time or fewer numbers of days that affect one nurse only (see Figure 1).

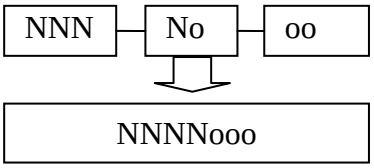

Figure 1. Combine three shift patterns to form a one-week shift pattern

Table 4 shows some examples of one-week valid shift patterns after we combined two 2-day and 3-day shift patterns. Table 5 shows the possible one-week patterns. In the literature, shift patterns have been used to effectively construct rosters in highly constrained nurse scheduling problems [32, 38, 43].

Table 4 Example of one-week valid shift patterns

No.	Pattern	No.	Pattern	No.	Pattern	No.	Pattern
1.	MMMMooo	3.	NNNNooo	5.	ooEEEEo	7.	oNNNNoo
2.	EEEEooo	4.	oMMMMo	6.	ooMEEMo		

Table 5 The possible one-week valid patterns

	First week							Second week						
P1	N	N	N	N	o	o	A	A	A	A	A	A	A	A
P2	o	N	N	N	N	o	o	A	A	A	A	A	A	A
P3	A	o	N	N	N	N	o	A	A	A	A	A	A	A
P4	A	A	o	N	N	N	N	o	o	A	A	A	A	A
P5	A	A	A	o	N	N	N	N	o	o	A	A	A	A
P6	A	A	A	A	o	N	N	N	N	o	o	A	A	A
P7	A	A	A	A	A	o	N	N	N	N	o	o	A	A
P8	A	A	A	A	A	A	o	N	N	N	N	o	o	A
P9	A	A	A	A	A	A	A	o	N	N	N	N	o	o
P10	A	A	A	A	A	A	A	A	o	N	N	N	N	o
P11	A	A	A	A	A	A	A	A	A	o	N	N	N	N
P12	A	A	A	A	A	A	A	A	A	A	o	N	N	N
P13	A	A	A	A	A	A	A	A	A	A	A	o	N	N
P14	A	A	A	A	A	A	A	A	A	A	A	A	o	N
P15	N	o	o	A	A	A	A	A	A	A	A	A	A	A
P16	N	N	o	O	A	A	A	A	A	A	A	A	A	A
P17	N	N	N	O	o	A	A	A	A	A	A	A	A	A
P18	A	A	A	A	A	A	A	A	A	A	A	A	A	A

A: any (Morning, Evening or day off), N: night, o: day off

In this work, a list of valid shift patterns that satisfy the imposed hard constraints are created. Each shift pattern represents a schedule of a working week. For the required two-week working period in UKMMC, in a solution vector x^i, two one-week valid shift patterns are allocated for each nurse. Based on the objective function values the candidate solutions in the HM are sorted in an ascending order. Figure 2 presents the pseudo code of building the HM.

Building the Harmony Memory (HM) for the UKMMC problem

```
begin
    for i = (1 to HMS) do
        x^i = ø;
        for j = (1 to n) do /* n is the number of nurses */
            choose two shift patterns randomly from the shift patterns pool for nurse j
        endfor
        calculate the objective function value f(x^i)
        add x^i to HM
    endfor
    sort the solutions based on its objective function value in HM in an ascending order
end
```

Figure 2 Building the harmony memory

Figures 3 and 4 present illustration examples of the construction of solution vectors and HM, respectively. Assume that there are 503 shift patterns. *M*, *E*, *N* and *O* represent morning shift, evening shift, night shift and day off, respectively. In Figure 3, two shift patterns 2 and 415 are randomly selected from the shift patterns pool and the schedule (2,415) is used for the first and second week time slots, respectively, of Nurse 1 in the new solution. This process is repeated for all nurses, as shown in Figure 4. HSA then calculates the penalty value for each solution vector and add them to the HM.

