2011 Sixth International Conference on Bio-Inspired Computing: Theories and Applications

# Nurse Scheduling Using Harmony Search

Mohammed A. Awadallah[1], Ahamad Tajudin Khader[1], Mohammed Azmi Al-Betar[1,2], Asaju La'aro Bolaji[1]
[1] School of Computer Sciences, Universiti Sains Malaysia (USM), 11800, Pulau Pinang, Malaysia.
[2] Department of Computer Science, Al-zaytoonah University, Amman-Jordan.
Emails: {mama10_com018@student, tajudin@cs, mohbetar@cs, abl10_sa0739@student}.usm.my

***Abstract*—In this research an adaption of Harmony Search Algorithm (HSA) for Nurse Scheduling Problem (NSP) is presented. Nurse scheduling problem is a task of assigning shifts to nurses for the duties that have to carry out. The difficulty of handling this problem is due to the high number of constraints to be satisfied. Thus, we are proposing an adaptation of HSA i.e. a new population-based metaheuristic algorithm that mimics the musical improvisation process which has been successfully applied for wide range of optimisation problems. The performance of HSA is evaluated using datasets established by International Nurse Rostering Competition 2010 *(INRC2010)*. The results obtained were compared with the best results reported in the competition. The results show that the proposed method can compete well in comparison with those reported results.**

***Index Terms*—Harmony Search, Metaheuristic Algorithms, Population-based, Nurse Scheduling.**

## I. Introduction

Transportation sectors, hospitals, companies, and academic institutions are but a few examples of the bodies that have to run business within a framework of scheduling to ensure smooth transactions. Nurse scheduling problem is an example considered in this work. Nurse scheduling problem can be defined as a task of assigning shifts to personnel members who should belong to the proper qualification categories required for the duties that have to carry out [1]. In general, the overall aim of scheduling is to organise people's lives, works and activities that are needed to be performed timely and without any forms of disruptions.

The assignment of nurses to shifts periodically takes into account a number of requirements depend on the regions which includes country and administrative laws, contract information and nurses' preferences, etc. In particular, the process of scheduling the nurses to shifts seems overtiring and baffling. The main objective to hospital management is to generate high quality nurse rosters quickly which satisfies all the requirements as much as possible. If the management requirements are not satisfied in the schedule, then the schedule will be rejected, and also if nurses preferences are not satisfied, this can lead to increase in dissatisfaction rate and absenteeism among nurses.

The interests of the researchers have been drawn to the problem for the past five decades and several methods have been applied to solve different versions of the problem. These includes: (i) Mathematical Programming methods such as integer programming [2][3], and goal programming [4]. (ii) Artificial Intelligent methods which includes case-based reasoning [5][6], and Constraint Programming [7][8]. (iii) And Approximation methods such as Ant colony optimization [9], Genetic Algorithm [10][11][12], Tabu search [13][14], variable neighbourhood search [15][16], and memetic algorithm [17].

Several nurse scheduling problem instances have been made available for purpose of evaluation, these include a real-world datasets from Belgian hospitals [16], Queens Medical Centre University Hospital Trust (QMC) [6][5], ORTEC [15] [18], Vienna hospital [9], Riyadh Al-Kharj hospital [4]. Recently, *INRC2010* proposed an artificial datasets which is used to attracts the attention of researchers in various fields in order to close the gap between research and practice is the main focus of this research. The problem instances are divided into three categories due to their sizes and complexities: akin to sprint, middle, and long distance running. Each category is further classified into four types: early, hidden, late and hint. More details can be sought in [http://www.kuleuvenkortrijk.be/nrpcompetition].

The harmony search algorithm is a metaheuristic algorithm developed by Geem et al.,[19]. It is a stochastic search mechanism, simple in concepts, and no derivation information is required in the initial search [20]. It has been successfully applied to wide range of optimisation and scheduling problems such as course timetabling [21][22], examination timetabling [23], traveling salesman problem [19], and water network design [24].

