JOURNAL OF INDUSTRIAL AND
MANAGEMENT OPTIMIZATION
Volume **17**, Number **4**, July **2021**

doi:10.3934/jimo.2020058

pp. **2051–2072**

A HYBRID VARIABLE NEIGHBOURHOOD SEARCH AND DYNAMIC PROGRAMMING APPROACH FOR THE NURSE ROSTERING PROBLEM

Mohammed Abdelghany* and Amr B. Eltawil

Industrial Engineering and Systems Management
Egypt-Japan University of Science and Technology
New Borg Elarab City, Alexandria 21934, Egypt

Zakaria Yahia

Mechanical Engineering Department
Faculty of Engineering, Fayoum University
Fayoum, Egypt

Kazuhide Nakata

Industrial Engineering and Economics
School of Engineering, Tokyo Institute of Technology
Tokyo, Japan

(Communicated by Mutsunori Yagiura)

Abstract. Nurse Rostering is the activity of assigning nurses to daily shifts in order to satisfy the cover requirements, taking into account the operational requirements and nurses' preferences. The problem is usually modeled as sets of hard and soft constraints with an objective function to minimize violations of soft constraints. The nurse rostering problem is known to be NP-hard. Many metaheuristics were used to tackle this problem. One of the frequently used heuristics is the Variable Neighbourhood Search (VNS). The VNS is usually used as a standalone method or in integration with another exact or heuristic method. In this paper, a new hybrid VNS and Dynamic Programming based heuristic approach is proposed to handle the nurse rostering problem. In the proposed approach, two perturbation mechanisms are adopted simultaneously. The proposed approach is tested on two different benchmark data sets. A comparison with state-of-the-art methods from literature revealed the competitive performance of the proposed approach.

1. **Introduction.** Hospitals have to be staffed 24 hours a day, seven days a week. Nurses are one of the most crucial resources in hospitals. Therefore, proper nurses' schedules have to be prepared frequently (daily/weekly/monthly) to maintain the quality of healthcare services. The Nurse Rostering Problem (NRP) deals with the assignment of nurses to daily shifts in order to satisfy cover requirements while considering the legal regulations, nurses' preferences and some other requirements. Switching from the manual nurse rostering process to an automated one is resulting

2020 *Mathematics Subject Classification.* Primary: 90B35; Secondary: 90C59, 90C39.

Key words and phrases. Variable neighbourhood search, heuristics, dynamic programming, nurse rostering, timetabling.

* Corresponding author: Mohammed Abdelghany.

in higher quality rosters with reduced institutional costs and better nurses' satisfaction [13]. Thus, the NRP has attracted many researchers to examine different solution approaches during the last few decades [17]. Multiple Integer Programming (IP) models were reported in literature to solve real NRPs. A Mixed Integer Programming (MIP) model was presented in [24] to develop nurses' timetables in a Kuwaiti healthcare unit. Computational results showed the advantage of the mathematically generated timetables over those manually generated. In [30] another IP model was developed to generate optimal nurses' schedules for an emergency center of a Specialist Hospital. An extended model of standard IP formulation for NRP was proposed in [26] to solve the multistage NRP of the Second International Nurse Rostering Competition [14], where competitive results were reported.

However, most of the personnel rostering problems are hard to be solved in polynomial time, especially if they involve succession constraints such as those that limit the maximum and minimum allowed number of consecutive working days or days-off [31]. Indeed, almost all NRPs in reality imply such succession constraints. Therefore, due to the computational complexity of the problem, exact solution methods mostly fail to solve large size instances in a reasonable time [33], [28]. Therefore, heuristic solution methods are adopted to obtain good solutions within reasonable computation time. As the focus of this work is mainly on the NRP, readers are referred to [17] and [7] for more information about the general personnel scheduling/rostering problems and solution approaches.

There are many metaheuristic methods that were adopted for solving the NRP. A harmony search algorithm was proposed in [20] for a real NRP at a Malaysian hospital. A particle swarm optimization method was proposed in [18] to maximize the nurses work happiness. Another particle swarm optimization algorithm was reported in [34] to maximize the fairness of allocation of different shift types and days between nurses over the planning horizon. A chemical reaction optimization heuristic was proposed in [35] for handling the NRP. In addition, there are multiple IP based heuristic solution approaches that were reported in literature, such as the IP based Ant Colony Optimization approach proposed in [8], where IP was used to facilitate finding feasible solutions for highly constrained problems. Another IP technique with improved cut generation strategies and primal heuristics was proposed in [29] and tested on instances of the First International Nurse Rostering Competition [21], revealing competitive performance to other heuristics.

The Variable Neighbourhood Search (VNS) heuristic was used a lot in the literature either as a standalone method or in integration with an exact or other metaheuristic methods. Four neighbourhood search structures that were used within a Tabu search procedure and an iterated local search procedure were presented in [6]. The two procedures were tested on real NRP instances, and both of them brought good results in terms of the solution quality and computational time. A hybrid VNS, heuristic ordering and back-tracking algorithm was proposed in [10]. In this hybrid algorithm, the VNS was based on two neighbourhood search structures; making more shifts assignments to the nurses and swapping of a pair of already assigned shifts. The VNS was followed by a repairing step where the worst individual schedules are destroyed and the nurses are reassigned again using the heuristic ordering method. The back-tracking method was used to further improve the quality of the produced schedules. The hybrid algorithm showed comparable results with a commercial genetic algorithm.

An adaptive neighbourhood search algorithm was presented in [23] to handle the instances of the First International Nurse Rostering Competition [21]. In their algorithm, they applied a combined two neighbourhood structures along with three intensification and diversification search strategies. Their algorithm was able to provide new upper bounds for 12 instances and matching the best known results for 39 instances out of 60 instances. A two-stage VNS algorithm for the NRP was proposed in [33], [32], where the first stage deals with the assignment of nurses to working days, then the second stage deals with the assignment of nurses to shift types. Experimental results showed that, in addition to the superiority of the two-stage VNS algorithm compared to six other effective algorithms [32], very comparable results with the five finalists of the First International Nurse Rostering Competition were reported [33].

A matheuristic approach of VNS and Integer Programming (IP) was proposed in [12]. The IP was used to solve a subproblem that considers all the hard constraints and a subset of the soft constraints. Then a VNS was used to further improve the resulting solutions of the IP by considering the excluded soft constraints from the IP stage. Promising results were reported in comparison with a commercial genetic algorithm. Another matheuristic approach was reported in [16] for a NRP in an Italian private hospital. In this approach, 12 different neighbourhood structures were used. The problem was decomposed into subproblems based on a set of parameters. For each subproblem, some parts of the main problem were fixed and the remaining were solved using an IP solver. The approach showed a good behaviour in terms of solutions quality.

In [28], a hybrid algorithm was presented where Integer Programming and Constraint Programming were used in co-operative manner to handle the NRP. Their algorithm was able to solve a wide range of benchmark instances effectively. A flexible mixed integer programming model introduced in [22] that is able to model many of the available benchmark instances. A simulated annealing heuristic approach was then used to handle their model. Although, their approach was able to produce good results for many of the tested benchmark instances, it could not perform well for some instances where solutions need to be produced in a very short time. A matheuristic approach was proposed in [19] for a set of instances from the Second International Nurse Rostering Competition [14]. In their approach, a heuristic based on VNS was used to speed up the convergence of a column generation method. Their approach was able to improve the best known solutions for the tested instances.

Indeed, almost all the reported VNS based approaches adopted simple conventional neighbourhood structures, where a pair of assignments or a pair of blocks of assignments are swapped together. None of the reported VNS based approaches implied soft constraints related neighbourhoods. They usually handle all soft constraints in similar ways regardless the different weights that usually assigned to each constraint. However, in the soft constraints related neighbourhoods, the algorithm is blind to the total penalty of the solution, focusing on satisfying a specific soft constraint [9]. Therefore, giving more priority for satisfying constraints with higher weights is expected to speed up the solution convergence. This priority can be achieved as will be demonstrated in section 3.1.

