A Variable Neighbourhood Search for Nurse Scheduling with Balanced Preference Satisfaction

Ademir Aparecido Constantino[1], Everton Tozzo[1], Rodrigo Lankaites Pinheiro[2], Dario Landa-Silva[2] and Wesley Romão[1]

[1]*Informatics Department, State University of Maringá, Maringá, PR, Brazil*

[2]*ASAP Research Group, School of Computer Science, University of Nottingham, Nottingham, U.K.*

Keywords: Nurse Scheduling Problem, Bottleneck Assignment Problem, Variable Neighbourhood Search, Combinatorial Optimisation.

Abstract: The nurse scheduling problem (NSP) is a combinatorial optimisation problem widely tackled in the literature. Recently, a new variant of this problem was proposed, called nurse scheduling problem with balanced preference satisfaction (NSPBPS). This paper further investigates this variant of the NSP as we propose a new algorithm to solve the problem and obtain a better balance of overall preference satisfaction. Initiall, the algorithm converts the problem to a bottleneck assignment problem and solves it to generate an initial feasible solution for the NSPBPS. Posteriorly, the algorithm applies the Variable Neighbourhood Search (VNS) metaheuristic using two sets of search neighbourhoods in order to improve the initial solution. We empirically assess the performance of the algorithm using the NSPLib benchmark instances and we compare our results to other results found in the literature. The proposed VNS algorithm exhibits good performance by achieving solutions that are fairer (in terms of preference satisfaction) for the majority of the scenarios.

1 INTRODUCTION

In this paper we consider a variant of the well-known Nurse Scheduling Problem (NSP) referred as the Nurse Scheduling Problem with Balanced Preference Satisfaction (NSPBPS). The NSP is a NP-Hard combinatorial optimisation problem (Osogami and Imai, 2000) that consists of: given a period of time, a set of staff requirements for the given period and a team of nurses, the aim is to assign shift patterns for the team of nurses in order to satisfy the staff requirements. On the basic version of the NSP, the objective is to maximise the number of requirements fulfilled. The problem has several variants where the objectives include the minimisation of the number of nurses used, the minimisation of the overall assignment costs (given that the cost of each nurse assignment can be calculated), the satisfaction of nurses preferences, and more (Petrovic and Vanden Berghe, 2008).

The NSPBPS was proposed by Constantino et al. (2011) and it is a variant of the NSP where each nurse can have preferences regarding her/his shift assignments. In the NSP, maximising the overall number of preferences attended may lead to some nurses being largely benefited while other nurses are neglected. The purpose of fulfilling nurses' preferences is to improve work conditions and raise overall staff satisfaction. However, by providing individual timetables that are unbalanced in terms of individual preferences satisfaction, provokes that some nurses feel disfavoured when the level of satisfaction of their individual preferences is lower than for other nurses. Therefore, the objective in the NSPBPS is to fulfill staff requirements but attending individual nurse preferences in a balanced manner instead of just maximising the overall number of preferences. In the NSPBPS we aim to obtain a fair assignment by minimising the difference between the preference satisfaction of the most favoured nurse and the least favoured nurse.

Surveys of the NSP were provided by Cheang et al. (2003) and Burke et al. (2004), where problem variants, features and solving methods were covered and discussed. Both works observed that at the time there were no sets of benchmark instances to cross-evaluate solving methodologies. Maenhout and Vanhoucke (2005) presented a comprehensive dataset of varied NSP instances called NSPLib. Later, the NSPLib was further improved with the addition

Aparecido Constantino A., Tozzo E., Lankaites Pinheiro R., Landa-Silva D. and Romão W..
A Variable Neighbourhood Search for Nurse Scheduling with Balanced Preference Satisfaction.
DOI: 10.5220/0005364404620470
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 462-470
ISBN: 978-989-758-096-3

of new features and updated results for comparison (Maenhout and Vanhoucke, 2006, 2007, 2008).

Several heuristic techniques have been applied to solve the NSP and its variants. Examples include the recent works of Maenhout and Vanhoucke (2007) which uses the Electromagnetic Method, Maenhout and Vanhoucke (2006) using Scatter Search, several applications of Evolutionary Algorithms (Maenhout and Vanhoucke, 2008; Ohki et al., 2008; Tsai and Li, 2009; Beddoe et al., 2009), Oughalime et al. (2008) using Tabu Search, Burke et al. (2008) using Variable Neighbourhood Search, Goodman et al. (2009) which uses GRASP and Cheng et al. (2007) using Simmulated Annealing. Additionally, exact methods were also employed to tackle the NSP. We can highlight the works of Yilmaz (2012) using Integer Programming and Bard and Purnomo (2005) that employs Column Generation. Constantino et al. (2013) used a hybrid algorithm for the NSP with preference satisfaction.