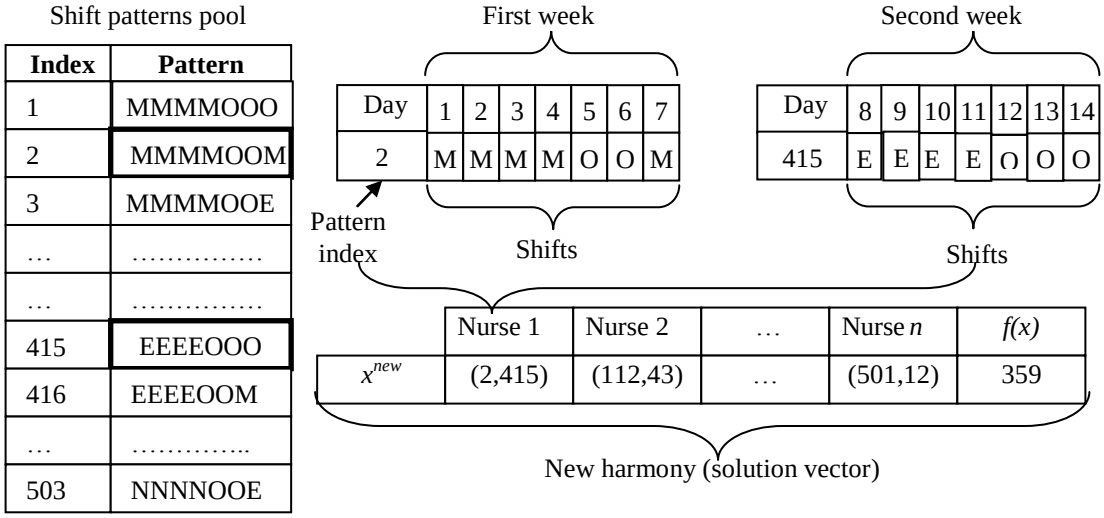

Figure 3 Example of constructing the solution vectors

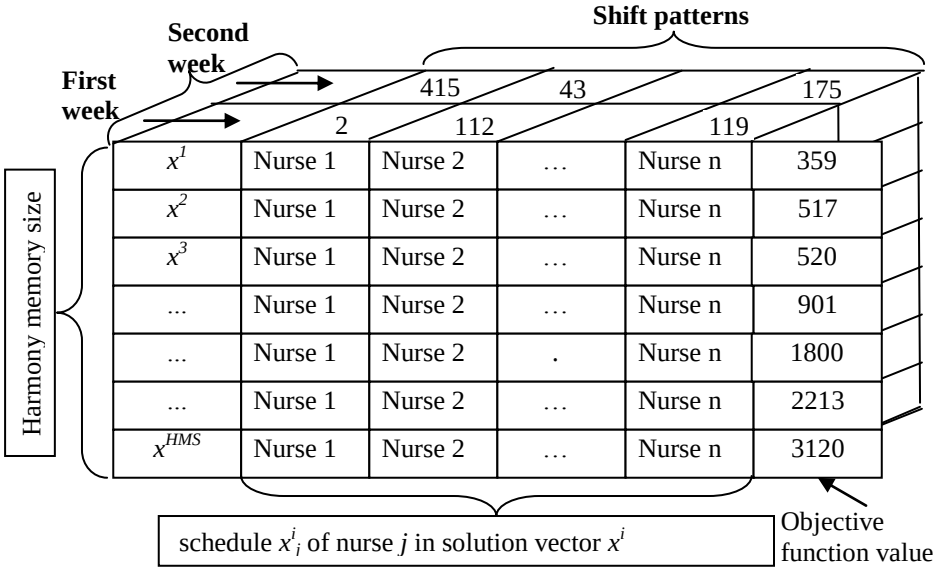

Figure 4 The HM representation

As for the benchmark instances, the set of initial solutions in HM is created by using a neighborhood operator which incrementally adds new shifts to the roster until all nurses have been scheduled [31] [32].

3.3 Improvise New Solutions

The diversification and intensification of the search in HSA are maintained in this step. Parameters HMCR and PAR are the main factors in intensifying or diversifying the search for globally and locally improved solutions [45]. In this step, new solutions (complete rosters) are constructed stochastically using one of the following three operators: (i) memory consideration (based on the HMCR) (ii) random consideration (based on 1-HMCR) and (iii) pitch adjustment (based on the PAR).

HMCR is the probability which decides whether to choose the shift pattern p_i of the solution vector x^i randomly from the HM, or from the shift patterns pool. The search with higher HMCR is guided more by HM, where the history and experience of the search process are stored. It thus helps to speed up the convergence by reducing the level of diversification and randomness in the search. Whereas, lower value of HMCR increases the diversification which, in turn, may cause a slow convergence due to the search jumps around potential candidate solutions.

If the new shift pattern of the solution x_i^{new} is selected from the HM, then the solution may be adjusted based on the PAR. The diversification is mainly controlled by the two parameters PAR and bandwidth. Higher values of these parameters lead the search to explore different areas in the solution space. Therefore, this will slow down the convergence of the algorithm. In contrast, lower values of the PAR and the bandwidth help in decreasing the degree of diversification. However, this may limit the exploration and lead to premature convergence where HSA is easily trapped into the local optima [23] [36].

The improvisation step in HSA is similar to reproduction in GAs that uses crossover and mutation operators [46]. In HSA, however, the improvisation utilizes the full HM to construct the solution, while in GA new chromosomes are generated by crossover of two parents or

mutation. Operators in GA need to be carefully designed especially in highly constrained problems to avoid destroying building blocks of optimal solutions [34]. Figure 5 presents the pseudo code of the improvisation process. Figure 6 shows an illustrative example of improvising the new solution.

Step 3 Improvise a new harmony (generate new solution)

```
begin
  for i = (1 to NI) do /* NI is the maximum number of improvisations*/
    x_i^new = ø
    for j = (1 to n) do /* n is the number of nurses */
      if (rand(0,1) ≤ HMCR) then
        choose shift pattern p_j randomly from the HM:
        if (rand(0,1) ≤ PAR) then
          adjust the chosen shift pattern p_j according to bandwidth
          /* bandwidth = (rand(5,-5)) is used to change the index of p_j*/
        else
          add the chosen p_j to solution x_i^new without changing
        endif
      else choose p_j randomly from the shift patterns pool
        add the chosen p_j to solution x_i^new
      endif

    endfor
    update HM
  endfor
end
```

Figure 5 The pseudo code of improvising new solutions

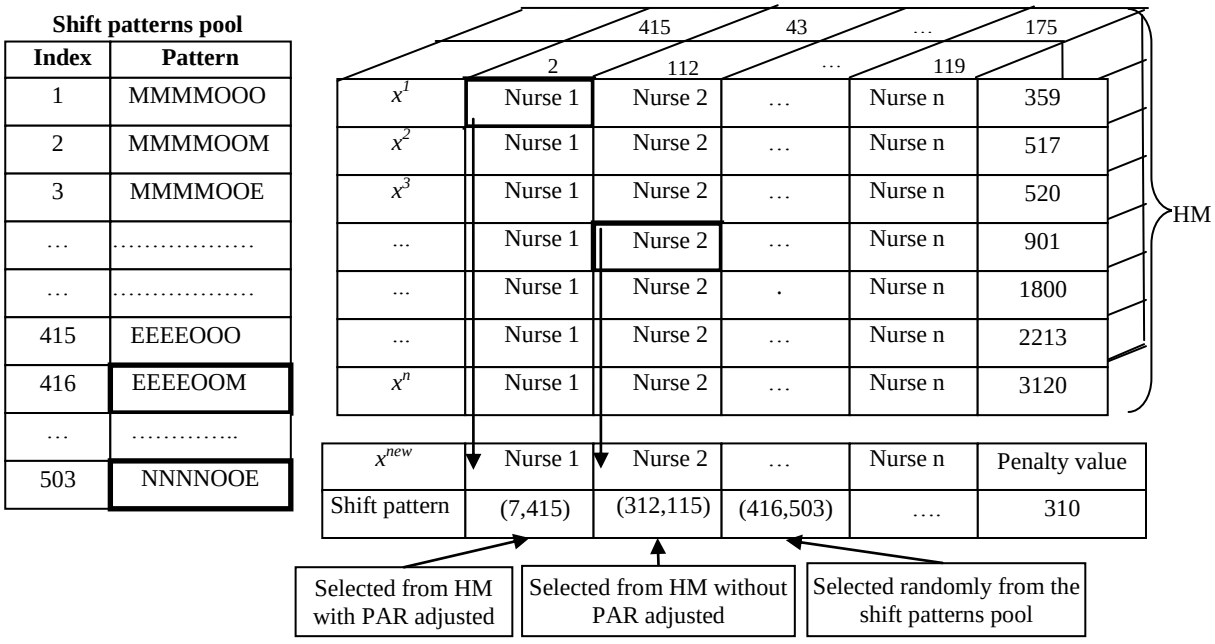

Figure 6 Example of improvisation process in HSA

In the example in Figure 6, two shift patterns 2 and 415 are firstly allocated to Nurse 1 in x^{new} from the HM based on HMCR (see Figure 4). The bandwidth {5,-5} is applied which changes the index of the shift pattern in the first week from shift pattern 2 to 7. The resulting

shift pattern (7,415) means the first week is assigned shift pattern 7 and the second week assigned shift pattern 415. For Nurse 2, two shift patterns (312,115) are selected from the HM without adjusting. Next Nurse 3 is randomly assigned two shift patterns from the shift patterns pool based on (1 – HMCR). This process is repeated until a complete solution is constructed.