Although the NRP is considered to be a multistage decision problem (the nurses' assignment at each day) where approaches like Dynamic Programming (DP) may fit, it is rare to find a work in literature where the DP has been used for solving the NRP. That is due to a fact known as "the curse of dimensionality", where the search space and computation time increase exponentially with the problem size. However, some bounding strategies or approximation methods can be adopted to tackle this drawback of the DP. Readers are referred to [5], [3] as examples for the approximation and bounding strategies. Recently, a Bounded Dynamic Programming algorithm [1] was proposed for solving a set of instances of the First International Nurse Rostering Competition. It was able to provide near optimal solutions for some medium size instances. References [2] and [11] discussed the use of the shortest path technique of the DP approach to handle the subproblems of the nurse rostering problem – in particular to generate feasible/optimal schedules for individual nurses.

In [27], an IP solver was used within a VNS algorithm through a ruin-and-recreate strategy. In their approach, after each iteration of the VNS algorithm, the most contributed parts of the solution to the overall penalty are destroyed and recreated using an IP solver. According to computational tests of the algorithm on benchmark instances, although the algorithm outperformed two other state-of-the-art algorithms in most of the tested instances, it failed to maintain a similar good performance with the largest size instance. This can be traced to the large size even for the destroyed parts of the solution that might be hard to be solved by the IP solver within the assigned time. However, replacing such exact method (IP) with another method that can handle the large size instances deserves investigation. Therefore, we seek in this work to investigate the use of a Dynamic Programming based heuristic instead of the IP solver within a destroy-and-recreate strategy in the VNS algorithm. So, in this paper, a new hybrid Variable Neighbourhood Search and Dynamic Programming based heuristic (VNS-DP) algorithm is proposed for the NRP.

The contribution of this work can be summarized in four main features of the solution method. First, proposing a handling method for the soft constraints that gives more priority for satisfying the soft constraints with the highest penalty weight. This is achieved using two new High-Weight-Constraints-Focused neighbourhood structures. Second, the adoption of two perturbation mechanisms – with different disruption levels – that are used iteratively to improve the solution; a classical mechanism that is fully disrupting the solution structure, and the destroy-and-recreate mechanism that is disrupting the solution structure partially. Third, embedding of the Dynamic Programming based heuristic in the VNS procedure through the destroy-and-recreate perturbation mechanism in order to help improve the individual nurses' schedules as well as, simultaneously, diversify the search process. Lastly, the introduction of a new bounding strategy that is based on filtering of potential infeasible assignments in order to limit the search space of the DP algorithm.

The rest of this paper is organized as follows. In section 2, a description for the studied NRP is provided. Then, a description for the proposed hybrid VNS and DP based heuristic approach is presented in section 3. In section 4, experimental results and performance evaluation of the proposed approach on two different NRP benchmark data sets is provided. Also, a comparison with other state-of-the-art methods from literature is presented. Finally, section 5 concludes the paper and suggests future work directions.

2. Problem description.

The studied NRP in this paper was introduced in [15], where 24 benchmark instances were provided. The provided instances vary in terms of size (number of nurses, shift types, and planning horizon) from very small instances to huge ones. Table 1 summarizes the main characteristics of these instances.

This problem has two types of constraints; (1) hard constraints that must be satisfied to obtain a feasible solution, and (2) soft constraints that may be violated, but the violations are penalized in the objective function. There are 10 hard constraints and two main soft constrains as follows.

2.1. Hard constraints.

1. Single assignment; a nurse cannot be assigned more than one shift per day.
2. Shift patterns; certain shift types cannot follow each other on consecutive days.
3. MaxShifts; each nurse has a maximum number of shifts of each shift type that can be assigned to him/her during the planning horizon.
4. MaxWorkingTime; each nurse cannot work more than a certain amount of time during the planning horizon.
5. MinWorkingTime; each nurse cannot work less than a certain amount of time during the planning horizon.

Table 1. Benchmark instances main characteristics

#	Days	Nurses	Shift types	Days-off	Shifts on/off
01	14	8	1	8	26
02	14	14	2	14	62
03	14	20	3	20	64
04	28	10	2	20	71
05	28	16	2	32	106
06	28	18	3	36	135
07	28	20	3	40	168
08	28	30	4	60	225
09	28	36	4	72	232
10	28	40	5	80	284
11	28	50	6	100	336
12	28	60	10	120	422
13	28	120	18	240	841
14	42	32	4	128	359
15	42	45	6	180	490
16	56	20	3	120	280
17	56	30	4	160	480
18	84	22	3	176	414
19	84	40	5	320	834
20	182	50	6	900	2318
21	182	100	8	1800	4702
22	364	50	10	1800	4638
23	364	100	16	3600	9410
24	364	150	32	5400	13809

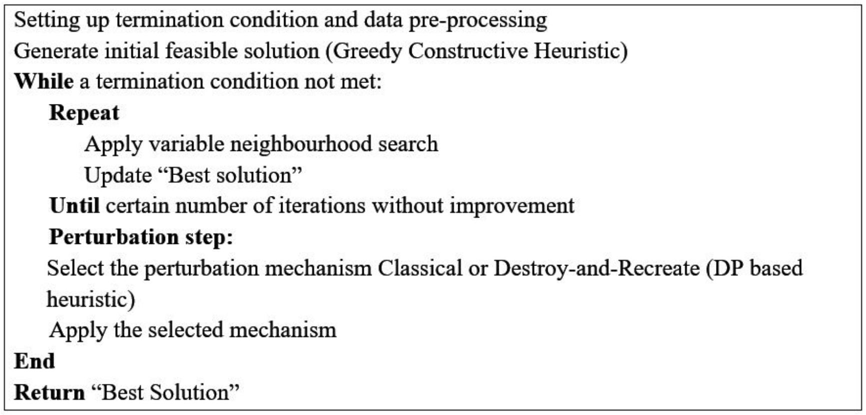
6. MaxConsecDays; each nurse cannot work more than a certain number of consecutive days before having a day-off.
7. MinConsecDays; each nurse cannot work less than a certain number of consecutive days before having a day-off.
8. MinConsecDays-off; each nurse cannot take less than a certain number of consecutive days-off.
9. MaxWeekends; each nurse cannot work more than a certain number of weekends during the planning horizon.
10. Days-off; nurses cannot work on certain days of the planning horizon.

2.2. Soft constraints.

1. Shifts on/off requests; a nurse may request to be on or off in a certain shift of a specific day. If this request is violated, a solution penalty is applied.
2. Cover requirements; if the required number of nurses for a certain shift of a certain day is not satisfied, a solution penalty is applied as follows. The absolute difference between the required and the assigned number of nurses is multiplied by the plenty weight for shortage or surplus according to the case.

Indeed, as the shortage in the cover requirement is more risky and most probably would negatively affect the patients' satisfaction and the service level, a larger penalty weights are usually associated with such violations. Therefore, the objective function of the problem is to minimize the total weighted penalty index of the solution that results from the violations for the soft constraints.

