Hyper-heuristic approach for solving Nurse Rostering Problem

Khairul Anwar[1], Mohammed A. Awadallah[1,3], Ahamad Tajudin Khader[1], Mohammed Azmi Al-Betar[1,2]
[1] School of Computer Sciences, Universiti Sains Malaysia (USM), 11800
Pulau Pinang, Malaysia
[2] Department of Information Technology, Al-Huson University College,
Al-Balqa Applied University P.O. Box 50, Al-Huson, Irbid-Jordan
[3]Faculty of Computer Science, Al-Aqsa University, P.O. Box 4051, Gaza, Palestine
Email: ka10_com097@student.usm.my, mama10_com018@student.usm.my, tajudin@cs.usm.my, mohbetar@cs.usm.my

***Abstract*—. Hyper-heuristic (HH) is a higher level heuristic to choose from a set of heuristics applicable for the problem on hand. In this paper, a Harmony Search-based Hyper-heuristic (HSHH) approach is tested in solving nurse rostering problems (NRP). NRP is a complex scheduling problem of assigning given shifts to a given nurses. We test the proposed method by using the First International Nurse Rostering Competition 2010 (INRC2010) dataset. Experimentally, the HSHH approach achieved comparable results with the comparative methods in the literature.**

Keywords: Nurse Rostering Problems; Harmony Search ALgorithm; Hyper-heuristic methods

I. INTRODUCTION

Employee scheduling or personnel scheduling is a process of the allocation of timeslots and possibly locations and other resources to people. This problem has been extensively addressed in the literature more than four decades. Personnel scheduling problems arise in a variety of services like companies, institutions, hospitals, etc. One of the challenging problems in personnel scheduling is the scheduling of hospital personnel. In this paper, we concentrate on Nurse Rostering Problem (NRP). NRP is tackled by assigning a set of shifts to a set of nurses with different skills and work contracts over a scheduling period. Furthermore, the NRP is subject to *hard* and *soft* constraints. Hard constraints must be satisfied, while the soft constraints should be respected as much as possible. Over the years, many methods have been presented by the researchers to solve the problem. These methods include Tabu Search [1], Variable Neighborhood Search [2, 3], Simulated Annealing [4], Ant Colony Optimization [5], Branch-and-price Algorithm [6], Harmony Search [7, 8] and Scatter Search [9]. More details about some of these methods can be seen in [10], and [11].

Recently, there has been a growing trend toward more general method that can be used to solve multi optimization problems. Hyper-heuristic (HH) represents one of the approaches. Generally, hyper-heuristics are known as a heuristic (or meta-heuristic) to select one or more low-level heuristics for the particular problem in the hope of solving the problem on hand. The main defining property of hyper-heuristics is that, this method (hyper-heuristics) operates on the search space of heuristics (or heuristic component) rather than directly on the search space of solution [12]. A hyper-heuristic could be regarded as an 'off-the-peg' method as opposed to a 'made-to-measure' bespoke meta-heuristic [13]. In traditional framework of hyper-heuristic, it consists of two main mechanisms: *a heuristic selection mechanism* and *move acceptance mechanism*. Heuristic selection mechanisms choose the best low-level heuristic for generating a new (partial or complete) solution in the current optimization step and move acceptance mechanism to decide on the acceptability of the new solution.

Hyper-heuristic has been used in many different optimization problem domains such as personnel scheduling problem, university timetabling problems (i.e., course and examination), and others reported in [14]. Burke et al. [1] used tabu search as high-level heuristic selection mechanism in a hyper-heuristic framework in order to rank the low-level heuristic. In their approach, the ranking of the low-level heuristics is based on a reinforcement learning mechanism that considers the change in the candidate solution. The proposed method has been successfully applied to course timetabling and nurse rostering problems with a comparative result, although it is not as good as the published results.

Bilgin et al. [15] present one general hyper-heuristic approach for addressing two scheduling problems in health care domain: the patient admission scheduling problem and the nurse rostering problem. In this study, they test 36 hyper-heuristic variations, combination of three types of selection mechanism such as simple random, choice function and dynamic heuristic set strategy and four different move acceptance criteria such as only improve, improve and equal, simulated annealing and great deluge with three different tournament size (i.e., 4, 16, 64). From the experiments they conclude that the great deluge move acceptance criterion with tournament size 4 results in the best average performance. However, the heuristic selection mechanism is different on both occasions as in the example of choice function for nurse rostering problem and simple random for patient admission scheduling problem.