3.4 Update the Harmony Memory

The solutions in HM are sorted in an ascending order based on their objective function values. If the new solution x_i^{new} is better than the worst one in HM, it will replace the worst solution in HM. Otherwise, x_i^{new} is discarded.

3.5 Check the Stopping Criteria

The evolution of solution improvisation and HM update (i.e. steps 3 and 4) is carried out until reaching the stopping condition. Normally the stopping criterion is a certain number of improvisations [19]. In this work we have employed three stopping criteria. These are: (1) reaching the maximum number of improvisations, (2) no improvement happened after a certain number of improvisations, and (3) a solution with objective function value 0 (no soft constraint violations) is obtained.

4. COMPUTATIONAL EXPERIMENTS AND RESULTS

In this work, we have tested HSA over a real world problem and a set of benchmark problems in NRPs. The UKMMC dataset is collected from a large hospital in Malaysia. The benchmark dataset [32] is widely used by the scientific literature [31][32]. Because there is no method tested on the UKMMC dataset, we conducted an empirical study of the impact of HSA parameters when solving the eight real world instances in UKMMC. We have also implemented a standard genetic algorithm to compare it with the HSA on the UKMMC dataset. The experiments were carried out on a Windows Vista 32-bit laptop with Intel 1.73 GHz and 2-GB RAM.

4.1 HSA for solving the UKMMC dataset

Finding proper balance between diversification and intensification is one of the important issues in devising meta-heuristic algorithms [29]. A series of experiments were conducted to study the behavior of HSA with different parameter values for HMS, HMCR, PAR and NI to strike a balance between diversification and intensification of the HSA search. We have used the eight instances in the UKMMC dataset shown in Table 6 to test the proposed HSA and GA.

Table 6 The instances of the UKMMC dataset

Instance	Total nurses	Seniors	Weekday demand (Mon-Fri)			Weekend demand (Sat-Sun)		
			Morning	Evening	Night	Morning	Evening	Night
CICU	11	8	3	3	2	2	2	2
SGY5	18	11	4	4	3	4	4	2
MD1	19	12	4	4	3	4	4	2
NICU	49	29	10	10	10	8	8	7
N50	50	31	10	10	10	10	10	10
ED	57	32	13	13	10	11	11	8

GICU	73	38	16	16	15	15	15	14
ICU	79	41	17	17	16	16	16	15

4.1.1 Harmony Memory Size (HMS)

We first examine the impact of HMS by a set of experiments. Four instances from the UKMMC dataset with different problem size have been chosen for these experiments. The two small instances (CICU and MD1) include 11 and 19 nurses, respectively. The medium instance (NICU) has 49 nurses and the large instance (ICU) includes 79 nurses. In this experiment test, we use the following parameter values: HMCR = 0.89, PAR = 0.2, bandwidth = (-5 or 5) and the number of improvisations NI=100. Table 7 presents the experimental results (average out of 30 runs) of HSA using different HMS parameter values. Figure 7 presents the plots of these results.

Table 7 Results of HSA with different parameter values of HMS

HMS	**CICU**	**MD1**	**NICU**	**ICU**
HMS=1	53754.4	35130.4	90302.6	150920.8
HMS= 10	40664.4	30180.2	75933.4	143153.8
HMS= 20	43511.6	26400.6	74238.6	141439.4
HMS= 30	41468.0	26808.2	72808.6	131151.6
HMS= 40	39934.4	21964.6	70976.6	130464.0
HMS= 50	40660.8	21689.4	71118.8	129177.6
HMS= 60	39799.6	24302.0	70810.4	123352.4
HMS= 70	35663.0	22576.0	69700.4	121558.4
HMS= 80	36996.6	**20660.8**	**61936.4**	**119825.8**
HMS= 90	**35048.2**	20923.6	62543.2	120815.0
HMS= 100	35874.4	21140.6	62371.4	120828.4

Based on the experimental results in Table 7 and Figure 7, HMS = 80 shows to be the most suitable parameter value compared to other HMS values. Results also indicate that the bigger the HMS, the better is the chance to start the improving process of the candidate solutions with lower objective function values. This might be due to a larger number of solutions in HM provide more good shift patterns, which are more likely to be combined into good new solutions. However, larger values of HMS do not contribute to better results. This may be due to that during the evolution, information of high quality shift patterns have been stored and updated in HM. More patterns may contain redundant information, thus do not necessarily contribute to a better performance. Therefore, HMS = 80 is chosen for all tested instances. Note that when HMS = 1, HSA behaves as a local search based method, where HMCR does not play a role and PAR assists a local search with the bandwidth.