HSA mimics the musical improvisation process in which a group of musicians play the pitches of their musical instruments together seeking a pleasing harmony as determined by audio-aesthetic standard. It is considered a population-based algorithm with local search based concepts [20][21]. HSA start with a population of solutions. It improves the population iteratively using three operators: memory consideration which make use of accumulative search, random consideration which is the source of randomness, and pitch adjustment which is the source of local improvement. At each iteration, only one new solution is generated and substituted with the worst solution in the population. This process is repeated until it converges.

The main objective of this research is to adapt Harmony Search Algorithm to Nurse Scheduling Problem as a spark work which explores the efficiency of this algorithm for the problem. Results show that the harmony search can be used to tackle the problem intelligently and offers a good quality solution when compared with competitors results.

The organisation of this paper includes the following sections: Section II discusses the nurse scheduling problem while

978-0-7695-4514-1/11 $26.00 © 2011 IEEE
DOI 10.1109/BIC-TA.2011.75

the explanations on the fundamentals of harmony search algorithm were discussed in Section III. The application of harmony search algorithm to nurse scheduling problem is presented in Section IV. Section V discusses the experimental results and compares with the best results reported in the website. The final section presents the conclusion and some possible future directions.

## II. NURSE SCHEDULING PROBLEM

The problem of nurse scheduling is exemplified by the existence of a set of constraints, which is typically divided into two types: hard and soft constraints. Hard constraints are those that should be met or satisfied at all times, while violations of the soft constraints are allowed but should be avoided, if possible. Shifts have to be assigned to a set of nurses such that at any time the imposed demanded constraints on personnel regarding the required skills are met and no nurses can take more than one shift per day. Each nurse is described by a skill category (or a set of skill categories) and a work regulation or contract. Each nurse works according to exactly one contract, which determines the percentage of her/his employment. Usually, problems with a large number of possible shift types, a large number of nurses with different skills are more difficult to solve. The objective is to find a nurse scheduling solution that satisfy all hard constraints while minimising the violation of soft constraints.

The following shows the hard and soft constraints as provided by INRC2010:

**Hard Constraints**

- $H_1$**:** All demanded shifts must be assigned to a nurse.
- $H_2$**:** A nurse can only work one shift per day, i.e. no two shifts can be assigned to the same nurse on a day.

**Soft Constraints**

- $S_1$ **:** Maximum and minimum number of assignments for each nurse during the scheduling period.
- $S_2$ **:** Maximum and minimum number of consecutive working days.
- $S_3$ **:** Maximum and minimum number of consecutive free days.
- $S_4$ **:** Assign complete weekends.
- $S_5$ **:** Assign identical complete weekends.
- $S_6$ **:** Two free days after a night shift.
- $S_7$ **:** Requested day on/off.
- $S_8$ **:** Requested shift on/off.
- $S_9$ **:** Alternative skill.
- $S_{10}$**:** Unwanted patterns (where pattern as a set of legal shifts defined in terms of work to be done during the shifts[25]).

The nurse scheduling problem version tackled in this work can be formulated as follows:

- A set of N $=\{n_1,n_2,...,n_I\}$ of I nurses, each nurse have one or more of different skill categories and can take different types of shifts.
- A set of C $=\{c_1,c_2,...,c_R\}$ of R contracts, each nurse have exactly one contract.
- A set of D $=\{d_1,d_2,...,d_M\}$ of M maximum days of period time schedule, each day is divide to set of shifts types.
- A set of K $=\{k_1,k_2,...,k_J\}$ of J skill categories.
- A set of S $=\{s_1,s_2,...,s_T\}$ of T shifts types.

Furthermore, the set of problem instances produced by *INRC2010* provide the following information:

- A Cover_request matrix CR, contains number of nurses that must be available on each shift *t* on each day *j* during time period schedule $CR_{jt} \geq 0$, where $j \in D$, and $t \in S$.
- A Day_off matrix DF, that a nurse *i* prefers not to work on day *j* if and only if $DF_{ij}$=1, where $i \in N$, and $j \in D$.
- A Day_on matrix DO, that a nurse *i* prefers to work on day *j* if and only if $DO_{ij}$=1, where $i \in N$, and $j \in D$.
- A Shift_off matrix SF, that a nurse *i* prefers not to work on specific shift *t* on day *j* if and only if $SF_{ijt}$=1, where $i \in N$, $j \in D$, and $t \in S$.
- A Shift_on matrix SO, that a nurse *i* prefers to work on specific shift *t* on day *j* if and only if $SO_{ijt}$=1, where $i \in N$, $j \in D$, and $t \in S$.