Pillay and Banzhaf [16], used genetic programming (GP) for the evolution on hyper-heuristics framework to solve uncapacitated examination timetabling problems. In their study, the GP framework has a sequence of constructive low-level heuristics (such as largest degree, largest enrolment, largest weight degree, saturation degree and highest cost) and specifies their order in genetic solution. Note that each genetic solution contains a sequence of low-level heuristic ordered to construct a single timetable. The genetic algorithm uses the generational control model, and its iterative run is terminated once the maximum number of generations has been reached. The hyper-heuristic that has produced the timetable with lowest violation of hard constraints and soft constraints during the run is retained as the final solution. From the research, it appears that the genetic programming system was comparable to the other search algorithm, and in some cases it can produce better quality timetables.

Previously we investigated harmony search as high level heuristic in the hyper-heuristic framework for solving examination timetabling problem [17]. The original idea was to evolve a sequence of low-level heuristics in order to produce good quality solutions to given problems. In this paper we demonstrate the generality of our approach by applying the method to solve the nurse rostering problem. The performance of the proposed hyper-heuristic is evaluated using a dataset established by the First International Nurse Rostering Competition (INRC2010) organized by the CODeS research group at Katholieke Universiteit Leuven in Belgium, SINTEF Group in Norway and the University of Udine in Italy. The dataset comprises three tracks determined based on complexity and size: sprint, medium, and long. Each track is further subdivided into early, late, hidden and hint. Interestingly, the proposed method HSHH obtained promising results in comparison with the other methods in the literature.

This paper is organized as follows: Section II discusses about the nurse rostering problem and benchmark dataset INRC2010. The details of harmony search hyper-heuristic (HSHH) algorithm is presented in Section III. Section IV discusses the computational results and comparison with other results that have been published. Finally, the conclusion and future works are provided in Section V.

II. NURSE ROSTERING PROBLEM

The Nurse Rostering Problem (NRP) can be defined as assigning a set of nurses to a set of different shifts on daily basis over given time periods [11]. Each nurse has a specific title with skills determined by qualification and experience. Furthermore, each nurse is employed through a contract agreed upon by the hospital administration. This contract determines the nurse job specifications as formulated in the soft constraint. Like other scheduling problems, NRP is also divided into two different types of constraints: *hard* and *soft* constraints as shown in Table 1. The hard constraints must be fulfilled to obtain a *feasible* roster while the violations of soft constraints are allowed but should be avoided, if possible. The quality of the roster is evaluated based on the fulfilment of the soft constraints. Based on the above, the basic objective is to obtain a feasible roster with high quality. However, it is almost impossible to find a roster that satisfies all constraints, since this problem is classified as a combinatorial optimization problem [18], and [19]. The nurse roster is evaluated using the objective function formalized in equation (1) that adds up the penalty of soft constraint violations in a feasible roster.

$$\min f(\boldsymbol{x}) = \sum_{s=1}^{10} c_s . g_s(\boldsymbol{x}) \qquad (1)$$

Note that s refers to the index of the soft constraint, c_s refers to the penalty weight for the violation of the soft constraint s, and $g_s(\boldsymbol{x})$ is the total number of violations for the soft constraint s in solution roster $\boldsymbol{x}$.

TABLE I: INRC2010 Hard and Soft constraints

Symbol	*The constraint*
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.
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. The nurse can request to work or not to work on certain day.
S_8	Requested shift-on/off. The nurse can request to work or not to work with particular shift type on a certain day.
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.

III. HARMONY SEARCH HYPER-HEURISTICS

Recently we investigated a harmony search algorithm (HSA) as high level heuristic in the hyper-heuristic framework named as Harmony Search Hyper-heuristic (HSHH). The original idea was to apply a sequence of low-level heuristics to a selected solution in order to produce good quality solutions to given problems. HSHH consists of two different search spaces: *heuristic search space* and *solution search space*. Heuristic search space contains sets of heuristic vectors (or individual) where every vector is a heuristics sequence. The representation of heuristic vector is a sequence of integers each of which represents one low-level heuristic. This sequence of heuristic, tells us which low level heuristic to use and in what order to apply them. Solution search space consists of sets of complete feasible solutions. In this study, we called this two search space: Heuristic Harmony Memory for heuristic search space and Solution Harmony Memory for solution search space. Details about the method will be discussed in the next sub-sections.

A. Hyper-heuristic framework

Basically HSHH has five main steps as follows:

Step 1: Initialization. The HSHH begins by setting the harmony search parameters: harmony memory size (HMS), harmony memory consideration rate (HMCR), number of iterations (NI) and Harmony Memory Length (HML) which represents the length of heuristic vector.