To the best of our knowledge, only Constantino et al. (2011) presented results for the NSPBPS. In that work they proposed a hybrid algorithm that uses a bottleneck assignment problem exact algorithm to construct an initial solution and a local search to improve that solution. The present work further extends that concept by using the same mechanism to generate the initial solution. However, instead of a simple local search, we propose the use of a VNS metaheuristic with a new set of neighbourhoods. Experimental results on the NSPLib showed that we managed to improve the balance of preference satisfaction.

Section 2 describes the NSPBPS. In section 3, the bottleneck assignment problem used to generate the initial solution, is described. Section 4 presents the proposed VNS algorithm with the new set of neighbourhoods. Section 5 presents the experimental results and a discussion on the algorithm performance. Finally we conclude our work in Section 6.

2 THE NURSE SCHEDULING PROBLEM WITH BALANCED PREFERENCE SATISFACTION

In this work we consider the NSP as presented by Maenhout and Vanhoucke (2007). Nevertheless, in the NSPBPS the objective is not to minimise the number of constraints violations but instead to optimise the balance of attended nurses' preferences. The problem is composed by a number of hard and soft constraints to be considered when generating each individual schedule. The hard constraints define strict work regulations and the soft constraints define working policies and desirable aspects.

- *Hard Constraints.* The NSPBPS considers two hard constraints, both related to avoid certain shift patterns. The first forbids the assignment of an afternoon shift before a morning shift. The second forbids the assignment of a night shift before a morning or an afternoon shift.
- *Soft Constraints.* There are four soft constraints that represents nurses' preferences. The first one defines a minimum and maximum number of working days within the scheduling period. The second one defines a minimum and maximum number of consecutive working assignments. The third one represents a minimum and maximum number of assignments of each shift type in the scheduling period. The fourth soft constraint sets a minimum and maximum number of consecutive shift assignments of the same type.
- *Nurses Preferences.* The preferences represent the main objective of the NSPBPS. They are declared in advance and then a cost (inversely proportional to the preference) is associated to each shift.

We can define the NSPBPS as follows. Given a set N of nurses, where $N = \{n_1, n_2, \ldots, n_{max}\}$, a set D of days where $D = \{d_1, d_2, \ldots, d_{max}\}$ and a set S of shifts where $S = \{s_1, s_2, \ldots, s_{max}\}$. For each day $d_i \in D$, each nurse $n_j \in N$ has to be assigned a shift $s_k \in S$ such that the hard and soft constraints are considered and the differences of the overall individual costs for the nurses preferences are minimised.

In this work we use the tests scenarios provided by the NSPLib, a library with problem instances for the NSP. For more information, please refer to the work of Maenhout and Vanhoucke (2005). The NSPLib uses shifts that refers to a given working period (early, day or night shift) or a rest period (free shift). A duty roster, or roster, is a sequence of $|D|$ shifts assigned to a nurse. We consider S_d to be the set of shifts required for day d.

Additionally, let pf_{n_j} be a sum of the costs (preferences non-satisfaction) for nurse n_j regarding his/her individual roster. Thus, if we wish to optimise the overall nurse satisfaction, we must build and assign rosters to nurses in such way that the sum of pf_{n_j} for all $n_j \in N$ is minimised. However, to aim for rosters with balanced preference satisfaction, one approach is to minimise the maximum pf_{n_j} for all $n_j \in N$. This objective is used in the well-known bottleneck assignment problem as discussed next.

3 THE BOTTLENECK ASSIGNMENT PROBLEM

The NSP considered here involves the construction of the duty roster for a number of nurses in a scheduling period of D days. For each nurse, a shift must be assigned for each day in the scheduling period. In this paper, we model the NSP as an acyclic multipartite graph with $|D|+1$ partitions, where the vertices in the first partition corresponds to nurses and the vertices in the remaining $|D|$ partitions correspond to shifts, that is, one partition per day d_i. An edge represents the possibility of assigning a shift to a nurse or shift (depends on the algorithm phase).