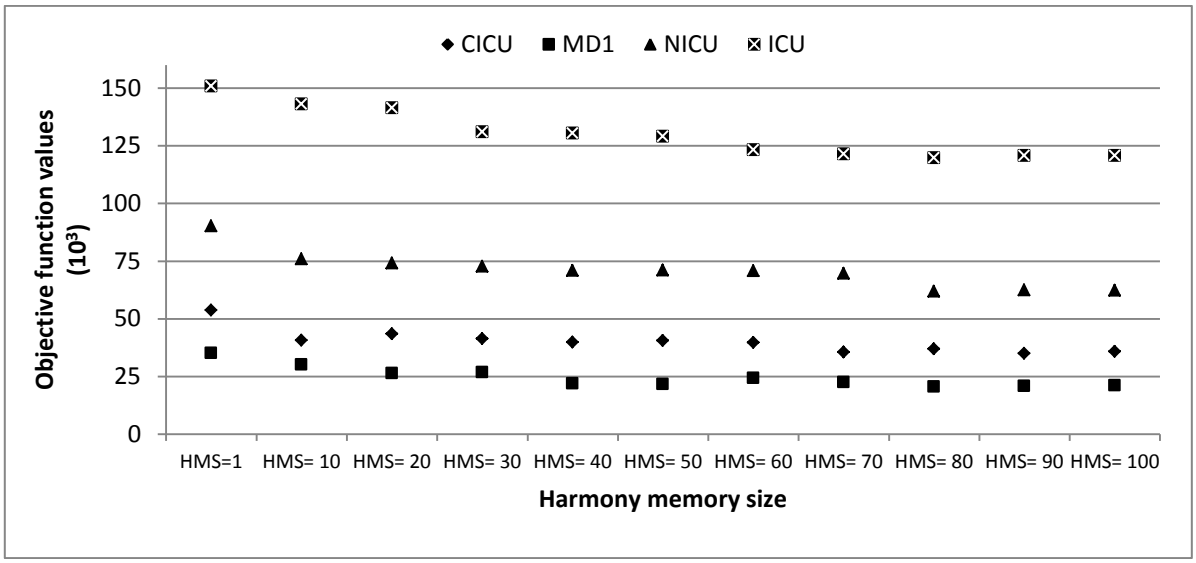

Figure 7 Comparison of different HMS parameter values on the UKMMC dataset

4.1.2 HMCR and PAR

In the literature, the recommended values range from 0.79 to 0.99 for HMCR and 0.1 to 0.3 for PAR [41, 47]. In this work, we examine several HMCR and PAR values as shown in Table 8 and Figure 8. We fix the number of improvisations NI to100 iterations.

Table 8 Results of using HSA with different parameter values of HMCR and PAR

HMCR	PAR	CICU	MD1	NICU	ICU
0.79	0.1	31715.47	19635.33	59420.83	97991.47
	0.2	31879.13	19512.5	57558.9	95136.07
	0.3	29120.0	19368.03	57621.13	95848.47
0.89	0.1	25695.53	18510.57	53556.6	95623.93
	0.2	23358.6	17768.4	53346.33	91488.17
	0.3	23349.9	17713.6	51605.73	88517.67
0.95	0.1	21672.27	16095.27	49554.97	86614.07
	0.2	**19769.87**	**15319.63**	**44755.6**	82163.2
	0.3	22867.77	15423.1	48016.33	**81151.2**
0.99	0.1	22099.13	16174.6	48134.47	85991.8
	0.2	21572.6	16286.67	49973.73	81463.4
	0.3	23467.27	15843.6	48522.07	84536.27

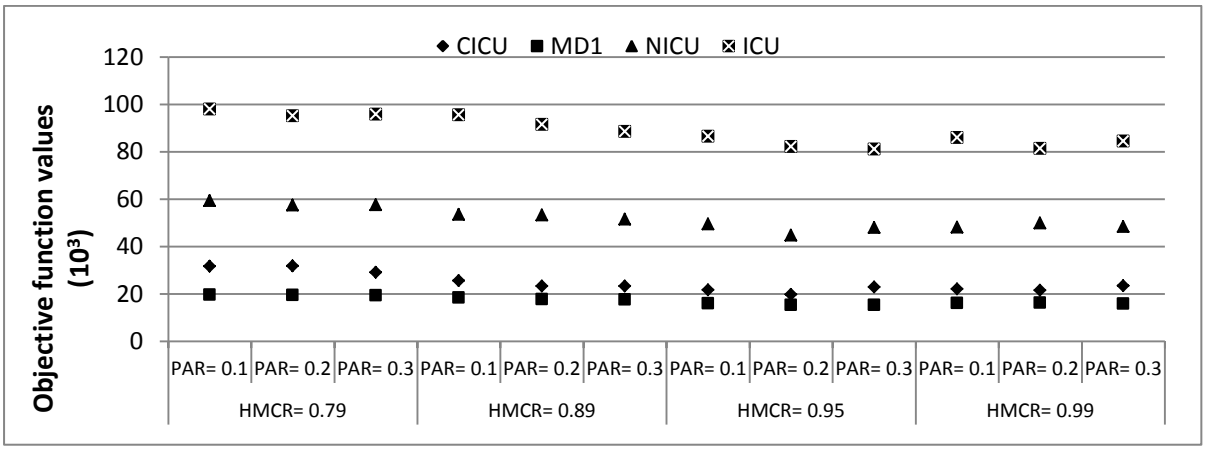

Figure 8 Results of using HSA with different parameter values of HMCR and PAR on UKMMC dataset

From Table 8, we found that best results were obtained when HMCR = 0.95 and PAR = 0.2. These are also the recommended parameter values in the literature [33][35][37]. For the bandwidth, we set the range (5, -5) as it changes the indexes of the shift patterns only. Based on the results, we set the HMCR as 0.95 and PAR as 0.2 for all tested instances (with regard to this dataset).

4.1.3 Number of Improvisations NI

In order to determine the suitable parameter value for the number of improvisations NI (i.e. number of iterations), we have studied the status of the candidate solutions in the HM during the run (see Table 9). These experiments help us to observe the behavior of the HM and to decide the suitable maximum number of NI. Figure 9 plots the results presented in Table 9.

It is noticed that a significant decrease takes place to the objective function values during the first 25000 iterations. Within 25000 and 50000 iterations, the amount of change becomes very small. After 45000 iterations there is no improvement at all in all instances. All that leads us to choose the number of iterations to be NI = 50000.

Table 9 Objective function values of candidate solutions in HM using HSA during the run

No. of Iterations	CICU	MD1	NICU	ICU
Initial values	36130.0	23710.1	53517.3	111620.4
After 1000	23419.5	16178	39841.2	81986.1
After 5000	16721.7	12734.1	28741.6	62167.9
After 10000	9819.1	6718.8	28741.6	47452.1
After 15000	5180.2	2180.1	19651.4	31954.6
After 20000	1211.0	1091.0	11894	23278.1
After 25000	617.1	1091.0	7861.1	11674.9
After 30000	411.3	719.6	4367.1	7681.5
After 35000	401.8	518.8	2311.0	3211.2
After 40000	397.0	410.0	1218.9	2017.0
After 45000	397.0	410.0	927.0	1711.0
After 50000	397.0	410.0	927.0	1711.0
After 55000	397.0	410.0	927.0	1711.0
After 60000	397.0	410.0	927.0	1711.0