Step 2: Initializing of Harmony Memory. In initializing Heuristic Harmony Memory (HHM) and Solution Harmony Memory (SHM), the HSHH, firstly constructs the feasible solution by using heuristic ordering as in [3]. In order to fill the SHM, this process will be repeated until HMS is completed (see equation (3)). The different heuristic vectors in the HHM are initialized by using the random technique to select from the available set of heuristics. This process will also be continued until the HMS is completed (see equation (2)).

Step 3: Improvise a new heuristic vector. In this step, a new heuristics vector ***h'*** $= (h'_1, h'_2, \ldots h'_{HML})$ is generated from scratch, based on two HSA operators: *memory consideration,* and *random consideration*.

$$HHM = \begin{bmatrix} h_1^1 & h_2^1 & \cdots & h_{HML}^1 \\ h_1^2 & h_2^2 & \cdots & h_{HML}^2 \\ \vdots & \vdots & \ddots & \vdots \\ h_1^{HMS} & h_2^{HMS} & \cdots & h_{HML}^{HMS} \end{bmatrix} \quad (2)$$

$$SHM = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_N^{HMS} \end{bmatrix} \begin{bmatrix} f(x^1) \\ f(x^2) \\ \vdots \\ f(x^{HMS}) \end{bmatrix} \quad (3)$$

In *memory consideration operator*, the new heuristic index of h'_i in the new heuristic vector (***h'***) is randomly selected from the historical indexes (e.g. $h_i^1, h_i^2, \ldots.. h_i^{HMS}$), stored in the heuristic harmony memory with probability (i.e., w.p.) of HMCR, where HMCR ϵ [0, 1] .

For *Random consideration operator*, the new heuristic index is randomly assigned from the set of heuristics h'_i ϵ *{h1, h2, h3, h4}* with probability of (1-HMCR) as in equation (4).

$$h'_i \in \begin{cases} \{h_i^1, h_i^2, \ldots.. h_i^{HMS}\} & w.p.\ HMCR \\ \{h1, h2, h3, h4\} & w.p.\ (1 - HMCR) \end{cases} \quad (4)$$

Then the new harmony of heuristic vector ***h'*** will be applied to a selected solution ($\boldsymbol{x}^{rand}$) where $\boldsymbol{x}^{rand}$ is randomly selected from SHM ($\boldsymbol{x}^{rand} \in (\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{HMS})$). The HSHH used random selection to select the solution from the SHM to avoid the local optima. In this process, the heuristic in ***h'*** will be executed sequentially to the selected solution ($\boldsymbol{x}^{rand}$). The process will continue until all the heuristics in ***h'*** have been applied, and a new solution (***x'***) will be produced. Algorithm 1 shows the pseudo-code for step 3 in HSHH.

Algorithm 1: *Pseudo-code for selecting heuristic and generating new heuristic vector in HSHH*

```
h'= 0; //vector of heuristic sequences
for $l$ = 0,…,HML do
    if (U(0,1) ≤ HMCR) then
        $h'_i \in \{h_i^1, h_i^2, \ldots.. h_i^{HMS}\}$ ; //Memory consideration;
    else
        $h'_i$ ϵ {h1, h2, h3, h4}; //Random consideration;
    end if
end for
$x^{rand} \in (x^1, x^2, \ldots, x^{HMS})$; //Select random solution from SHM;
x' = apply h' to $x^{rand}$ ;
```

Step 4: Update HHM and SHM. In hyper-heuristic framework, this step is called move acceptance mechanism. HSHH will decide either to accept or neglect the new heuristic vector ***h'***. In this process, the new solution (***x'***) will be evaluated using the objective function. The new solution must be a complete and feasible solution. If the new solution is better than the selected solution ($\boldsymbol{x}^{rand}$) in SHM, the new ***h'*** and ***x'*** will be accepted and saved in the memory (***h'*** in HHM and ***x'*** in SHM) and the heuristic vector $\boldsymbol{h}^{rand}$ and solution $\boldsymbol{x}^{rand}$ will be excluded from the memory (i.e., HHM and SHM). The pseudo-code for this process is shown in Algorithm 2.

Algorithm 2: *Pseudo-code for move acceptance mechanism in HSHH*

```
if ($f(x') < f(x^{rand})$) then
    accept and include h' and x' in memory;
    exclude $x^{rand}$ and $h^{rand}$ from memory;
end if
```

Step 5: Check the stop criterion. Step 3 and step 4 in this approach are repeated until the stop criterion (i.e., NI) is met.