Formally, lets have a graph $G = (V,E)$, where V is the set of vertices and E is the set of edges. The set V is composed by the subsets (partitions) of vertices $V_0, V_1, V_2, ..., V_{d_{max}}$, where V_0 is the set of vertices representing the nurses and V_d (d from d_1 to d_{max}) is the set of vertices representing day d of the scheduling period.

Figure 1 presents a diagram of the problem representation for two nurses in a seven day period. The two leftmost (rectangular) vertices, first on each partition, are the vertices representing the nurses, respectively nurses one and two. The vertices to the right correspond to a day of the schedule and can contain one of the following shifts: (E)arly, (D)ay, (N)ight, and (F)ree shift. Additionally, the * symbol defines a spare shift.

The optimisation problem is to find n_{max} paths going from the first to the last partition while minimising the total cost (each path corresponds to an individual roster). For this, we propose a heuristic algorithm that solves successive bottleneck assignment problems each corresponding to two consecutive partitions. The goal in the bottleneck assignment problem (BAP) is to make a one-to-one assignment between two sets of the same size in such way that the maximum assignment costs is minimised. Then, solving each successive BAP in our approach corresponds to minimising the maximum cost due to the non-satisfaction of individual preferences. The bottleneck assignment problem is formulated as follows:

$$\begin{aligned} \text{minimise} \quad & \max_{1\le i,j\le n} c_{ij}x_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1,\ldots,n \\ & \sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,\ldots,n \\ & x_{ij} \in \{0,1\}, \qquad i,j = 1,\ldots,n \end{aligned}$$

In this case, index i sometimes corresponds to a nurse and other times to a shift while index j can be a shift or a roster (more details are given below). The term c_{ij} corresponds to the cost of assigning i to j. The main advantage of tackling the NSP in this way is that the bottleneck assignment problem can be solved in polynomial time using the algorithm proposed by Carpaneto and Toth (1981).

4 THE PROPOSED VNS ALGORITHM

This work presents a two-phases algorithm to tackle the NSPBPS. First, an initial solution is obtained by solving a bottleneck assignment problem for each day of the rostering period using the algorithm proposed by Carpaneto and Toth (1981). Then, a general VNS metaheuristic (Mladenović and Hansen, 1997) is applied in order to improve this initial solution by exploring two domain-specific neighbourhoods, Cutting and Recombination procedure and k-Swap.

4.1 Initial Solution

We now briefly present the construction algorithm for the initial solution. Note that we use the same technique as proposed by Constantino et al. (2011). For further information, please refer to that work.

The construction phase starts by generating a multipartite graph as defined in Section 3. An initial solution is obtained through the resolution of $|D|$ successive BAPs from the first to the last day. Each BAP is defined by the square cost matrix $[c^d_{ij}]$ of order n_{max}, where c^d_{ij} is the cost of assigning shift j to nurse i on

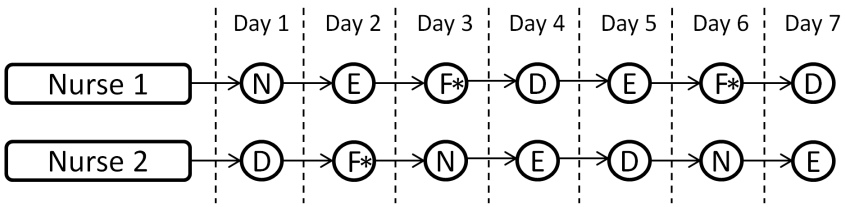


Figure 1: An initial solution for a problem containing two nurses. The letters E, D, N, F mean Early, Day, Night and Free shift, respectively, and * means spare shift.

day d. This cost is defined by the following function:

$$f(i,j,d) = pc(i,j,d) + P_h \times nHCV + P_s \times nSVC$$

where $pc(i,j,d)$ is the preference cost associated to assigning a nurse i to shift j on day d; $nHCV$ is the number of hard constraint violations; P_h is the penalty due to the violation of a hard constraint; $nSCV$ is the number of soft constraint violations and P_s is the penalty cost due to the violation of a soft constraint.

In this problem the number of nurses is always greater than or equal to the number of demanded tasks, i.e. $n_{max} \geq |S_d|$, then it might be necessary to complete the cost matrix with spare shifts in order to form a square matrix. Hence, matrix $[c_{ij}^d]$ may be divided into two blocks. Block I contains the shifts that satisfy the required coverage of that day. Block II contains the added spare shifts (in the case that in day d the number of nurses is greater than the number of demanded tasks). Since the minimum coverage requirement is guaranteed by block I, then any shift assignment in block II (spare shifts) is permitted, including the assignment of a free shift (with no given coverage requirement).