3. The proposed VNS-DP approach. The framework of the proposed hybrid approach is demonstrated in the pseudo code shown in Figure 1. The approach starts by setting up the termination condition as well as data pre-processing. Then, a greedy constructive heuristic is adopted to generate an initial feasible solution that satisfies all the hard constraints of the problem. The idea of the constructive heuristic is based on generating an individual feasible schedule for each nurse through a random shift assignment day by day. At each day the feasibility of the partial solution is checked till the last day of the planning horizon. If the partial solution is infeasible, another random shift type is assigned to that day and rechecked. By



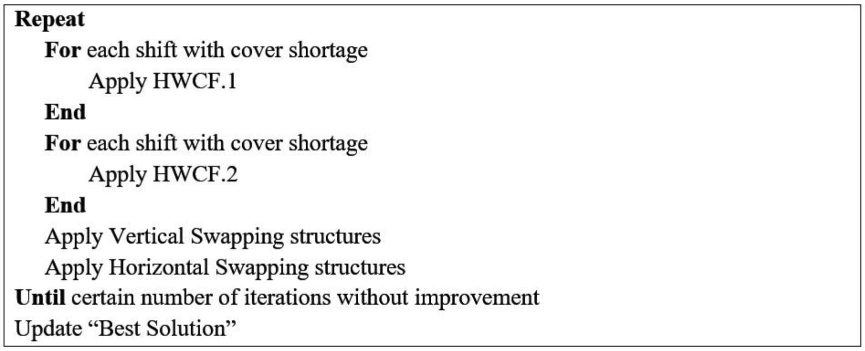
```
Setting up termination condition and data pre-processing
Generate initial feasible solution (Greedy Constructive Heuristic)
While a termination condition not met:
    Repeat
        Apply variable neighbourhood search
        Update "Best solution"
    Until certain number of iterations without improvement
    Perturbation step:
    Select the perturbation mechanism Classical or Destroy-and-Recreate (DP based
    heuristic)
    Apply the selected mechanism
End
Return "Best Solution"
```

Figure 1. Pseudo code for the hybrid VNS-DP approach

generating feasible schedules for each nurse individually, the initial feasible solution is formed.

After that, an iterated local search procedure is performed using sets of neighbourhood search structures. After a certain number of iterations without improvement in the solution quality, a perturbation procedure is performed mainly to help escape from local optimal solutions. Two different perturbation mechanisms are employed; a classical perturbation mechanism and a destroy-and-recreate mechanism that will be described later. At each iteration of the algorithm, the perturbation mechanism is selected randomly with a fifty-fifty chance (50% probability) of selection for each mechanism. Until the termination condition is met, the algorithm is repeated. Then, the best solution found so far is reported.

3.1. **The Variable Neighbourhood Search.** When an initial solution is generated using the greedy constructive heuristic, a variable neighbourhood local search algorithm is applied to improve the solution locally using sets of neighbourhood structures that are iterated until no further improvement can be achieved. Figure 2 shows a pseudo code for the neighbourhood search algorithm. In the search process, three different sets of neighbourhood structures are used as follows.



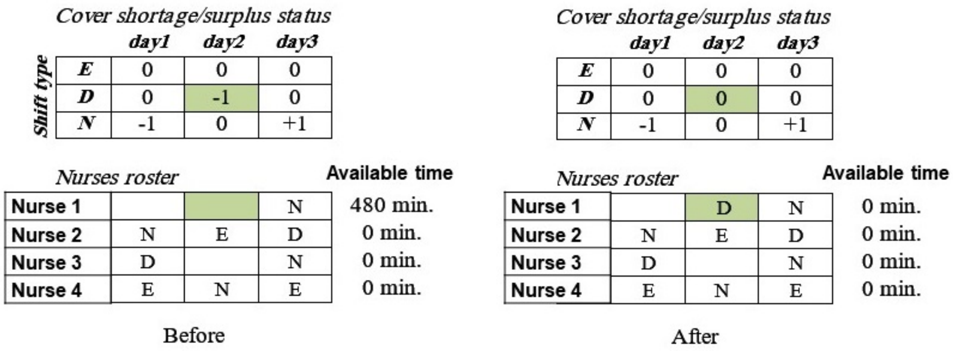
```
Repeat
    For each shift with cover shortage
        Apply HWCF.1
    End
    For each shift with cover shortage
        Apply HWCF.2
    End
    Apply Vertical Swapping structures
    Apply Horizontal Swapping structures
Until certain number of iterations without improvement
Update "Best Solution"
```

FIGURE 2. Pseudo code for VNS algorithm

3.1.1. *High-Weight-Constraints-Focused Structures.* In this section, a new way for handling the soft constraints is proposed to give more priority for satisfying the soft constraints with the highest weight in the objective function. This way is expected to speed up the solution convergence. This priority is achieved by using two new High-Weight-Constraints-Focused (HWCF) Structures. These two structures are focusing on improving the highest weighted soft constraint of the problem. The soft constraint of interest here is the cover requirements constraint. As mentioned in section 2, the cover shortage usually has the highest penalty weights in the objective function. Therefore, focusing on reducing this soft constraint violations is expected to achieve rapid improvements in the overall objective function value. For this aim, these two new HWCF structures are introduced.

HWCF.1: This structure aims to fully utilize the available time for each nurse as much as possible to reduce the cover shortages. This is achieved by assigning shifts to the current off-days of the nurse as long as the solution is feasible. Before applying this structure, there are two conditions to be checked; the available time for a nurse has to be longer than or equal to the duration of the intended shift to be assigned to the nurse, and the available number of shifts of the intended shift type has to be larger than zero. So, for a day/shift with a cover shortage, all nurses that are off in that day and satisfying the two conditions are candidates for the application of this structure. In this step, only feasible and solution improving applications are accepted. This structure is repeated for all shifts with a cover shortage. Figure 3 demonstrates with an illustrative example the application of this structure. The tables on the top represent the cover shortage or surplus status for each day and shift. The negative values represent cover shortages while the positive values represent the cover surplus. A zero means that the exact cover requirement has been assigned. E, D and N represent three shift types; Early, Day and Night shifts.

HWCF.2: This structure focuses on using the cover surplus in some shifts to compensate for the shortages in the other shifts. Therefore, this structure helps to reduce the cover surplus as well as the cover shortages. As per the shown example in Figure 4, for a day/shift with a cover shortage, all off nurses at that day are candidates for the application of this structure. Therefore, for a candidate nurse, an assignment of the intended shift type is done in the cover shortage day, and a random selected working day of the nurse is set to be off. Similar to the previous structure, only the feasible and solution improving applications of the structure are accepted. The structure is repeated for all shifts with a cover shortage.



Cover shortage/surplus status

Shift type	day1	day2	day3
E	0	0	0
D	0	-1	0
N	-1	0	+1

Nurses roster

				Available time
Nurse 1			N	480 min.
Nurse 2	N	E	D	0 min.
Nurse 3	D		N	0 min.
Nurse 4	E	N	E	0 min.

Before

Cover shortage/surplus status

	day1	day2	day3
E	0	0	0
D	0	0	0
N	-1	0	+1

Nurses roster

				Available time
Nurse 1		D	N	0 min.
Nurse 2	N	E	D	0 min.
Nurse 3	D		N	0 min.
Nurse 4	E	N	E	0 min.

After

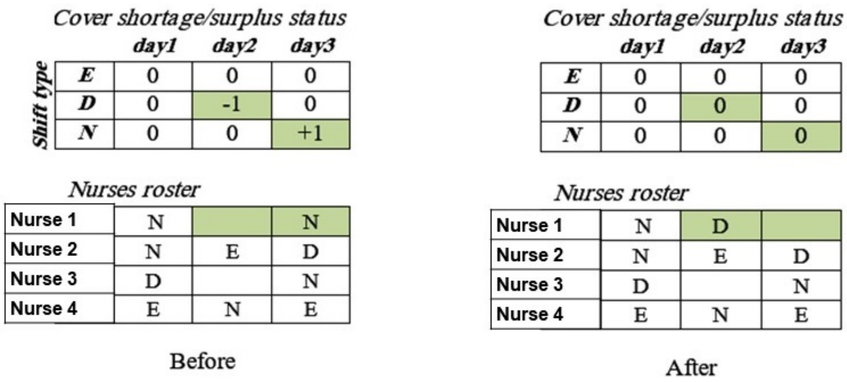
FIGURE 3. Example for HWCF.1 structure

3.1.2. *Vertical swapping structures.* In these structures, shifts are exchanged vertically between nurses on the same day. So, definitely they will not affect the cover requirements constraint. However, they are only improving the individual schedules of the nurses. They help to reduce the violations for shifts on/off requests of nurses. There are six vertical structures with different sizes that are considered. Examples for these vertical swaps are illustrated in Figure 5, where E, D, N and L represent Early, Day, Night, and late shift types.