B. Low-level heuristic

As in a previous work of Harmony Search Hyper-Heuristic (HSHH), several heuristics have been utilized as low-level heuristics depending on the problem domain. For this paper, we used four low-level heuristics as in [7]. They can be summarized as follows:

- ***h1***: *Move one shift*. This heuristic is used to move the shift of current nurse to another nurse who is selected randomly while maintaining feasibility.
- ***h2***: *Swap one shift*. In this heuristic, the shift allocated to a specific nurse is exchanged with the shift of another nurse while maintaining feasibility. Note that both nurses are assigned to two different shifts on the same day.
- ***h3***: *Shuffle moves*. This heuristic swaps patterns between the worst nurse schedules based on their penalty values from the roster with a random nurse. The length of the patterns to be exchanged is specified by day, beginning with one day and increased by a day each time until the scheduling period is completed.
- ***h4***: do nothing.

IV. Experiments And Results

In this section, Harmony Search Hyper-heuristic is evaluated using the INRC2010 dataset for nurse rostering problems. The proposed method is coded in Microsoft Visual C++ 6 and the experiments were conducted under windows 7 on Intel processor with 2G RAM. After running several experiments we decided to set the parameters for our hyper-heuristic method as shown in Table II. We chose to test the proposed method with early track for sprint, medium and long problem instances in INRC2010. The characteristics of the datasets are shown in Table III. This table includes the number of nurses, shifts, nurse skills, contracts, and unwanted patterns. Furthermore, the nurse preferences such as day off and shift off are also included. We run the experiment 10 times for each problem instance for the statistical purpose.

TABLE II: Parameters setting for HSHH

Parameter	Values
HMS	10
HMCR	0.99
Heuristic HML	10
NI	50000

TABLE III: INRC2010 dataset characteristics for early track

Information	Sprint Early	Medium Early	Long Early
Shifts	4	4	5
Skills	1	1	2
Contracts	4	4	3
Unwanted	3	0	3
Day Off	√	√	√
Shift Off	√	√	√
Period	1-28/01/2010	1-28/01/2010	1-28/01/2010

Table V provide the comparison results of the HSHH and the other comparative methods that are working at the same dataset. The different comparative methods are provided as shown in Table IV. The first column in Table V is the averages running time required for each problem instances using HSHH method. The numbers in others column (i.e., second column until the last column) in Table V referred to the penalty value of the soft constraint violations (lowest is the best). The best results are highlighted in bold.

TABLE IV: Comparison Methods for INRC2010

Key	*Method*	*Author(s)*
M_1	Hyper-heuristic combined with a greedy shuffle approach.	Bilgin et al. [20]
M_2	Variable Depth Search Algorithm and Branch and Price Algorithm.	Burke and Curtois [21]
M_3	Tabu search with restart mechanism.	Lu and Hao [22]
M_4	Constraint Optimization Problem solver.	Nonobe [23]
M_5	Integer programming with set of neighbourhood structures.	Valouxis et al. [24]
M_6	Global best Harmony Search with a new pitch adjustment designed.	Awadallah et. al. [7]
M_7	Harmony search algorithm with greedy shuffle.	Awadallah et. al. [8]

As shown in Table V, HSHH is able to produce competitive results as the other methods. The proposed method obtained better results compared to the results produced by Global best Harmony Search (M_6) and Hybrid Harmony Search (M_7) especially for medium and long problem instances. In the early track for sprint (i.e., *sprint_early05* and *sprint_early08*), the proposed method is able to produce best published result as other methods in the literature, while the performance of the proposed method is comparable with other comparative methods for medium and long track instances.

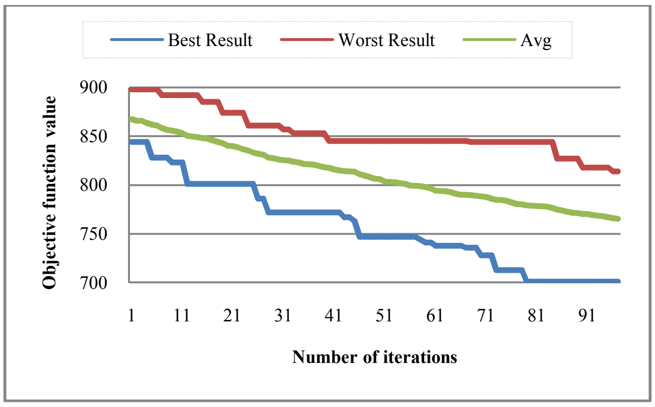

Fig 1: Distribution results in SHM for *long_early01* instance.