One BAP is constructed and solved for each day d of the scheduling period. Note that in the first assignment there is no shift previously assigned to each nurse. But from the second partition, previously assigned shifts are considered in order to calculate the cost matrix. At the end, after solving the $|D|$ BAPs, we have constructed an initial solution (duty roster) for the set N of nurses. The construction phase works as follows:

```
Procedure Construction
begin
   for d ← 1 to d_max do:
      construct the cost matrix for day d;
      solve the BAP for the cost matrix;
      assign shifts to nurses according to
         the BAP solution;
   end-for
end Construction
```

4.2 Solution Improvement: Variable Neighbourhood Search

The proposed algorithm implements a general Variable Neighbourhood Search (VNS) as proposed by Mladenović and Hansen (1997). The VNS starts from an initial solution and randomly picks a neighbour in order to disturb the current solution and avoid local optima. Then, it systematically inspects the neighbourhoods around that solution in pursuance of improving the current solution using a Variable Neighbourhood Descent (VND) algorithm. The VND is a simpler version of the VNS that does not possess the stochastic factor, hence it cycles through a set of neighbourhoods, accepting improvements, until the solution cannot be improved.

Below we present the pseudocode of the general VNS:

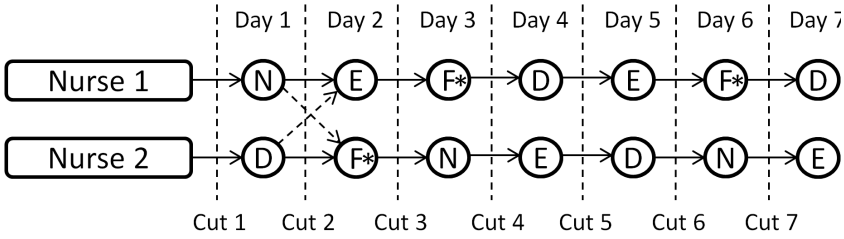
```
Procedure VNS
begin
   normalize the input data.
   s ← initial solution;
   while stop criteria not satisfied do:
      for i ← 1 to k do:
         s' ← choose random neighbour of
            s in N_k;
         s'' ← VND(s');
         if f(s'') < f(s) then:
            s ← s'';
            i ← 0;
         else
            i ← i+1;
         end-if
      end-for
   end-while
   return s;
end VNS
```

The chosen neighbourhoods plays a major role in the performance of the algorithm. Hence, we tailored two types of neighbourhoods in order to make better use of the data structures selected and improve the convergence of the algorithm to good solutions. We now describe the proposed neighbourhoods.

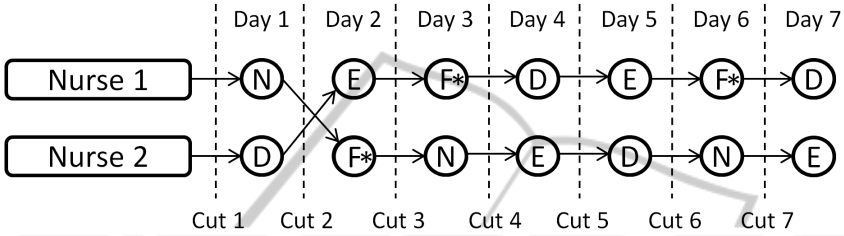
4.2.1 Neighbourhood 1: Cutting and Recombination Procedure (CRP)

The first neighbourhood, Cutting and Recombination Procedure, is the same neighbourhood employed in the local search technique proposed by Constantino et al. (2011). The procedure takes a current solution and performs successive 'cuts' between each pair of consecutive days. For each cut, it recombines all rosters and uses the BAP to search for the best feasible solution.

For every $d_{cut} \in (d_1, d_2, \ldots, d_{max} - 1)$, the procedure virtually splits the duty roster in two sides, the left, containing the roster for the days $(d_1, d_2, \ldots, d_{cut})$ and the right, containing the roster for the remaining days $(d_{cut} + 1, \ldots, d_{max})$. Now, each duty roster on the left side can be recombined with $|N|$ rosters on the right side (including the original combination), resulting in a total of $|N|^2$ possible recombinations. The procedure then creates a new cost matrix $[c_{ij}^{d_{cut}}]$, with size $|N| \times |N|$. Each value $c_{ij}^{d_{cut}}$ of the cost matrix is



(a) Possible recombinations for the CRP on the second cut point.