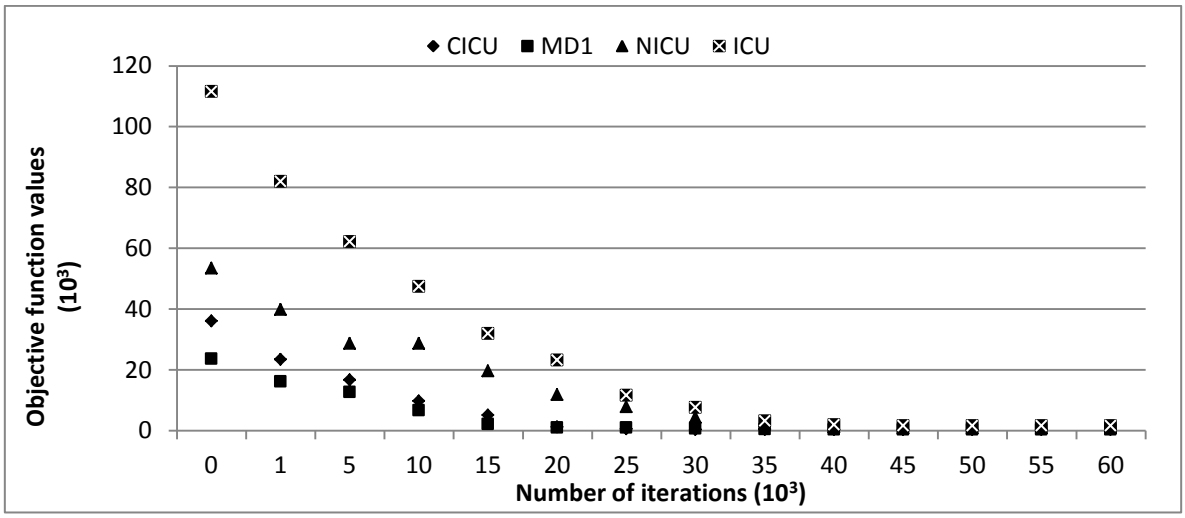

Figure 9 Status of candidate solutions in HM during the run using HSA

4.1.4 Results of HSA on the UKMMC Dataset

We present the performance of HSA on the UKMMC dataset. In HSA, the search stops if one of the following stopping criteria is met: (1) If the maximum number of improvisations NI reaches 50000, (2) If the value of objective function reaches 0 or (3) If no improvement occurs during 10000 improvisations. Based on the preliminary experiments in the previous sections, we use the following parameter values: HMCR = 0.95, PAR = 0.2 and bandwidth $\in$ {-5, 5}. For each instance, we ran the HSA for 20 times.

4.1.5 Comparison between GA and HSA

GA is a population based meta-heuristic algorithm [48] that mimics the process of natural evolution. Due to its many similar characteristics to HSA, GAs quite often has been compared with HSA [33]. It works by managing a population of individuals which evolves by using three genetic operators: (i) selection, (ii) crossover, and (iii) mutation.

In this work, we have implemented the stochastic ranking based GA for NRPs proposed in [49] using the same parameter settings as follows: the number of individuals = 1000, crossover rate = 0.75 and mutation rate = 0.02. For the selection, we used tournament selection with stochastic ranking (tournament size = 7) and elitism. For the crossover operator, we used a single point crossover as in [49] and [50]. The mutation operator is carried out by randomly changing one shift pattern for one nurse selected randomly. For each instance, we ran the basic GA for 20 times. To assure a fair comparison between GA and HSA, the stopping criteria for GA is similar to HSA as follows: 1) If the maximum number of function evaluations reach 50000 (80 individual * 625 generations, which is equal to NI in HSA, 2) If the value of objective function reaches 0, or 3) If no improvement occurs within 10000 generations (improvisations in HSA). Table 10 shows the best, the average, the worst, the median, the standard deviation of the objective function values in addition to the number of generated desirable patterns and the execution time.

Table 10 Comparison results: HSA and GA on UKMMC dataset

nst ce	HSA	GA

	Best PV	Average	Worst PV	Median	Stddv.	DPs	Time	Best PV	Average	Worst PV	Median	Stddv.	DPs	Time
CICU	310	341.05	481	316.5	47.865	5	185.1	2791	3786.6	5510	3589	904.7	2	81.2
SGY5	221	301.3	410	285	66.778	8	115.3	3680	5015.9	6930	4904.5	1053.4	3	96
MD1	339	406.4	534	385.5	66.239	9	127	4219	5661.9	7450	5803.5	1054.3	5	99.6
NICU	786	981.2	1255	955.5	151.897	18	168.8	786	981.2	1255	955.5	151.89	18	168.8
N50	821	1075.45	1398	1044.5	187.083	21	175	1021	1258.06	1345	1198.4	250.01	18	847
ED	998	1194.25	1517	1134	181.194	24	265.9	1430	1287.43	1670	1208.5	196.07	20	310.4
GICU	1481	1606.95	1812	1582.5	120.036	27	295.2	1850	1675.5	1983	1582.5	160.42	24	274
ICU	1518	1770.45	2097	1777	175.923	29	345.7	1621	1854.13	2186	1846	212.31	21	310.2

Stddv: standard deviation. DPs: desirable patterns. Time: time in seconds

The comparison between the HSA and the basic GA in Table 10 shows that HSA outperformed GA on all tested instances of UKMMC. Due to the fact that NRPs are highly constrained problems, basic GA might struggle in getting good quality solution without using specialized operators (i.e. specialized repair mechanism and crossover operators). This is noticed by many researchers [51, 52]. This fact is experimentally supported in Table 10 where the basic GA fails to obtain good quality results for most of the tested instances. This is partially due to the drawbacks of the basic GA which relies on two parents only to produce the new offspring [34].

To investigate the performance differences between HSA and GA, a Wilcoxon test is carried out with 95% confidence level. A *p*-value less than 0.05 means there is a significant difference between these methods. The *p*-value of HSA versus GA is reported in Table 11, where HSA is statistically significant to GA on 6 out of 8 instances (i.e. *p*-value less than 0.05). The results also support the fact that the HSA outperformed GA on the majority of problem instances.

Table 11 The *P*-value of HSA versus GA

HSA vs.	GA
Instances	*P*-value
CICU	0.732
SGY5	0.541
MD1	0.000
NICU	0.026
N50	0.041
ED	0.024
GICU	0.000
ICU	0.000

4.2 HSA for the benchmark NRPs dataset

In this section, the HSA is tested on the widely used benchmark instances in NRPs. These instances, together with their problem description and formulation are taken from (www.cs.nott.ac.uk/~tec/NRP/) [32]. Table 12 shows the characteristics of the tested instances.