- VS.1: two shifts are swapped between two different nurses at the same day.
- VS.2: two blocks of "two consecutive shifts" are swapped between two different nurses on two consecutive days.
- VS.3: two blocks of "three consecutive shifts" are swapped between two different nurses on three consecutive days.

- VS.4: two blocks of "four consecutive shifts" are swapped between two different nurses on four consecutive days.
- VS.5: two blocks of "five consecutive shifts" are swapped between two different nurses on five consecutive days.
- VS.6: two weekends (Saturday and Sunday) are swapped between two different nurses.

3.1.3. *Horizontal swapping structures.* These structures represent the horizontal exchanges of shifts between different days for the same nurse as shown in Figure 6. These horizontal swaps affect both the cover requirements and the nurses' requests



Cover shortage/surplus status

Shift type	day1	day2	day3
E	0	0	0
D	0	-1	0
N	0	0	+1

Nurses roster

Nurse 1	N		N
Nurse 2	N	E	D
Nurse 3	D		N
Nurse 4	E	N	E

Before

Cover shortage/surplus status

	day1	day2	day3
E	0	0	0
D	0	0	0
N	0	0	0

Nurses roster

Nurse 1	N	D	
Nurse 2	N	E	D
Nurse 3	D		N
Nurse 4	E	N	E

After

FIGURE 4. Example for HWCF.2 structure

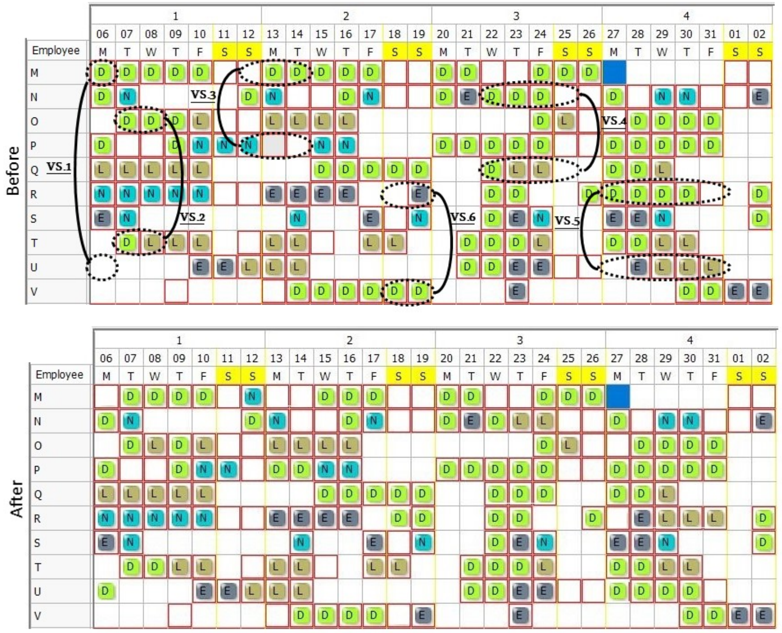


FIGURE 5. Examples of vertical swaps

constraints. Similar to the vertical structures, six horizontal structures with different sizes are considered as follows for a nurse.

- HS.1: two different shifts are swapped between two different days.
- HS.2: two blocks of "two consecutive shifts" are swapped horizontally.
- HS.3: two blocks of "three consecutive shifts" are swapped horizontally.
- HS.4: two blocks of "four consecutive shifts" are swapped horizontally.
- HS.5: two blocks of "five consecutive shifts" are swapped horizontally.
- HS.6: two weekends (Saturday and Sunday) are swapped horizontally.

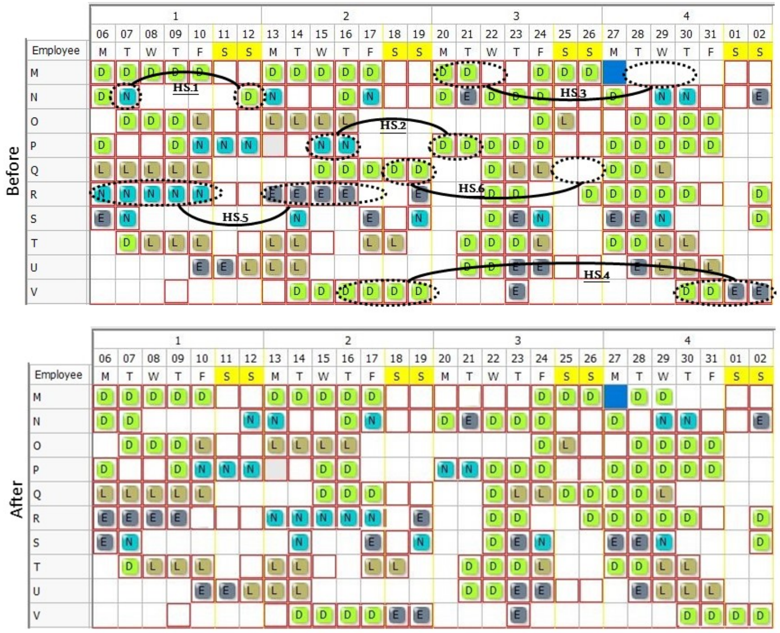


Figure 6. Examples of horizontal swaps

3.2. **The implemented perturbation mechanisms.**

3.2.1. *The classical perturbation mechanism.* In order to diversify the search process and to help escaping from local optimal solutions, a classical perturbation mechanism is applied after a certain number of local search iterations without improvement. In this classical mechanism, two similar sets of the vertical and horizontal swapping structures are applied. However, in order to disrupt the solution structure, all feasible swaps are accepted regardless of their effect on the solution quality. So, we may get a new worse solution with a different structure that can be refined locally through the VNS algorithm

3.2.2. *The destroy-and-recreate perturbation mechanism.* In this mechanism, a certain number of nurses are selected randomly. Schedules of the selected nurses are destroyed (fully unassigned). Then, a DP based heuristic algorithm is used to recreate schedules for the destroyed nurses. In Dynamic Programming, the problem is divided into smaller subproblems that are solved successively, then the original problem is solved by aggregating the subproblems' solutions [5]. The NRP requires multistage decisions (nurses' assignments at each day). Therefore, the problem can be described using dynamic programming according to the shown structure in Figure 7. The proposed DP structure can be described as follows.

- ***Stages;*** represent the D days of the planning horizon. Each day is a stage.
- ***States;*** represent the different possible combinations of nurses assignments to shift types at each stage (day). As shown in Figure 7, a state is represented by a rectangle (vector) which is divided into a number of apartments (cells) equal to the number of nurses (N). Each cell is filled by a letter (M, E, N or O) that represents the assigned shift type to that nurse. The different shift types are: the morning shift (M), the evening shift (E), the night shift (N) and the day off (O).
- ***Recursive function;*** the objective function to evaluate each path at different stages is to minimize the total solution penalty. So, the recursive function can be represented as follow.

$$TP_d^i = min_{j=(1,2,3...S)} \quad [TP_{(d-1)}^i + P_d^j] \qquad \forall i \tag{1}$$

Where; TP_d represents the total penalty of soft constraints violations from the first stage/day in the planning horizon till the current stage/day d. TP_{d-1} represents the total penalty of soft constraints violations from the first stage/day in the planning horizon till stage/day $(d-1)$. While i represents the different partial solutions available at a stage/day $(d-1)$, and j represents the different possible states (assignments combinations) at a stage/day d, S represents the number of candidate states in day d. P_d^j represents the penalty results from associating state j at stage/day d with partial solution i at day $(d-1)$. For each partial solution i, all the nurses assignments till the current day, the current day index d , as well as the associated total penalty are saved.