(b) New timetable after choosing a better combination.

Figure 2: Example of CRP application for two nurses.

calculated as the cost of assigning roster j to the nurse i from the day d_{cut} onwards. After the entire matrix is calculated, a BAP is solved using $[c_{ij}^{d_{cut}}]$ and the results are used to recombine the current solution.

The pseudocode for the CRP procedure is shown below:

```
Procedure CRP
begin
   for d_cut ← 1 to d_max − 1 do:
      construct the cost matrix for day
         d_cut;
      solve the BAP for the cost matrix;
      recombine the rosters according to
         the BAP solution;
   end-for
end CRP
```

Figure 2 presents two diagrams with an example of the application of the CRP procedure. The example considers the cut between days one and two for a two nurses scenario. In Figure 2(a) we have the two possible recombinations – the current assignment with solid lines and the other possibility with dashed lines. Then, in Figure 2(b), assuming that the previous dashed lines assignment poses a better solution, we have the new solution after effecting the changes.

4.2.2 Neighbourhoods $2 \ldots k$: k-Swap

The following neighbourhoods are obtained through a new procedure proposed in this work: k-Swap. The k-Swap is inspired in the original CRP as it is also based on cutting points and recombination. However, the k-swap seeks to cut entire blocks of size k and recombine these blocks, thus, each neighbourhood is composed by the allowed recombinations of blocks of size k.

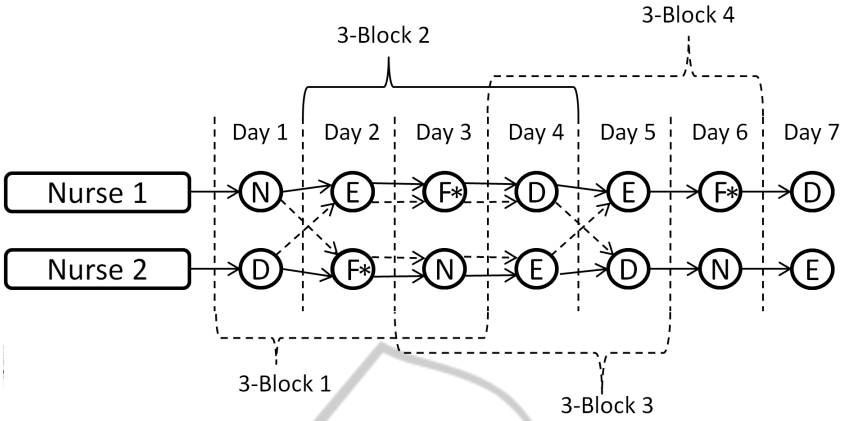
The procedure works as follows: for every $d_{cut} \in (d_1, d_2, \ldots, d_{max} - k - 1)$, the procedure virtually splits the duty roster in three parts, namely:

- **left side**: the roster for the days $(d_1, d_2, \ldots, d_{cut})$;
- k-**block**: the roster for the days $(d_{cut} + 1, \ldots, d_{cut} + k)$; and
- **right side** the roster for the days $(d_{cut} + k + 1, \ldots, d_{max})$.

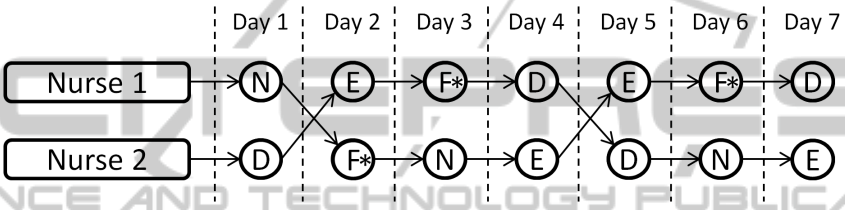
Now, instead of recalculating all possible recombinations between the block and the sides, which would generate $|N|^4$ possible recombinations (and arguably be a slow procedure), the k-Swap recombines the block such that each recombination on the left side is symmetric to the right side, hence for every left side recombination, there is only one possible outcome on the right side, generating a total of $|N|^2$ possible recombinations. The procedure also creates a new cost matrix $[c_{ij}^{d_{cut}}]$, with size $|N| \times |N|$, where each value $c_{ij}^{d_{cut}}$ is the cost of assigning the nurse i (along with the left roster i) to the k-block roster j and the k-block roster j to the right roster i. After $[c_{ij}^{d_{cut}}]$ is obtained, the BAP is solved and the results are used to recombine the current solution.