Table 12 The characteristics of the tested instances

Instance	No. of nurses	No. of shift types	Planning period (days)	Best known solution

ORTEC01	16	4	31	270
ORTEC02	16	4	31	270
BCV-1.8.1	8	4	28	252
BCV-2.46.1	46	4	28	1572
BCV-3.46.1	46	3	26	3280
BCV-3.46.2	46	3	26	894
BCV-4.13.1	13	4	29	10
BCV-5.4.1	4	4	28	48
BCV-6.13.1	13	4	30	768
BCV-7.10.1	10	6	28	381
BCV-8.13.1	13	4	28	148
BCV-A.12.1	12	4	31	1294
BCV-A.12.2	12	4	31	1953
CHILD-A2	41	5	42	1111
ERRVH-A	41	4	48	795

The HSA parameter values are also fixed the same as follows: HMCR = 0.95, PAR = 0.2, bandwidth ∈ {-5, 5} and NI = 50,000. Table 13 shows the obtained results of the HSA. Please note that we used the same parameter setting for both datasets. We believe that this is an important element of this paper.

Table 13, on the other hand, shows a comparison between the HSA with other existing meta-heuristic algorithms from the literature on these benchmark instances. In the literature, some mathematical approaches and other exact methods have also been investigated for these instances [53, 44]. We compare our proposed HSA with only meta-heuristics from the literature.

Table 13 HSA for the benchmark datasets

Instance	**Our Method HSA**				
	Best	**Average**	**Worst**	**Stddv.**	**Time**
ORTEC01	310	334	360	19.493	412
ORTEC02	330	356	372	16.792	446
BCV-1.8.1	270	272.6	279	3.649	187
BCV-2.46.1	1612	1630.2	1644	12.194	281
BCV-3.46.1	3380	3391.6	3400	8.763	463
BCV-3.46.2	905	909.8	913	3.114	181
BCV-4.13.1	11	12	14	1.224	212
BCV-5.4.1	48	48	48	0	31
BCV-6.13.1	796	869	960	65.144	149
BCV-7.10.1	386	411.4	463	33.178	234
BCV-8.13.1	158	164.4	188	13.221	267
BCV-A.12.1	2210	2491.8	2614	173.950	243
BCV-A.12.2	1998	2223.6	2550	259.528	298
CHILD-A2	1222	1278	1340	53.028	518
ERRVH-A	2820	3011.6	3120	122.526	409

Table 14 Comparison of HSA results and other meta-heuristic methods

Instance	**HSA**		**M1**		**M2**		**M3**		**M4**		**M5**	
	Best OV	**Time (S)**	**Best OV**	**Time (S)**	**Best OV**	**Time (S)**	**Best OV**	**Time (S)**	**Best OV**	**Time (S)**	**Best OV**	**Time (H)**
ORTEC01	**310**	412	1580	3351	601	713	405	1381	-	-	541	12
ORTEC02	**330**	446	-	-	-	-	-	-	-	-	-	-

BCV-1.8.1	270	187	275	99	263	66	**253**	815	352	6	-	-
BCV-2.46.1	1612	281	1574	2560	**1573**	1665	1575	1076	1594	191	-	-
BCV-3.46.1	3380	463	3439	10714	3379	5226	**3344**	3814	3724	137	-	-
BCV-3.46.2	**905**	181	-	-	-	-	**-**	-	-	-	-	-
BCV-4.13.1	11	212	12	93	11	114	**10**	374	18	10	-	-
BCV-5.4.1	**48**	31	**48**	27	**48**	9	**48**	126	200	1	-	-
BCV-6.13.1	796	149	815	385	806	207	**768**	592	986	15	-	-
BCV-7.10.1	386	234	**381**	66	**381**	76	**381**	361	472	10	-	-
BCV-8.13.1	158	267	**148**	219	**148**	123	**148**	226	**148**	11	-	-
BCV-A.12.1	2210	243	1990	929	1685	518	**1434**	1786	3335	44	-	-
BCV-A.12.2	**1998**	298	-	-	-	-	-	-	-	-	-	-
CHILD-A2	**1222**	518	-	-	-	-	-	-	-	-	-	-
ERRVH-A	**2820**	409	-	-	-	-	-	-	-	-	-	-

OV: objective function value. S: second. H: hour.

Where:
HSA Our proposed Harmony Search Algorithm
M1 A Memetic Approach in [54].
M2 A Scatter Search Approach (SS1) in [18].
M3 A Scatter Search Approach (SS2) in [18].
M4 A Shift Sequence Based Approach in [37].
M5 A Hybrid Heuristic Ordering and Variable Neighbourhood Search in [38].

Results in Table 14 demonstrate that HSA is able to obtain competitive results for some instances. For the ORTEC01 and BCV-A.12.2 instances, we have obtained new lower bound results compared with other methods in the literature. For the BCV-5.4.1 instance, HSA also obtains the same best known result by other approaches. In fact, the results of most instances (except only two instances BCV-2.46.1 and BCV-A.12.1) are only slightly worse than the best known results. For the five instances ORTEC02, BCV-3.46.2, BCV-A.12.2, CHILD-A2 and ERRVH-A, we report the first results, i.e. no results have been reported in the literature.

Table 15 shows the average results of HSA compared to existing meta-heuristic algorithms. As can be seen, HSA matched the best average results on one instance and achieved better average results for one out of 10 instances.

Table 15 Comparison of HSA average results and other meta-heuristic methods

Instance	HSA	M1	M2	M3	M4
	Average	Average	Average	Average	Average
ORTEC01	**334**	2904	1707	445	-
ORTEC02	**356**	-	-	-	-
BCV-1.8.1	272.6	285	268	**253**	365
BCV-2.46.1	1630.2	1589	**1588**	1594	1629
BCV-3.46.1	3391.6	3471	3396	**3380**	3789
BCV-3.46.2	**909.8**	-	-	-	-
BCV-4.13.1	12	19	12	**10**	84
BCV-5.4.1	**48**	**48**	135	**48**	200
BCV-6.13.1	869	959	904	**768**	1209
BCV-7.10.1	411.4	390	385	**381**	507
BCV-8.13.1	164.4	166	**148**	**148**	151
BCV-A.12.1	2491.8	2349	1813	**1522**	3972

BCV-A.12.2	**2223.6**	-	-	-	-
CHILD-A2	**1278**	-	-	-	-
ERRVH-A	**3011.6**	-	-	-	-

To find out whether the performance of HSA is statistically different from existing meta-heuristic algorithms (M1, M2, M3 and M4), we have performed a multiple comparison statistical test as follows: the Friedman and Iman-Davemport tests with a critical level of 0.05 are conducted to detect whether there are statistical differences between the results of these methods [55]. The *p*-value of Friedman (*p*-value =0.000) and Iman-Davemport (*p*-value =0.000) are less than the critical level 0.05, indicating that a significant difference between the compared methods. Therefore, a post-hoc statistical test is used to detect the correct difference between the methods. Table 16 shows the average ranking (the smaller the better) produced by the Friedman test for each method. As can be seen, M3 ranked the first, followed by M2, HSA, M1 and M4. Table 17 summarizes the *p*-value of the Holm and Hochberg statistical tests [55] where M3 is the controlling algorithm. As can be seen, M3 is statistically better than M1, M4 and HSA (i.e., *p*-value less than 0.05) but does not outperform M2. The results also demonstrate that HSA results are different from M1 and M4.