Fig 1 shows the distribution of results in solution harmony memory (SHM) between best results, average and worst results. The graph in Fig 2 shows how frequent the heuristics were applied for the complete iterations. From the graphs in Fig 2, it can be shown that heuristics H3 (or *h3*) is more frequently applied compared to the other heuristics (i.e., *h1, h2, h4*) and heuristic H4 (or *h4*) are less applied.

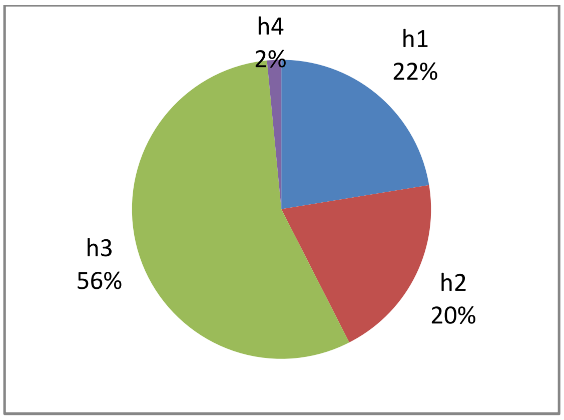

Fig 2: Heuristics usage for the complete iterations for *long_early01* instance.

V. Conclusion And Future Directions

This paper presented Harmony search based hyper-heuristics (HSHH) for solving nurse rostering problem using INRC2010 datasets. This algorithm employed harmony search at the high level to evolve a sequence of improvement low level heuristic. Four different neighborhood structures are used as low level heuristics. The main objective of testing the HSHH with INRC2010 is to test the generality of the HSHH method, but through the experimental results it can be shown that the HSHH are able to produce comparable results with those produced by the INRC2010 winners' (i.e., M_1, M_2, M_3, M_4 and M_5). For future research directions, we plan to adapt learning mechanisms within the HSHH algorithm in order to improve the heuristic selection.

TABLE V: The results and average running time for the proposed method and comparison with others method

Dataset	**Avg. Time (s)**	**Our Approach**	**M_1**	**M_2**	**M_3**	**M_4**	**M_5**	**M_6**	**M_7**
sprint_early01	*2398.11*	58	57	**56**	**56**	**56**	**56**	58	**56**
sprint_early02	*2251.70*	60	59	**58**	**58**	**58**	**58**	60	62
sprint_early03	*2372.22*	53	**51**	**51**	**51**	**51**	**51**	53	57
sprint_early04	*2363.67*	62	60	**59**	**59**	**59**	**59**	62	65
sprint_early05	*2377.15*	**58**	**58**	**58**	**58**	**58**	**58**	59	61
sprint_early06	*2411.84*	55	**54**	**54**	**54**	**54**	**54**	56	56
sprint_early07	*2287.78*	58	**56**	**56**	**56**	**56**	**56**	58	59
sprint_early08	*2423.82*	**56**	**56**	**56**	**56**	**56**	**56**	57	**56**
sprint_early09	*2301.39*	57	**55**	**55**	**55**	**55**	**55**	57	59
sprint_early10	*2363.93*	54	**52**	**52**	**52**	**52**	**52**	53	54
medium_early01	*3274.74*	249	242	**240**	**240**	241	**240**	270	274
medium_early02	*3313.16*	251	241	**240**	**240**	**240**	**240**	275	275
medium_early03	*3305.65*	247	238	**236**	**236**	**236**	**236**	265	281
medium_early04	*3308.62*	248	238	**237**	**237**	238	**237**	263	272
medium_early05	*3319.77*	315	304	**303**	**303**	304	**303**	334	324
long_early01	*2341.42*	214	**197**	**197**	**197**	**197**	**197**	256	332
long_early02	*4294.65*	245	220	**219**	222	**219**	**219**	299	392
long_early03	*4395.42*	248	**240**	**240**	**240**	**240**	**240**	286	342
long_early04	*4377.30*	317	**303**	**303**	**303**	**303**	**303**	356	404
long_early05	*4427.37*	298	**284**	**284**	**284**	**284**	**284**	337	376

References

[1] E. K. Burke, G. Kendall, and E. Soubeiga, "A tabu-search hyperheuristic for timetabling and rostering," *Journal of Heuristics,* vol. 9, pp. 451-470, 2003.