The main objective of solving this problem is to find an optimal solution in a feasible region. Where the solution is evaluated by an objective function as in Eq.(1), that calculates the total number of soft constraints violations while the two hard constraints are satisfied.

$$\min G(x) = \sum_{s=1}^{10} C_s \cdot g_s(x) \tag{1}$$

## III. OVERVIEW OF HARMONY SEARCH ALGORITHM

Algorithm 1 shows the pseudo-code of the HSA with five main steps that will be described in the following:

***Step 1*. Initialize the problem and HSA parameters**. Suppose that the discrete optimization problem is modeled as in Eq.(2).

$$\min_{\boldsymbol{x} \in \mathbf{X}} f(\boldsymbol{x}) \text{s.t.} (g(\boldsymbol{x}) < 0) \wedge (h(\boldsymbol{x}) = 0) \tag{2}$$

Where $f(\boldsymbol{x})$ is the objective function; $\boldsymbol{x} = \{x_i | i = 1, \dots, N\}$ is the set of each decision variable. $\mathbf{X} = \{\boldsymbol{X}_i | i = 1, \dots, N\}$ contains all the possible *discrete* values of each decision variable, i.e., $\boldsymbol{X}_i = \{v_{i,1}, v_{i,2}, \dots, v_{i,K_i}\}$. $N$ is the number of decision variables, and $K_i$ is the number of possible values of decision variable $x_i$. $g(\boldsymbol{x})$ are inequality constraint functions and $h(\boldsymbol{x})$ are equality constraint functions.

The parameters of the HSA required to solve the optimization problem are also specified in this step: (i) the Harmony Memory Consideration Rate (HMCR), used in the improvisation process to determine whether the value of a decision variable is to be selected from the solutions stored in the Harmony Memory (HM), or randomly selected from the available range of possible values.

(ii) The Harmony Memory Size (HMS) is similar to the population size in genetic algorithm. (iii) The Pitch Adjustment Rate (PAR), decides whether the decision variables are to be adjusted to a neighbouring value. (iv) The Number of

**Algorithm 1** The basic harmony search algorithm

**STEP1 Initialize the problem and HSA parameters**

Input data instance of the optimisation problem
Set the HSA parameters (HMCR, PAR, NI, HMS).

**STEP2 Initialize the harmony memory**

Construct vectors of the harmony memory, $\textbf{HM} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{\text{HMS}}\}$
Recognize the worst vector in **HM**, $\boldsymbol{x}^{\text{worst}} \in \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{\text{HMS}}\}$

**STEP3 Improvise a new harmony**

```
x' = φ // new harmony vector
for i = 1, ··· , N do
    if (U(0,1) ≤ HMCR) then
        x'_i ∈ {x_i^1, x_i^2, ..., x_i^HMS} {memory consideration}
        if (U(0,1) ≤ PAR) then
            x'_i = v_{i,k±m} { pitch adjustment }
        end if
    else
        x'_i ∈ X_i { random consideration }
    end if
end for
```

**STEP4 Update the harmony memory**

```
if (f(x') < f(x^worst)) then
    Include x' to the HM.
    Exclude x^worst from HM.
end if
```

**STEP5 Check the stop criterion**

```
while (not termination criterion is specified by NI) do
    Repeat STEP3 and STEP4
end while
```

Improvisations (NI) corresponds to the number of iterations. Note that the HMCR and PAR are parameters used in the improvisation process. These parameters will be explained in more detail in the next steps.

***Step 2*. Initialize the harmony memory**. The harmony memory (HM) is an augmented matrix which contains sets of solution vectors determined by HMS (see Eq.3). In this step, these vectors are randomly constructed and stored in the HM according to the values of the objective function.

$$HM = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{\text{HMS}} & x_2^{\text{HMS}} & \cdots & x_N^{\text{HMS}} \end{bmatrix} \tag{3}$$

***Step 3*. Improvise a new harmony**. In this step, the HSA will construct (or *improvise*) a new harmony vector from scratch, $\boldsymbol{x}' = (x_1', x_2', \cdots, x_N')$, based on three operators: (1) memory consideration, (2) random consideration, and (3) pitch adjustment.