A major drawback of using DP is the curse of dimensionality, as the search space increases exponentially with the problem size, and accordingly the computation

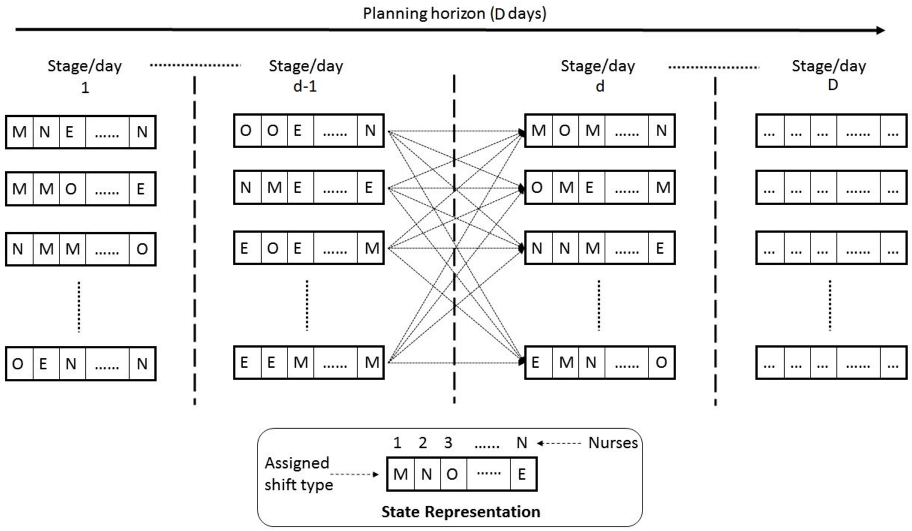
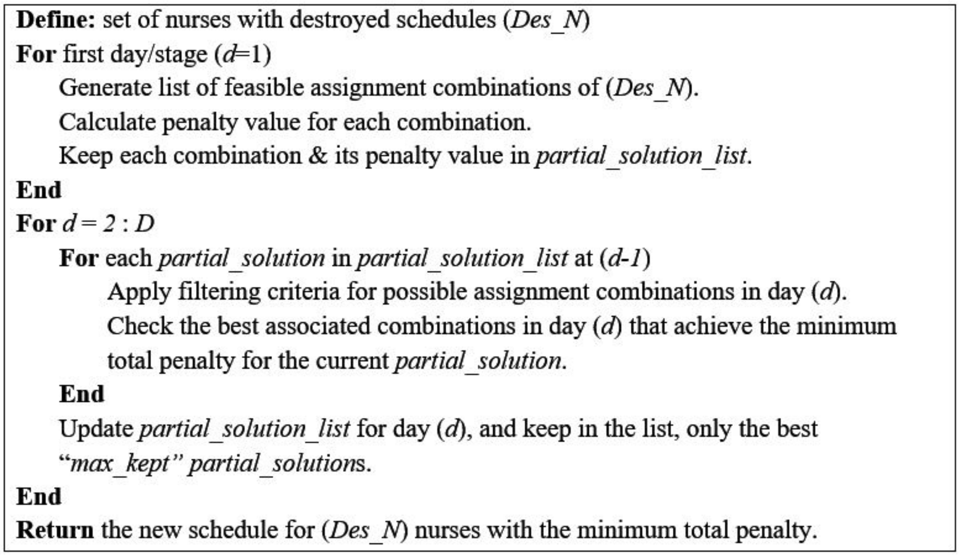


FIGURE 7. A dynamic programming structure for the nurse rostering problem



```
Define: set of nurses with destroyed schedules (Des_N)
For first day/stage (d=1)
    Generate list of feasible assignment combinations of (Des_N).
    Calculate penalty value for each combination.
    Keep each combination & its penalty value in partial_solution_list.
End
For d = 2 : D
    For each partial_solution in partial_solution_list at (d-1)
        Apply filtering criteria for possible assignment combinations in day (d).
        Check the best associated combinations in day (d) that achieve the minimum
        total penalty for the current partial_solution.
    End
    Update partial_solution_list for day (d), and keep in the list, only the best
    "max_kept" partial_solutions.
End
Return the new schedule for (Des_N) nurses with the minimum total penalty.
```

Figure 8. Pseudo code for the proposed Dynamic Programming based heuristic algorithm

time. To tackle this issue, a DP based heuristic algorithm is proposed in Figure 8. In this algorithm, two bounding strategies are adopted to limit the search space. The first strategy is to use a control parameter called *max_kept* which defines the maximum number of partial solutions that are allowed to be kept from a day to the following day. The partial solutions to be kept are selected to be the ones with the minimum total penalty.

The second strategy aims to filter the possible states (assignments combinations) in the next day from states known to be infeasible beforehand for each partial solution using a set of hard-constraints-based criteria that are applied for each nurse of the destroyed schedule according to the following order:

1. Check if there is a day-off request on the next day. If so, then the next day must be off, and all assignments combinations where this nurse is working on that day are removed from the list of possible states.
2. Check the maximum consecutive working days for the partial solution of the nurse. If the maximum allowed value is already reached, then the next day must be off, and all assignments combinations where this nurse is working on that day are removed from the list of possible states.
3. Check the maximum total working time for the partial solution of the nurse. If the remaining allowed working time is less than the length of the shift type with the minimum shift length, then this nurse in the next day and all the remaining days of the planning horizon must be off. Therefore, all assignments combinations where this nurse is working on that day and following days are removed from the list of possible states.
4. If the last day of the partial solution is off, check the number of consecutive days-off. If the minimum allowed value is not reached yet, then the next day of this nurse must be off, and all assignments combinations where this nurse is working on that day are removed from the list of possible states.

5. If the next day is a weekend day, check the number of working weekends for the partial solution of the nurse. If the maximum allowed value is already reached, then the next day for this nurse must be off, and all assignments combinations where this nurse is working on that day are removed from the list of possible states.

Those five filtering criteria help to reduce the number of reachable states from the set of partial solutions that are kept from the previous day. Therefore, eliminating unnecessary computational time that may be wasted in evaluating many infeasible solutions. After that, the feasibility of the remaining reachable states – after applying the five filtering criteria – is evaluated using only the remaining three hard constraints (Unwanted shift patterns, Maximum number of shifts, and Minimum consecutive working days). However, the minimum total working time constraint is checked only on the last day of the planning horizon.

The contribution of using DP based heuristic to recreate the destroyed schedules is not only diversifying the VNS process but also improving the nurses' individual schedules themselves. After recreating the destroyed nurses' schedules, the solution can be passed again to the VNS algorithm for further possible improvements.

4. **Experimental results and analysis.**

The proposed VNS-DP algorithm was completely coded using MATLAB, and converted into MEX files aiming to improve the code performance. The MEX files were run on MATLAB R2018a. All experiments were implemented on an Intel Core-i7 – 3.6GHz PC with 8 GB RAM.

4.1. **2014 benchmark instances.**

The 24 benchmark instances introduced in [15] were used to evaluate the performance of the proposed approach. A termination condition of one hour runtime is used. Regarding the vertical swapping structures, a maximum allowed number of 10,000 swaps for each structure is set. While, for each horizontal swapping structure, a maximum number of 100 swaps for each nurse is allowed. Two individual nurses' schedules can be destroyed in the destroy-and-recreate perturbation mechanism. Regarding the DP based heuristic algorithm, a maximum number of 200 partial solutions are allowed to be kept from a day to a following day. All of these values are based on trial and error experimentation.

In order to demonstrate the effectiveness of each of the proposed perturbation mechanisms, a comparison between the VNS with the classical mechanism and the VNS with the destroy-and-recreate mechanism is presented in Table 2. The results shown in Table 2 represent the best solutions (objective function values) that could be achieved using each perturbation mechanism individually. The bold face numbers refer to the best results. As seen in the table, each mechanism outperformed the other on half of the tested instances. While the classical mechanism demonstrated better performance in most of the small and medium size instances, the destroy-and-recreate mechanism achieved remarkable better results for the large size instances (instance 20 till instance 24). Referring to Table 1, it can be noticed that those last five instances have significant larger numbers of shifts on/off requests which means a larger contribution in the solution penalty. Therefore, that good performance of the destroy-and-recreate mechanism in the large size instances can be attributed to the use of the DP based heuristic, which in addition to diversifying the search process, helped to improve the individual schedules of the nurses by reducing the violations for this large number of nurses' requests.