[2] B. Bilgin, P. De Causmaecker, B. Rossie, and G. V. Berghe, "Local search neighbourhoods for dealing with a novel nurse rostering model," *Annals of Operations Research,* vol. 194, pp. 33-57, 2012.

[3] E. K. 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, pp. 330-341, 2008.

[4] M. J. Brusco and L. W. Jacobs, "Cost analysis of alternative formulations for personnel scheduling in continuously operating organizations," *European Journal of Operational Research,* vol. 86, pp. 249-261, 1995.

[5] W. J. Gutjahr and M. S. Rauner, "An ACO algorithm for a dynamic regional nurse-scheduling problem in Austria," *Computers & Operations Research,* vol. 34, pp. 642-666, 2007.

[6] 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, pp. 77-93, 2010.

[7] M. A. Awadallah, A. T. Khader, M. A. Al-Betar, and A. L. a. Bolaji, "Global best Harmony Search with a new pitch adjustment designed for Nurse Rostering," *Journal of King Saud University-Computer and Information Sciences,* vol. 25, pp. 145-162, 2013.

[8] M. A. Awadallah, A. T. Khader, M. A. Al-Betar, and A. L. a. Bolaji, "Harmony search with greedy shuffle for nurse rostering," *International Journal of Natural Computing Research (IJNCR),* vol. 3, pp. 22-42, 2012.

[9] E. K. Burke, T. Curtois, R. Qu, and G. V. Berghe, "A scatter search methodology for the nurse rostering problem," *Journal of the Operational Research Society,* vol. 61, pp. 1667-1679, 2010.

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

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

[12] E. Burke, G. Kendall, J. Newall, E. Hart, P. Ross, and S. Schulenburg, "Hyper-heuristics: An emerging direction in modern search technology," in *Handbook of Metaheuristics*, ed: Springer, 2003, pp. 457-474.

[13] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and R. Qu, "A survey of hyper-heuristics," *Computer Science Technical Report No. NOTTCS-TR-SUB-0906241418-2747, School of Computer Science and Information Technology, University of Nottingham,* 2009.

[14] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *Journal of the Operational Research Society,* vol. 64, pp. 1695-1724, 2013.

[15] B. Bilgin, P. Demeester, M. Misir, W. Vancroonenburg, and G. V. Berghe, "One hyper-heuristic approach to two timetabling problems in health care," *Journal of Heuristics,* vol. 18, pp. 401-434, 2012.

[16] N. Pillay and W. Banzhaf, "A genetic programming approach to the generation of hyper-heuristics for the uncapacitated examination timetabling problem," in *Neves J, Santos MF, Machado J (Ed.), Progress in Artificial Intelligence, 13th Portuguese Conference on Artificial Intelligence*, 2007, pp. 223-234.

[17] K. Anwar, A. T. Khader, M. A. Al-Betar, and M. A. Awadallah, "Harmony search-based hyper-heuristic for examination timetabling," in *9th International Colloquium on Signal Processing and its Applications (CSPA)*, 2013, pp. 176-181.

[18] J. J. Bartholdi III, "A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering," *Operations Research,* vol. 29, pp. 501-510, 1981.

[19] H. H. Millar and M. Kiragu, "Cyclic and non-cyclic scheduling of 12 h shift nurses by network programming," *European Journal of Operational Research,* vol. 104, pp. 582-592, 1998.

[20] B. Bilgin, P. Demeester, M. Mısır, W. Vancroonenburg, G. V. Berghe, and T. Wauters, "A hyper-heuristic combined with a greedy shuffle approach to the nurse rostering competition," in *Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling PATAT*, 2010.

[21] E. K. Burke and T. Curtois, "An ejection chain method and a branch and price algorithm applied to the instances of the first international nurse rostering competition, 2010," in *Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling PATAT*, 2010, p. 13.

[22] Z. Lü and J.-K. Hao, "Adaptive Local Search for the first International Nurse Rostering Competition," *INRC2010 (http://www. kuleuven-kortrijk. be/nrpcompetition),* 2010.

[23] K. Nonobe, "INRC2010: An approach using a general constraint optimization solver," *INRC2010 (http://www. kuleuven-kortrijk. be/nrpcompetition),* 2010.

[24] C. Valouxis, C. Gogos, G. Goulas, P. Alefragis, and E. Housos, "A systematic two phase approach for the nurse rostering problem," *European Journal of Operational Research,* vol. 219, pp. 425-433, 2012.