*Memory consideration*. In this operator, the value of the first decision variable $x_1'$ is randomly assigned from the historical values, $\{x_1^1, x_1^2, \ldots, x_1^{HMS}\}$, stored in HM vectors. Values of the other decision variables, $(x_2', x_3', \ldots, x_N')$, are sequentially assigned in the same manner with probability (w.p.) of HMCR where $0 \leq \text{HMCR} \leq 1$. The work of this operator is similar to the recombination operator in other population-based methods and is a good source of exploitation [26].

*Random consideration*. Decision variables that are not assigned with values according to memory consideration are randomly assigned according to their possible range by random consideration with a probability of (1-HMCR) as in Eq.(4).

$$x_i' \leftarrow \begin{cases} x_i' \in \{x_i^1, x_i^2, \ldots, x_i^{\text{HMS}}\} & \text{w.p.} \quad \text{HMCR} \\ x_i' \in \boldsymbol{X}_i & \text{w.p.} \quad \text{(1 - HMCR)} \end{cases} \tag{4}$$

Random consideration is functionally similar to the mutation operator in Genetic Algorithm which is a source of global exploration in HSA [26]. The HMCR parameter is the probability of assigning one value of a decision variable, $x_i'$, based on historical values stored in the HM. For instance, if HMCR =0.90, this means that the probability of assigning the value of each decision variable from historical values stored in the HM vectors with the probability of 0.9, and the value of each decision variable is assigned from its possible value range with the probability of 0.1.

*Pitch adjustment*. Every decision variable $x_i'$ of a new harmony vector, $\boldsymbol{x}' = (x_1', x_2', x_3', \ldots, x_N')$, that has been assigned a value by memory considerations is examined for whether or not it should be pitch adjusted with the probability of PAR ($0 \leq \text{PAR} \leq 1$):

$$\text{Pitch adjust for } x_i'? \leftarrow \begin{cases} \text{Yes} & \text{w.p.} \quad \text{PAR} \\ \text{No} & \text{w.p.} \quad \text{(1-PAR)} \end{cases} \tag{5}$$

A PAR of 0.10 means that the HSA modifies the existing value of decision variables assigned by memory consideration with a probability of (PAR $\times$ HMCR), while the other values of decision variables assigned by memory consideration do not change. If the pitch adjustment decision for $x_i'$ is Yes, the value of $x_i'$ is modified to its neighboring value as follows:

$$x_i'(k) = v_{i,k\pm m} \tag{6}$$

Where $x_i'$ is assigned with value $v_{i,k}$, that is, the $k$th element in $\boldsymbol{X}_i$. $m$ is the neighbouring index, $m \in \mathbb{Z}$. The following summarizes the improvisation process of step 3, which is the main mechanism for iterating towards an optimal solution:

$$x_i' \leftarrow \begin{cases} x_i' \in \{x_i^1, x_i^2, \ldots, x_i^{\text{HMS}}\} & \text{w.p.} \quad \text{HMCR} \\ x_i' = v_{i,k\pm m} & \text{w.p.} \quad \text{HMCR} \times \text{PAR} \\ x_i' \in \boldsymbol{X}_i & \text{w.p.} \quad \text{1 - HMCR} \end{cases} \tag{7}$$

***Step 4*. Update the harmony memory**. If the new harmony vector, $\boldsymbol{x}' = (x_1', x_2', \cdots, x_N')$, is better than the worst harmony vector in HM, the new harmony vector replaces the worst harmony vector.

***Step 5*. Check the stop criterion**. Step 3 and step 4 of HSA are repeated until the stop criterion (maximum number of improvisation) is met. This is specified by the NI parameter.