Table 16 The average ranks of Friedman test

Algorithm	Ranking
HSA	3.166
M1	3.444
M2	2.333
M3	1.388
M4	4.666

Table 17 The p-value of Holm and Hochberg tests for the compared methods

Algorithm	unadjusted *P*	*P*Holm	*P*Hochberg
M4	0.000011	0.000044	0.000044
M1	0.005819	0.017457	0.017457
HSA	0.017073	0.034145	0.034145
M2	0.205118	0.205118	0.205118

Overall, our results demonstrate that HSA produced good results compared to some existing methods. This may be due to the characteristic of HSA in striking a well-balanced diversification and intensification for highly complex problems.

5. CONCLUSIONS

This paper has investigated the harmony search algorithm for solving the nurse rostering problem. The proposed harmony search algorithm evolves upon the harmony memory which stores adaptively updated solutions during the evolution. The proposed algorithm has been evaluated on eight instances collected form a real world hospital UKMMC and 15 problem instances from a widely used nurse rostering problem benchmark in the literature. For the

UKMMC instances, we have conducted a series of experiments to test the efficiency of harmony search algorithm with different parameter settings. The results demonstrate that higher values of harmony memory size contribute to a better starting point of the improving process to obtain good feasible solutions. In addition, HSA shows to outperform the basic GA. As for the benchmark instances, results showed that HSA obtained competitive results when compared to other state-of-the-art meta-heuristic methods in the literature (indeed obtained better results for few instances).

REFERENCES

[1] P. Brucker, R. Qu, E. Burke, Personnel scheduling: Models and complexity, European Journal of Operational Research, 210 (2011) 467-473.
[2] Y.A. Ozcan, Quantitative Methods in Health Care Management: Techniques and Applications, Jossey-Bass/Wiley, San Francisco, CA, 2005.
[3] E.K. Burke, P. De Causmaecker, G.V. Berghe, H. Van Landeghem, The State of the Art of Nurse Rostering, Journal of Scheduling, 7 (2004) 441-499.
[4] P. De Causmaecker, G. Vanden Berghe, A categorisation of nurse rostering problems, Journal of Scheduling, 14 (2011) 3-16.
[5] E. Özcan, Memes, Self-generation and Nurse Rostering, in: Practice and Theory of Automated Timetabling VI, Springer Berlin / Heidelberg, 2007, pp. 85-104.
[6] E.K. Burke, G. Kendall, Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, in, Springer, 2005, pp. 620.
[7] E.G. Talbi, Metaheuristics: From Design to Implementation, Wiley Online Library, 2009.
[8] B. Cheang, H. Li, A. Lim, B. Rodrigues, Nurse Rostering Problems - A Bibliographic Survey, European Journal of Operational Research, 151 (2003) 447-460.
[9] A.T. Ernst, H. Jiang, M. Krishnamoorthy, D. Sier, Staff Scheduling and Rostering: A Review of Applications, Methods and Models, European Journal of Operational Research, 153 (2004) 3-27.
[10] J.R. Thornton, A. Sattar, Applied Partial Constraint Satisfaction Using Weighted Iterative Repair, in: Australian Joint Conference on Artificial Intelligence Springer, Verlag, 1997, pp. 57 -66.
[11] Y.A. Ozcan, Quantitative Methods in Health Care Management: Techniques and Applications, in, Jossey-Bass/Wiley, San Francisco, CA, 2005.
[12] U. Aickelin, K.A. Dowsland, An indirect Genetic Algorithm for a Nurse-Scheduling Problem, Computers & Operations Research, 31 (2004) 761-778.
[13] M. Moz, M.V. Pato, A Genetic Algorithm Approach to a Nurse Rerostering Problem, Journal of Computers and Operations Research, 34 (2007) 667-691.
[14] E.K. Burke, P. Causmaecker, G. Vanden Berghe, A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem, in: Lecture Notes in Artificial Intelligence, Springer, 1998, pp. 187–194.
[15] K.A. Dowsland, Nurse scheduling with tabu search and strategic oscillation, European Journal of Operational Research, 106 (1998) 393-407.
[16] M. Hadwan, M. Ayob, A Constructive Shift Patterns Approach with Simulated Annealing for Nurse Rostering Problems, in: International Symposium in Information Technology (ITSim 2010) IEEE, Kuala Lumpur, Malaysia, 2010, pp. 1-6.
[17] C. Mingang, H.I. Ozaku, N. Kuwahara, K. Kogure, O. Jun, Simulated Annealing Algorithm for Daily Nursing Care Scheduling Problem, in: International Conference on Automation Science and Engineering, (CASE 2007), IEEE, 2007, pp. 507-512.
[18] E.K. Burke, T. Curtois, R. Qu, G.V. Berghe, A Scatter Search Methodology for the Nurse Rostering Problem, Journal of the Operational Research Society, 61 (2010) 1667-1679.