Thereupon, it was the idea of using the two perturbation mechanisms simultaneously in a single approach. So that to, benefit from the advantages of both

TABLE 2. Comparison for the performance of the VNS algorithm using classical and destroy-and-recreate perturbation mechanisms

Instance	Classical mechanism	Destroy-and-recreate mechanism
01	607	607
02	**828**	922
03	**1001**	1002
04	1721	**1717**
05	**1249**	1330
06	**2053**	2147
07	1077	**1073**
08	**1425**	1741
09	449	**446**
10	4679	**4320**
11	**3458**	3513
12	**4217**	4327
13	2423	**2396**
14	**1377**	1584
15	**4996**	5017
16	**3550**	3556
17	**6391**	6506
18	**5177**	5601
19	**4789**	5177
20	9097	**7689**
21	29486	**27071**
22	61690	**58226**
23	62195	**53694**
24	402313	**248016**

mechanisms to achieve a good performance for all different size problem instances. The use of the DP based heuristic within the destroy-and-recreate mechanism helps to improve the solution quality through improving the individual nurses' schedules. However, its ability to escape from the local optimal solutions is relatively limited due to the limited number of nurses that are disrupted at each iteration. While the classical perturbation mechanism has a relatively better capability to escape from local optimal as it disrupts the whole solution structure.

The results of the proposed VNS algorithm with two perturbation mechanisms are compared with state-of-the-art solution methods from literature for the purpose of performance evaluation. Table 3 compares the proposed VNS-DP approach with the ejection chain heuristic, branch and price algorithm, Gurobi 7.0 solver results, and a recent hybrid Constraint Programming and Iterated Local Search (CP-ILS) approach. The shown results of the compared ejection chain heuristic and the branch and price algorithm are the results reported by the authors in [15]. The Gurobi solver results are the recent results reported in [8]. The results of the CP-ILS approach was reported in [25]. The comparison shows that the proposed VNS-DP approach outperformed the ejection chain heuristic in 22 instances out of the 24 instances, produced the same result for the first instance, and failed to outperform it in the last instance only.

However, the proposed VNS-DP approach outperformed the Gurobi solver results in 7 instances and produced the same results for 4 instances only. Also, it outperformed the branch and price algorithm in 12 instances and produced the same results for 4 instances. It is worth notice that the VNS-DP approach was able to generate solutions for all the large size instances that the Gurobi solver and the branch and price algorithm could not solve during the same runtime of one hour. Comparing to the CP-ILS approach, the proposed VNS-DP approach was able to outperform it in 11 instances, achieve the same results for 4 instances, and produce very comparable results for most of the remaining instances.

Table 3. Comparison between the proposed VNS–DP approach and other methods in literature

Instance	VNS–DP	Ejection chain	Gurobi	Branch & price	CP–ILS
01	**607**	**607**	**607**	**607**	**607**
02	**828**	837	**828**	**828**	**828**
03	**1001**	1003	**1001**	**1001**	**1001**
04	**1716**	1718	**1716**	**1716**	**1716**
05	1237	1358	**1143**	1160	1147
06	2141	2258	**1950**	1952	2050
07	1080	1269	**1056**	1058	1084
08	1452	2260	**1306**	1308	1464
09	446	463	**439**	**439**	454
10	4656	4797	**4631**	**4631**	4667
11	3512	3661	**3443**	**3443**	3457
12	4119	5211	**4040**	4046	4308
13	**2120**	2663	3109	–	2961
14	1344	1847	**1278**	–	1432
15	4637	5935	4843	–	**4570**
16	3458	4048	**3225**	3323	3748
17	6190	7835	**5749**	–	6609
18	5095	6404	**4760**	–	5416
19	**4281**	5531	5078	–	4364
20	7274	9750	**3591**	–	6654
21	26263	36688	132445	–	**22549**
22	56091	516686	265504	–	**48382**
23	51699	54384	–	–	**38337**
24	226490	**156858**	–	–	177037

Another comparison with results presented in [27] for a hybrid Integer Programming and Variable Neighbourhood Search (IP-VNS) approach is shown in Table 4. The VNS-DP approach was able to produce better results than the IP-VNS in the largest size instance (instance 24) only and produce the same results also for the first 4 instances. Indeed, this humble performance of the VNS-DP approach compared to the IP-VNS can be traced back to the different methods integrated with the VNS in each approach. In the IP-VNS approach, an exact solution method of integer programming is integrated. This exact method is able to solve the subproblems to optimality which gives more probability to the overall approach to produce the optimal solutions at least for the small and medium size instances. While the VNS-DP approach used a DP based heuristic that could not prove the optimality of the produced nurses' schedules (subproblems). However, it can produce near optimal solutions.

The superiority of the IP-VNS over the VNS-DP in most of the instances can be justified to the ability of the IP solver of the IP-VNS approach to solve the generated subproblems in reasonable time. So, as long as the subproblems' sizes can be handled reasonably by the IP solver, the IP-VNS has a superior performance. However, once the subproblem size become larger than what an IP solver can tackle, the IP-VNS approach revealed a worse performance. This is what happened in the largest instance (instance 24). As the size of the main problem becomes huge with the large number of nurses, shift types, as well as the nurses requests, even the subproblems sizes still large to be solved exactly in reasonable time. Therefore, here, the advantage of our DP based heuristic is demonstrated. The DP based heuristic is able to handle even such large size subproblems that exact solution methods fail to solve in a reasonable time. However, the quality of the solutions produced by the VNS-DP approach for most of the large size instances is still worse than the IP-VNS results. Therefore, more investigations for possible improvements in the DP based heuristic have to be executed in order to improve the approach performance especially with the large size instances where the IP-VNS showed significantly better results.

4.2. **The First International Nurse rostering Competition.** For the sake of testing the ability of the proposed hybrid approach to tackle different versions of NRP,the approach was adapted to handle the well-known instances of the First International Nurse Rostering Competition [21]. Indeed, the competition instances are definitely different than the previously tested 24 benchmark instances in terms of the soft and hard constraints. In contrast to the 24 benchmark instances that comprised 10 hard constraints and two soft constraints, the competition instances implied only two hard constraints (single shift assignment per day for each nurse, and the cover requirements) and 17 soft constraints as follows.

1. Maximum number of assignments for each nurse during planning horizon
2. Minimum number of assignments for each nurse during planning horizon
3. Maximum number of consecutive working days
4. Minimum number of consecutive working days
5. Maximum number of consecutive free days
6. Minimum number of consecutive free days
7. Maximum number of consecutive working weekends
8. Minimum number of consecutive working weekends
9. Maximum number of working weekends in 4 weeks
10. Assignment of complete working weekends
11. Assignment of identical shift types in weekend
12. No night shift before free weekend
13. Two free days after night shifts
14. Alternative skill
15. Unwanted shift patterns
16. Day off/on requests

17. Shift off/on requests

The competition was composed of three tracks that vary in difficulty, problem size and allowed runtime. The first track is the *sprint* track that requires solution in 10 seconds. The second track is the *medium* track that requires solution within 10 minutes. However the third one is the *long* track that admitted a maximum runtime of 10 hours to get a solution.

In the adapted VNS-DP algorithm, in order to keep the solution feasibility during the search process, the HWCF and the horizontal swapping neighborhood structures are deactivated. As, according to the competition instances, the cover requirements are considered hard constraints. So, no violations for the cover requirements constraint can be occurred during the search process. In addition, the filtering criteria in the DP based heuristic were adapted to use only the two hard constraints in the competition instances.