## IV. THE METHOD

The nurse scheduling solution is represented by a vector of nurses $x = (x_1, x_2, ..., x_N)$. Each nurse, $x_i$ , is to be scheduled into a feasible allocation within the range between $[0, (S \times D) - 1]$, where S is the number of shifts and D represents scheduling period time (see the solution matrix in Eq.8).

$$Solution = \begin{bmatrix} x_{0,0} & x_{0,1} & \cdots & x_{0,(S*D-1)} \\ x_{1,0} & x_{1,1} & \cdots & x_{1,(S*D-1)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N-1,0} & x_{N-1,1} & \cdots & x_{N-1,(S*D-1)} \end{bmatrix} \quad (8)$$

For example, in the small problem instances $(sprint\_early)$ established by INRC2010, the number of nurses N = 10, the number of shifts S = 4 and the scheduling period time D = 28 days, the possible allocations of each nurse, $x_i$ , is within the range between 0 to 111. The HSA deduces the shifts of each nurse, $x_i$, as in Eq.(9):

$$x_i = S \times D + s \quad (9)$$

This means that nurse $x_i$ is scheduled in shift $s_T$ at day $d_M$, M is the day index (range 0 to D−1) and T is the shift index (range 0 to S−1). For example, let $x_i$ = (110, 28, 89, . . . , 0) be a feasible and complete shifts for nurse $x_i$. The HSA deduces the solution as follows: nurse $x_i$ is scheduled in allocation 110, the shift index 2 at day index 27; nurse $x_i$ is scheduled at allocation 28, the shift index 0 at day index 7.

***Step 1*. Initialize the problem and HSA parameters**. Within the NSP, the parameters are extracted from the problem instances as discussed in Section II, such as set of nurses *N*, set of contracts *C*, set of skill categories *K*, set of shift kinds *S*, period of schedule *D*, Cover_request matrix *CR*, Day_off matrix *DF*, Day_on matrix *DO*, Shift_off matrix *SF*, and Shift_on matrix *SO*. The HSA parameters such as HMS, HMCR, PAR and NI that are required to solve NSP are also defined.

***Step 2*. Initialize the harmony memory**. Here, the set of feasible solutions for NSP are generated and stored in HM. During the search process, infeasible schedule are not considered and regenerated again, this ensure that all solutions in HM are feasible. First of all, in this step, the pool of feasible allocations are constructed it contains all the shifts on different days with different skills for each nurse. In the second part a heuristic ordering is used to generate the feasible solutions as shown in [15]. The heuristic ordering is the process that sort all the shifts in order of the estimated difficulty of assignment that depend on the number of nurses request for each shift per day and number of skills of nurses to work in this shift. For example, shift type DH (Day Head) is assigned to nurses as "head nurse" before other shift types (see table I). Nurse with skill DH must be assigned to the DH shift type before other shift types. If assigned to other shift types before DH shift type, this can violate the hard constraint which may lead to incomplete solution.

| Shift | Nurses request | Nurses available | Order |
|---|---|---|---|
| D | 3 | 10 | 2 |
| N | 1 | 10 | 2 |
| E | 2 | 10 | 2 |
| L | 3 | 10 | 2 |
| DH | 1 | 1 | 1 |

TABLE I
SHIFT ORDERING

***Step 3*. Improvise a new harmony**. In this step, a new harmony solution, $x' = (x'_1, x'_2, ..., x'_N)$, is generated from scratch based on three operators: (i) memory consideration, (ii) random consideration, and (iii) pitch adjustment. The new harmony solution must be complete and feasible. In some iterations, the HSA operators may not improvise (generate) a complete and feasible solution. If this occurs, the solution is discarded and a new one is generated.

*Memory consideration*. The memory consideration selects feasible allocation (mapping between shift and day) for the nurses to be scheduled in the new harmony solution $x' = (x'_1, x'_2, ..., x'_N)$, sequentially from the solutions stored in HM with the probability of HMCR.

*Random consideration*. The remaining nurses that have not been scheduled by memory consideration will select any feasible allocation available to be scheduled in the new harmony solution with probability (1-HMCR).