[19] Z.W. Geem, J.H. Kim, G.V. Loganathan, Original Harmony Search, A New Heuristic Optimization Algorithm: Harmony Search, Journal of Simulation, 76 (2001) 60-68.
[20] M.A. Al-Betar, A.T. Khader, A Harmony Search Algorithm for University Course Timetabling, Annals of Operations Research, (2010): 1-29.
[21] Z.W. Geem, K.S. Lee, Y. Park, Application of Harmony Search to Vehicle Routing, American Journal of Applied Sciences, 2 (2005) 1552-1557.
[22] Z.W. Geem, Harmony Search Algorithm for Solving Sudoku, Lecture Notes in Artificial Intelligence, (2007).
[23] M. El-Abd, Performance assessment of foraging algorithms vs. evolutionary algorithms, Information Sciences, 182 (2012) 243-263.
[24] R. Forsati, M. Mahdavi, M. Shamsfard, M. Reza Meybodi, Efficient stochastic algorithms for document clustering, Information Sciences, 220 (2013) 269-291.
[25] L. Liu, H. Zhou, Hybridization of Harmony Search with Variable Neighborhood Search for Restrictive Single-machine Earliness / Tardiness Problem, Information Sciences.
[26] P. Yadav, R. Kumar, S.K. Panda, C.S. Chang, An Intelligent Tuned Harmony Search algorithm for optimisation, Information Sciences, 196 (2012) 47-72.
[27] A.R. Yildiz, A comparative study of population-based optimization algorithms for turning operations, Information Sciences, 210 (2012) 81-88.
[28] K.S. Lee, Z.W. Geem, A New Meta-heuristic Algorithm for Continuous Engineering Optimization: Harmony Search Theory and Practice, Computer Methods in Applied Mechanics and Engineering, 194 (2005) 3902–3933.
[29] X.S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, 2008.
[30] M.A. Awadallah, A.T. Khader, M.A. Al-Betar, A.L. Bolaji, Nurse Scheduling Using Harmony Search, in: Sixth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2011, pp. 58-63.
[31] A.E. Eiben, M. Zbigniew, M. Schoenauer, J.E. Smith, Parameter Control in Evolutionary Algorithms, in: F.G. Lobo, C.F. Lima, Z. Michalewicz (Eds.) Parameter Setting in Evolutionary Algorithms Springer, Charlotte, 2007, pp. 19-46.
[32] T. Curtois, Personnel scheduling data sets and benchmarks, in, University of Nottingham, 2009.
[33] G. Ingram, T. Zhang, Overview of Applications and Developments in the Harmony Search Algorithm, in: Z.W. Geem (Ed.) Music-Inspired Harmony Search Algorithm, Springer Berlin / Heidelberg, 2009, pp. 15-37.
[34] X.S. Yang, Harmony Search as a Metaheuristic Algorithm, in: Z.W. Geem (Ed.) Music-Inspired Harmony Search Algorithm, Springer Berlin / Heidelberg, 2009, pp. 1-14.
[35] S. Petrovic, G. Vanden Berghe, A comparison of two approaches to nurse rostering problems, Annals of Operations Research, (2010) 1-20.
[36] M. Hadwan, M.B. Ayob, An Exploration Study of Nurse Rostering Practice at Hospital Universiti Kebangsaan Malaysia, in: 2nd Conference on Data Mining and Optimization, 2009. DMO '09. , IEEE, Bangi, Selangor Malaysia, 2009, pp. 100 - 107.
[37] P. Brucker, E.K. Burke, T. Curtois, R. Qu, G. Vanden Berghe, A shift Sequence Based Approach for Nurse Scheduling and a New Benchmark Dataset, Journal of Heuristics, 16 (2010) 559-573.
[38] E.K. Burke, T. Curtois, G. Post, R. Qu, B. Veltman, A Hybrid Heuristic Ordering and Variable Neighbourhood Search for the Nurse Rostering Problem, European Journal of Operational Research, 188 (2008) 330-341.
[39] T. Curtois, Gabriela Ochoa, Matthew Hyde, J.A. Vázquez-Rodríguez, A HyFlex Module for the Personnel Scheduling Problem, Technical Report, School of Computer Science, University of Nottingham, (2010) 1-12.

[40] E.K. Burke, Tim Curtois, Matthew Hyde, Gabriela Ochoa, Jose A. Vazquez-Rodriguez, HyFlex: A Benchmark Framework for Cross-domain Heuristic Search, arXiv.org, (2011) 28.
[41] K.S. Lee, Z.W. Geem, A New Structural Optimization Method Based on the Harmony Search Algorithm, Journal of Computers and Structures, 82 (2004) 781-798.
[42] B. Maenhout, M. Vanhoucke, An Electromagnetic Meta-heuristic for the Nurse Scheduling Problem, Journal of Heuristics, 13 (2007) 359-385.
[43] P. Brucker, R. Qu, E.K. Burke, G. Post, A Decomposition, Construction and Post-processing Approach for A Specific Nurse Rostering Problem, in: G. Kendall, Lei, L., Pinedo, M. (Ed.) 2nd Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA'05), New York, USA, 2005, pp. 397–406.
[44] A. Ikegami, A. Niwa, A Subproblem-centric Model and Approach to the Nurse Scheduling Problem, Journal of Mathematical Programming 97 (2003) 517–541.
[45] M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search algorithm for solving optimization problems, Applied Mathematics and Computation, 188 (2007) 1567-1579.
[46] R. Poli, W.B. Langdon, Foundations of Genetic Programming, Springer-Verlag, Berlin, Germany, 2002.
[47] P.V. Ravikumar, B.K. Panigrahi, Dynamic Economic Load Dispatch using Hybrid Swarm Intelligence Based Harmony Search Algorithm, in: Expert Systems with Applications, 2011, pp. 8509-8514.
[48] I.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press., Michigan, 1975.
[49] R. Bai, E.K. Burke, G. Kendall, J. Li, B. McCollum, A hybrid evolutionary approach to the nurse rostering problem, IEEE Transactions on Evolutionary Computation, 14 (2010) 580-590.
[50] H. Kawanaka, K. Yamamoto, T. Yoshikawa, T. Shinogi, S. Tsuruoka, Genetic algorithm with the constraints for nurse scheduling problem, in: The 2001 Congress on Evolutionary Computation, 2001, pp. 1123-1130.
[51] U. Aickelin, K. Dowsland, Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem, Journal of Scheduling, 3 (2000) 139-153.
[52] M. Ohki, A. Morimoto, K. Miyake, Nurse Scheduling by Using Cooperative GA with Efficient Mutation and Mountain-Climbing Operators, in: 3rd International Conference on Intelligent Systems, 2006 IEEE, 2006, pp. 164-169.
[53] M. Azaiez, S. Al Sharif, A 0-1 Goal Programming Model for Nurse Scheduling, Computer Operational Research, 32 (2005) 491-507.
[54] E. Burke, P. Cowling, P. De Causmaecker, G. Vanden Berghe, A Memetic Approach to the Nurse Rostering Problem, Applied Intelligence, 15 (2001) 199–214.
[55] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Information Sciences, 180 (2010) 2044-2064.