Table 5 compares the best results obtained by the proposed hybrid VNS-DP approach with state-of-the-art approaches from literature. The results of the compared approaches are the best values obtained by the approaches and reported in literature by their authors. These compared approaches are an Adaptive Neighborhood Search (ANS) approach [23], a Variable Depth Search (VDS) approach [11], a Randomized Variable Neighborhood Search (RVNS) approach [36], and a Hybrid

Table 4. Comparison between the proposed VNS-DP approach and IP-VNS approach

Instance	VNS–DP	IP–VNS
01	**607**	**607**
02	**828**	**828**
03	**1001**	**1001**
04	**1716**	**1716**
05	1237	**1143**
06	2141	**1950**
07	1080	**1056**
08	1452	**1344**
09	446	**439**
10	4656	**4631**
11	3512	**3443**
12	4119	**4040**
13	2120	**1905**
14	1344	**1279**
15	4637	**3928**
16	3458	**3225**
17	6190	**5750**
18	5095	**4662**
19	4281	**3224**
20	7274	**4913**
21	26263	**23191**
22	56091	**32126**
23	51699	**3794**
24	**226490**	2281440

Harmony Search Algorithm (HHSA) [4]. As well as, the Best Known Solutions (BKS) reported in the competition website [37]. Note that the bold font numbers in table refer to the best result. The (–) means that the corresponding approach was not tested on that instance.

For the sprint track instances, the results show that the proposed VNS–DP approach was able to reach the best known solutions for 23 instances out of the 30 instances. However the VNS–DP approach achieved comparable results with the compared approaches in the remaining sprint instances.

In the medium track, the VNS–DP approach achieved the best known solutions for 9 instances out of 15 medium instances. The VNS–DP approach achieved comparable results with the results obtained by the other approaches for most of the remaining medium instances.

Finally, for the long tack instances, the proposed VNS–DP approach could reach the best known solutions for 12 out of the 15 long instances, and achieved comparable results with the BKS for the remaining three long instances.

Table 5: Comparison between results of the proposed VNS-DP approach and other state-of-the-art approaches tested on the competition instances

Instance	BKS	VNS–DP	ANS	VDS	RVNS	HHSA
Sprint_early01	**56**	**56**	**56**	**56**	**56**	**56**
Sprint_early02	**58**	**58**	**58**	**58**	**58**	**58**
Sprint_early03	**51**	**51**	**51**	**51**	**51**	**51**
Sprint_early04	**59**	**59**	**59**	**59**	**59**	**59**
Sprint_early05	**58**	**58**	**58**	**58**	**58**	**58**
Sprint_early06	**54**	**54**	**54**	**54**	**54**	**54**
Sprint_early07	**56**	**56**	**56**	**56**	**56**	**56**
Sprint_early08	**56**	**56**	**56**	**56**	**56**	**56**
Sprint_early09	**55**	**55**	**55**	**55**	**55**	**55**
Sprint_early10	**52**	**52**	**52**	**52**	**52**	**52**
Sprint_late01	**37**	**37**	**37**	**37**	**37**	**37**
Sprint_late02	**42**	**42**	**42**	**42**	**42**	**42**
Sprint_late03	**48**	**48**	**48**	**48**	**48**	**48**
Sprint_late04	**73**	**73**	**73**	75	**73**	**73**
Sprint_late05	**44**	45	**44**	**44**	**44**	**44**
Sprint_late06	**42**	43	**42**	**42**	**42**	**42**
Sprint_late07	**42**	47	**42**	**42**	43	43
Sprint_late08	**17**	**17**	**17**	**17**	**17**	**17**
Sprint_late09	**17**	**17**	**17**	**17**	**17**	**17**
Sprint_late10	**43**	44	**43**	**43**	**43**	**43**
Sprint_hidden01	**32**	34	**32**	–	**32**	**32**
Sprint_hidden02	**32**	**32**	**32**	–	**32**	**32**
Sprint_hidden03	**62**	**62**	**62**	–	**62**	**62**
Sprint_hidden04	**66**	**67**	**66**	–	**66**	**66**
Sprint_hidden05	**59**	**59**	**59**	–	**59**	**59**
Sprint_hidden06	**130**	141	**130**	–	–	**130**
Sprint_hidden07	**153**	**153**	**153**	–	–	**153**
Sprint_hidden08	**204**	**204**	**204**	–	–	**204**

Table 5 – continued from previous page

Instance	BKS	VNS–DP	ANS	VDS	RVNS	HHSA
Sprint_hidden09	**338**	**338**	**338**	–	–	**338**
Sprint_hidden10	**306**	**306**	**306**	–	–	**306**
Medium_early01	**240**	243	**240**	244	**240**	243
Medium_early02	**240**	241	**240**	241	**240**	242
Medium_early03	**236**	238	**236**	238	**236**	238
Medium_early04	**237**	240	**237**	240	**237**	240
Medium_early05	**303**	**303**	**303**	308	**303**	305
Medium_late01	**157**	**157**	164	187	158	169
Medium_late02	**18**	31	20	22	20	26
Medium_late03	**29**	**29**	30	46	30	34
Medium_late04	**35**	**35**	36	49	**35**	42
Medium_late05	**107**	140	117	161	111	131
Medium_hidden01	**111**	**111**	122	–	117	143
Medium_hidden02	**219**	**219**	224	–	225	248
Medium_hidden03	**34**	**34**	35	–	**34**	49
Medium_hidden04	**78**	**78**	80	–	79	87
Medium_hidden05	**118**	**118**	120	–	122	169
Long_early01	**197**	**197**	**197**	198	**197**	**197**
Long_early02	**219**	221	222	223	**219**	226
Long_early03	**240**	**240**	**240**	242	**240**	**240**
Long_early04	**303**	**303**	**303**	305	**303**	**303**
Long_early05	**284**	**284**	**284**	286	**284**	**284**
Long_late01	**235**	240	237	286	**235**	253
Long_late02	**229**	**229**	**229**	290	**229**	256
Long_late03	**220**	**220**	222	290	221	256
Long_late04	**221**	**221**	227	280	224	263
Long_late05	**83**	89	**83**	110	**83**	98
Long_hidden01	**346**	**346**	**346**	–	349	380
Long_hidden02	**89**	**89**	**89**	–	**89**	110
Long_hidden03	**38**	**38**	**38**	–	**38**	44
Long_hidden04	**22**	**22**	**22**	–	**22**	27
Long_hidden05	**41**	**41**	45	–	**41**	53

Moreover, for a clear evaluation of the performance of the proposed VNS–DP approach, a comparison for the percentage of best known solution that achieved by the VNS–DP approach in each track is presented in table 6 compared to ANS and HHSA approaches, where results for all the 60 instances were reported. The comparison shows that, although the VNS-DP approach showed worse performance than the other two approaches in the sprint track instances, it demonstrated an obviously better performance in both medium and long tracks.

TABLE 6. Comparison for number of best known solutions achieved by each approach

Competition track	VNS–DP	ANS	HHSA
Sprint (30 instances)	77%	100%	97%
Medium (15 instances)	60%	33%	0%
Long (15 instances)	80%	67%	27%

5. Conclusion.

In this paper, a new hybrid approach of Variable Neighbourhood Search and Dynamic Programming based heuristic was presented to handle two different benchmark sets of the Nurse Rostering problem. The approach started by generating an initial solution using a greedy constructive heuristic. Then a VNS algorithm is used to refine the solution locally. After that, two different perturbation mechanisms were adopted to diversify the search process as well as improve the individual nurses' schedules. The resulted solution is refined again using the VNS, and the process is repeated till a termination condition is met.

The effectiveness of adopting the two perturbation mechanisms simultaneously was demonstrated. The algorithm was tested on 24 benchmark instances from literature as well as the instances of the first international nurse rostering competition. The results showed a competitive performance of the proposed approach compared to multiple state-of-the-art approaches from literature. However, it showed a humble performance when compared to a recent hybrid integer programming and variable neighborhood search approach.