*Pitch adjustment*. The work of PAR is similar to that of neighborhood structures in local search-based methods as follows: the pitch adjustment operator PAR is divided into three procedures: (i) The pitch adjustment Move, (ii) the pitch adjustment SwapNurses, and (iii) the pitch adjustment SwapDays. Each shift $x'$ scheduled out of memory consideration is pitch adjusted with probability of PAR where $0 \leq PAR \leq 1$. The PAR in this study is divided into three parameters PAR1, PAR2 and PAR3, where PAR1=(PAR/3), PAR2=(2*PAR/3), PAR3=PAR, each of which controls the pitch adjustment procedure as in (see Eq.10):

$$x'_i \leftarrow \begin{cases} Move & 0 \leq U(0,1) < \text{PAR1} \\ SwapNurses & \text{PAR1} \leq U(0,1) < \text{PAR2} \\ SwapDays & \text{PAR2} \leq U(0,1) < \text{PAR3} \\ Do\ nothing & \text{PAR3} \leq U(0,1) \leq 1 \end{cases} \quad (10)$$

The three proposed pitch adjustment procedures are designed to work as follows:

- Pitch adjustment $Move$: A nurse allocation $x'_i$ that meets probability [0,PAR1) is randomly moved to any free feasible nurse allocation in the new harmony solution.
- Pitch adjustment $SwapNurses$: A nurse allocation $x'_i$ that meets the range of probability [Par1,Par2) is ran-

domly swapped with another nurse allocation (e.g., $x'_j$) that has already been scheduled in the new harmony while maintaining the feasibility.

- Pitch adjustment $SwapDays$: A nurse allocation $x'_i$ that meets the range of probability [Par2,Par3) is randomly swapped day with another nurse allocation (e.g., $x'_j$) that has already been scheduled in the new harmony while maintaining the feasibility.

In this work, any local change which obtained worse result during generation process is discarded.

## V. Experiments and Results

### A. The Problem Instances

The proposed method is evaluated using a set of 33 problem instances, publicly made available by INRC2010 and classified to four types: 10 *sprint_early*, 10 *sprint_late*, 10 *sprint_hidden*, and 3 *sprint_hint*. These instances differs in number of shifts, number of unwanted patterns, Day_OFF request, Shift_OFF request, number of contracts, and details of contracts like number of maximum shifts for nurse, maximum and minimum consecutive working days etc. Table II shows the characteristics of the instances used for the experiment.

| Characteristic | Early | Late | Hidden | Hint |
|---|---|---|---|---|
| Number of nurses | 10 | 10 | 10 | 10 |
| Number of Skills | 1 | 1 | 1 | 1 |
| Number of Shifts | 4 | 4 | 3,4 | 4 |
| Number of Contracts | 4 | 3 | 3 | 3 |
| Number of patterns | 3 | 0, 3, 4, 8 | 4,8 | 0, 8 |
| Period of schedule | 1 to 28 Jan. 2010 | 1 to 28 Jan. 2010 | 1 to 28 Jun. 2010 | 1 to 28 Jan. 2010 |
| Day Off Request | $\surd$ | $\surd, or X$ | $\surd$ | $\surd$ |
| Shift Off Request | $\surd$ | $\surd, or X$ | $\surd$ | $\surd$ |

TABLE II
CHARACTERISTICS OF PROBLEM INSTANCES

### B. Experimental Results and discussion

The proposed method is programmed in Microsoft Visual C++ 6.0 under Windows XP on an Intel Machine with CoreTM and a 2.66GHz processor and 2GB RAM. We ran the experiment 10 times for each problem instance for the purpose of statistical calculation. The parameters setting for HSA are: HMS =100, HMCR=0.99, PAR=0.01, and NI is between 100000 and 300000, where the search process stops after 5000 iterations without improvement. The first column in table III shows the best results as published by the INRC2010 website, the second column shows the best results of the proposed method, while third and fourth columns shows the worst and the average results, and the last column is for the standard deviation of the solution violations for proposed method. Notably, the results are very close to the competitors' results, though the techniques used to solve the problem by the competitors are not published yet. These results are the best cited results from 15 comparative methods.

| Instance Name | Competition Results | Achieved by HSA: Best | Worst | Average | Std. div |
|---|---|---|---|---|---|
| sprint_early01 | 56 | 61 | 69 | 65.6 | 2.73 |
| sprint_early02 | 58 | 63 | 69 | 66.6 | 1.85 |
| sprint_early03 | 51 | 56 | 70 | 63.5 | 4.46 |
| sprint_early04 | 59 | 66 | 75 | 70.4 | 2.76 |
| sprint_early05 | 58 | 63 | 70 | 66.3 | 2.53 |
| sprint_early06 | 54 | 60 | 66 | 62.6 | 2.11 |
| sprint_early07 | 56 | 64 | 73 | 66.4 | 2.91 |
| sprint_early08 | 56 | 61 | 68 | 64.4 | 2.06 |
| sprint_early09 | 55 | 61 | 75 | 66.0 | 3.69 |
| sprint_early10 | 52 | 60 | 72 | 62.8 | 3.46 |
| sprint_late01 | 37 | 56 | 73 | 64.4 | 5.20 |
| sprint_late02 | 42 | 61 | 73 | 66.6 | 3.41 |
| sprint_late03 | 48 | 68 | 78 | 72.8 | 3.37 |
| sprint_late04 | 75 | 136 | 183 | 157.4 | 16.97 |
| sprint_late05 | 44 | 61 | 74 | 67.9 | 3.75 |
| sprint_late06 | 42 | 50 | 64 | 56.1 | 3.78 |
| sprint_late07 | 42 | 70 | 135 | 87.0 | 17.80 |
| sprint_late08 | 17 | 20 | 87 | 46.9 | 20.89 |
| sprint_late09 | 17 | 23 | 72 | 40.8 | 16.53 |
| sprint_late10 | 43 | 77 | 171 | 102.3 | 27.81 |
| sprint_hidden01 | 33 | 57 | 75 | 67.3 | 5.20 |
| sprint_hidden02 | 32 | 55 | 64 | 60.5 | 2.91 |
| sprint_hidden03 | 62 | 90 | 105 | 96.5 | 4.50 |
| sprint_hidden04 | 67 | 94 | 219 | 118.8 | 40.94 |
| sprint_hidden05 | 59 | 81 | 96 | 87.8 | 4.66 |
| sprint_hidden06 | 134 | 238 | 547 | 313.9 | 81.90 |
| sprint_hidden07 | 153 | 288 | 384 | 333.4 | 33.37 |
| sprint_hidden08 | 209 | 317 | 510 | 375.7 | 54.39 |
| sprint_hidden09 | 338 | 606 | 1422 | 857.7 | 284.43 |
| sprint_hidden10 | 306 | 416 | 595 | 492.7 | 61.51 |
| sprint_hint01 | - | 110 | 158 | 141.5 | 13.76 |
| sprint_hint02 | - | 78 | 100 | 88.1 | 7.84 |
| sprint_hint03 | - | 111 | 163 | 137.3 | 14.84 |

TABLE III
EXPERIMENTAL RESULTS OF HARMONY SEARCH ALGORITHM

The result demonstrates that HSA could be used to solve NSP. Interestingly, the results shows that the proposed approach achieved results which are comparable with the best results as shown in table III within few seconds. The method need some improvement in order to enhance the performance of the algorithm and also to further enhance the results.

## VI. Conclusion and future directions

This paper presented the adaption of Harmony Search Algorithm for solving the NSP. As the results have shown, the algorithm is capable of solving nurse scheduling problem. Although the results produced by the algorithm in this study are presently not comparatively better than those already published, this is an initial adaptation of HSA.

It is highly recommendable that future work should be directed:

- To improve HSA for NSP by introducing more advanced neighborhood structures in pitch adjustment procedures.
- To integrate HSA with other metaheuristic algorithms.

## References

[1] E. Burke, P. De Causmaecker, and G. Vanden Berghe, "Novel meta-heuristic approaches to nurse rostering problems in Belgian hospitals," *Handbook of Scheduling: Algorithms, Models and Performance Analysis, pages*, pp. 44–1, 2004.

[2] A. Mason and M. Smith, "A nested column generator for solving rostering problems with integer programming," in *International conference on optimisation: techniques and applications*. Citeseer, 1998, pp. 827–834.

[3] B. Maenhout and M. Vanhoucke, "Branching strategies in a branch-and-price approach for a multiple objective nurse scheduling problem," *Journal of Scheduling*, vol. 13, no. 1, pp. 77–93, 2010.

[4] M. Azaiez and S. Al Sharif, "A 0-1 goal programming model for nurse scheduling," *Computers and Operations Research*, vol. 32, no. 3, pp. 491–508, 2005.