Future work includes investigating possible ways to improve the current dynamic programming based heuristic of the hybrid approach, so it maybe able to compete the hybrid IP-VNS approach. Another direction to be investigated, is studying the feasibility of incorporating another destroy-and-recreate vertical mechanism for individual days rather than individual nurses.

Acknowledgments. This research project is sponsored by the Egyptian ministry of higher education grant and the Japanese International Cooperation Agency (JICA) in the scope of the Egypt-Japan University of Science and Technology.

REFERENCES

[1] M. Abdelgalil, Z. Yahia and A. B. Eltawil, A proposed new dynamic programming formulation for the nurse rostering problem, in *Proceedings of 47th International Conference on Computers and Industrial Engineering*, 2017, 1–8.

[2] H. Akita and A. Ikegami, A network representation of feasible solution space for a subproblem of nurse scheduling, *Proceedings of the Institute of Statistical Mathematics*, **61** (2013), 79–95.

[3] G. Alpan, R. Larbi and B. Penz, A bounded dynamic programming approach to schedule operations in a cross docking platform, *Computers and Industrial Engineering*, **60** (2011), 385–396.

[4] M. A. Awadallah, M. A. Al-Betar, A. T. Khader, A. L. Bolaji and M. Alkoffash, Hybridization of harmony search with hill climbing for highly constrained nurse rostering problem, *Neural Computing and Applications*, **28** (2017), 463–482.

[5] J. Bautista and J. Pereira, A dynamic programming based heuristic for the assembly line balancing problem, *European Journal of Operational Research*, **194** (2009), 787–794.

[6] F. Bellanti, G. Carello, F. Della Croce and R. Tadei, A greedy-based neighborhood search approach to a nurse rostering problem, *European Journal of Operational Research*, **153** (2004), 28–40.

[7] S. Bouajaja and N. Dridi, A survey on human resource allocation problem and its applications, *Operational Research*, **17** (2017), 339–369.

[8] J. D. Bunton, A. T. Ernst and M. Krishnamoorthy, An integer programming based ant colony optimisation method for nurse rostering, in *Proceedings of the 2017 Federated Conference on Computer Science and Information Systems*, **11** (2017), 407–414.

[9] E. Burke, P. De Causmaecker, S. Petrovic and G. V. Berghe, Variable neighborhood search for nurse rostering problems, in *Metaheuristics: Computer Decision-Making*, Springer, Boston, MA, 2003, 153–172.

[10] 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*, **188** (2008), 330–341.

[11] E. K. Burke and T. Curtois, New approaches to nurse rostering benchmark instances, *European Journal of Operational Research*, **237** (2014), 71–81.

[12] E. K. 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*, **203** (2010), 484–493.

[13] E. K. Burke, T. Curtois, L. F. van Draat, J. Van Ommeren and G. Post, Progress control in iterated local search for nurse rostering, *Journal of the Operational Research Society*, **62** (2011), 360–367.

[14] S. Ceschia, N. Dang, P. De Causmaecker, S. Haspeslagh and A. Schaerf, Second international nurse rostering competition, *Annals of Operations Research*, **274** (2019), 171–186, `arXiv:1501.04177`.

[15] T. Curtois and R. Qu, *Technical Report: Computational Results on New Staff Scheduling Benchmark Instances*, ASAP Research Group, School of Computer Science, University of Nottingham, UK, 2014.

[16] F. Della Croce and F. Salassa, A variable neighborhood search based matheuristic for nurse rostering problems, *Annals of Operations Research*, **218** (2014), 185–199.

[17] J. Van den Bergh, J. Beliën, P. De Bruecker, E. Demeulemeester and L. De Boeck, Personnel scheduling: A literature review, *European Journal of Operational Research*, **226** (2013), 367–385.

[18] S. C. Gao and C. W. Lin, Particle swarm optimization based nurses' shift scheduling, in *Proceedings of the Institute of Industrial Engineers Asian Conference 2013*, 2013, 775–782.

[19] R. A. M. Gomes, T. A. M. Toffolo and H. G. Santos, Variable neighborhood search accelerated column generation for the nurse rostering problem, *Electronic Notes in Discrete Mathematics*, **58** (2017), 31–38.

[20] M. Hadwan, M. Ayob, N. R. Sabar and R. Qu, A harmony search algorithm for nurse rostering problems, *Information Sciences*, **233** (2013), 126–140.

[21] S. Haspeslagh, P. De Causmaecker, A. Schaerf and M. Stølevik, The first international nurse rostering competition 2010, *Annals of Operations Research*, **218** (2014), 221–236.

[22] F. Knust and L. Xie, Simulated annealing approach to nurse rostering benchmark and real-world instances, *Annals of Operations Research*, **272** (2019), 187–216.

[23] Z. Lü and J.-K. Hao, Adaptive neighborhood search for nurse rostering, *European Journal of Operational Research*, **218** (2012), 865–876.

[24] R. M'Hallah and A. Alkhabbaz, Scheduling of nurses: A case study of a Kuwaiti health care unit, *Operations Research for Health Care*, **2** (2013), 1–19.

[25] N. Musliu and F. Winter, A hybrid approach for the sudoku problem: Using constraint programming in iterated local search, *IEEE Intelligent Systems*, **32** (2017), 52–62.

[26] F. Mischek and N. Musliu, Integer programming model extensions for a multi-stage nurse rostering problem, *Annals of Operations Research*, **275** (2019), 123–143.

[27] E. Rahimian, K. Akartunali and J. Levine, A hybrid integer programming and variable neighbourhood search algorithm to solve nurse rostering problems, *European Journal of Operational Research*, **258** (2017), 411–423.

[28] E. Rahimian, K. Akartunali and J. Levine, A hybrid integer and constraint programming approach to solve nurse rostering problems, *Computers and Operations Research*, **82** (2017), 83–94.

[29] H. G. Santos, T. A. M. Toffolo, R. A. M. Gomes and S. Ribas, Integer programming techniques for the nurse rostering problem, *Annals of Operations Research*, **239** (2016), 225–251.

[30] A. M. Shahidin, M. S. M. Said, N. H. M. Said and N. I. A. Sazali, Developing optimal nurses work schedule using integer programming, *AIP Conference Proceedings*, **1870** (2017), 040031.

[31] P. Smet, P. Brucker, P. D. Causmaecker and G. V. Berghe, Polynomially solvable personnel rostering problems, *European Journal of Operational Research*, **249** (2016), 67–75.

[32] I. P. Solos, I. X. Tassopoulos and G. N. Beligiannis, A generic two-phase stochastic variable neighborhood approach for effectively solving the nurse rostering problem, *Algorithms*, **6** (2013), 278–308.

[33] I. X. Tassopoulos, I. P. Solos and G. N. Beligiannis, A two-phase adaptive variable neighborhood approach for nurse rostering, *Computers and Operations Research*, **60** (2015), 150–169.

[34] T.-H. Wu, J.-Y. Yeh and Y.-M. Lee, A particle swarm optimization approach with refinement procedure for nurse rostering problem, *Computers and Operations Research*, **54** (2015), 52–63.

[35] Z. Zheng and X. Gong, Chemical reaction optimization for nurse rostering problem, in *Frontier and Future Development of Information Technology in Medicine and Education*, Springer, Dordrecht, 2014, 3275–3279.

[36] Z. Zheng, X. Liu and X. Gong, A simple randomized variable neighbourhood search for nurse rostering, *Computers and Industrial Engineering*, **110** (2017), 165–174.

[37] Instances & Results – Nurse Rostering Competition, (Accessed: April 25, 2019), https://www.kuleuven-kulak.be/nrpcompetition/instances-results.

Received December 2018; revised December 2019.

E-mail address: mohammed.abdelgalil@ejust.edu.eg
E-mail address: eltawil@ejust.edu.eg
E-mail address: zakaria.yahia@fayoum.edu.eg
E-mail address: nakata.k.ac@m.titech.ac.jp