[5] G. Beddoe and S. Petrovic, "Enhancing case-based reasoning for personnel rostering with selected tabu search concepts," *Journal of the Operational Research Society*, vol. 58, no. 12, pp. 1586–1598, 2006.

[6] G. Beddoe, S. Petrovic, and J. Li, "A hybrid metaheuristic case-based reasoning system for nurse rostering," *Journal of Scheduling*, vol. 12, no. 2, pp. 99–119, 2009.

[7] R. Qu and F. He, "A hybrid constraint programming approach for nurse rostering problems," *Applications and Innovations in Intelligent Systems XVI*, pp. 211–224, 2009.

[8] H. Li, A. Lim, and B. Rodrigues, "A hybrid AI approach for nurse rostering problem," in *Proceedings of the 2003 ACM symposium on Applied computing*. ACM, 2003, pp. 730–735.

[9] W. Gutjahr and M. Rauner, "An ACO algorithm for a dynamic regional nurse-scheduling problem in Austria," *Computers & Operations Research*, vol. 34, no. 3, pp. 642–666, 2007.

[10] X. Cai and K. Li, "A genetic algorithm for scheduling staff of mixed skills under multi-criteria* 1," *European Journal of Operational Research*, vol. 125, no. 2, pp. 359–369, 2000.

[11] U. Aickelin and K. Dowsland, "An indirect genetic algorithm for a nurse-scheduling problem," *Computers & Operations Research*, vol. 31, no. 5, pp. 761–778, 2004.

[12] C. Tsai and S. Li, "A two-stage modeling with genetic algorithms for the nurse scheduling problem," *Expert Systems with Applications*, vol. 36, no. 5, pp. 9506–9512, 2009.

[13] K. Dowsland, "Nurse scheduling with tabu search and strategic oscillation," *European Journal of Operational Research*, vol. 106, no. 2-3, pp. 393–407, 1998.

[14] E. Burke, P. De Causmaecker, and G. Vanden Berghe, "A hybrid tabu search algorithm for the nurse rostering problem," *Simulated Evolution and Learning*, pp. 187–194, 1999.

[15] E. Burke, T. 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*, vol. 188, no. 2, pp. 330–341, 2008.

[16] B. Bilgin, P. De Causmaecker, B. Rossie, and G. Berghe, "Local search neighbourhoods to deal with a novel nurse rostering model," in *Proceedings of the 7th International Conference on Practice and Theory of Automated Timetabling, Montreal*, 2008.

[17] E. ozcan, "Memetic algorithms for nurse rostering," *Computer and Information Sciences-ISCIS 2005*, pp. 482–492, 2005.

[18] E. Burke, J. Li, and R. Qu, "A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems," *European Journal of Operational Research*, vol. 203, no. 2, pp. 484–493, 2010.

[19] Z. Geem, J. Kim, and G. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, p. 60, 2001.

[20] K. Lee, Z. Geem, S. Lee, and K. Bae, "The harmony search heuristic algorithm for discrete structural optimization," *Engineering Optimization*, vol. 37, no. 7, pp. 663–684, 2005.

[21] M. Al-Betar and A. Khader, "A harmony search algorithm for university course timetabling," *Annals of Operations Research*, pp. 1–29, 2008.

[22] ——, "A hybrid harmony search for university course timetabling," in *Proceedings of the 4nd Multidisciplinary Conference on Scheduling: Theory and Applications (MISTA 2009), Dublin, Ireland, August*, pp. 10–12.

[23] M. Al-Betar, A. Khader, and F. Nadi, "Selection mechanisms in memory consideration for examination timetabling with harmony search," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, 2010, pp. 1203–1210.

[24] Z. Geem, "Optimal cost design of water distribution networks using harmony search," *Engineering Optimization*, vol. 38, no. 3, pp. 259–277, 2006.

[25] A. Wren, "Scheduling, timetabling and rosteringa special relationship?" *Practice and Theory of Automated Timetabling*, pp. 46–75, 1996.

[26] X. Yang, "Harmony search as a metaheuristic algorithm," *Music-inspired harmony search algorithm*, pp. 1–14, 2009.