The pseudocode for the k-Swap procedure is shown below:

```
Procedure k-Swap
begin
   for d_cut ← 1 to d_max − k − 1 do:
```



(a) Possible recombinations for the *k*-Swap on the second block.



(b) Improved timetable after applying the *k*-Swap on the second block.

Figure 3: Example of *k*-Swap application for two nurses using $k = 3$.

```
      construct the cost matrix for the
         block d_cut ... d_cut + k;
      solve the BAP for the cost matrix;
      recombine the rosters according to
         the BAP solution;
   end-for
end k-Swap
```

Figure 3 presents a sample application of the *k*-Swap procedure for $k = 3$ to a scenario with two nurses and seven days. In Figure 3(a) we see the delimitation of each of the possible four blocks. The *3-Block 2*, presented in solid lines, is the one chosen for the current step of the procedure. Still, in Figure 3(a) we can observe all possible recombinations of rosters considering the current block, in this case, two recombinations. The first one, presented in solid lines is the current assignment while the second, presented in dashed lines is the alternative. Figure 3(b) shows the resulting roster assuming that the dashed-lined alternative from the previous figure gives better results.

5 EXPERIMENTAL RESULTS AND DISCUSSION

In order to evaluate the proposed algorithm, we used the same instances evaluated by Constantino et al. (2011). These instances were obtained from the NSPLib (Maenhout and Vanhoucke, 2005), a digital repository for instances of the NSP.

There are two types of problem instances available in the NSPLib. The first is the set of instances that contains weekly schedules. The number of nurses varies from 25 to 100 with an increment of 25 nurses. For each set of nurses there are 7290 files with different daily requirements and constraint values, hence totalling 29160 distinct scenarios. Additionally there are 8 separated files containing coverage constraints and preferences information that can be combined to any scenario, hence totalling 233280 problem instances that are available in the library. The second set of instances considers monthly schedules (28 days) but restricts the number of available nurses to 30 and 60. There are 960 files for each problem size and an additional 8 files containing preferences and coverage constraints, totalling 15360 possible scenarios. Therefore, when considering all the possible scenarios, we have a total of 248640 instances.

Due to time constraints we performed the tests on weekly schedules only, however we remind the reader that they constitute the majority of the scenarios in the NSPLib. We used all options of available nurses (25, 50, 75 and 100). In the objective evaluation function of our proposed algorithm, we considered the penalty for violating a hard constraint to be $P_h = 1000$ and the

Table 1: Summary of the results.

n	$\|D\|$	Case	# of Instances	Constantino's Algorithm			Proposed Algorithm		
				Best	Worst	Range	Best	Worst	Range
25	7	1	7290	12864	19099	6235	16051	19669	**3619**
25	7	2	7290	11774	18657	6884	16041	19637	**3596**
25	7	3	7290	13984	19415	5431	15406	19715	**4309**
25	7	4	7290	12344	19031	6688	16054	19682	**3627**
25	7	5	7290	13585	19492	5907	15765	19789	**4024**
25	7	6	7290	11812	18690	6877	16040	19650	**3610**
25	7	7	7290	14560	20102	**5541**	14219	20066	5847
25	7	8	7290	13030	19180	6149	15129	19834	**4705**
50	7	1	7290	13459	19842	6383	15719	19648	**3929**
50	7	2	7290	12480	19489	7009	15727	19584	**3857**
50	7	3	7290	14412	20147	5735	15143	19798	**4655**
50	7	4	7290	13034	19788	6754	15732	19644	**3912**
50	7	5	7290	14122	20213	6091	15364	19858	**4493**
50	7	6	7290	12541	19515	6974	15713	19603	**3890**
50	7	7	7290	14923	20787	**5864**	14083	20225	6142
50	7	8	7290	13539	19943	6404	14868	20004	**5135**
75	7	1	7290	13833	19530	5697	15464	19522	**4058**
75	7	2	7290	13122	19164	6042	15455	19493	**4038**
75	7	3	7290	14650	19927	5276	14901	19618	**4716**
75	7	4	7290	13398	19359	5961	15457	19526	**4069**
75	7	5	7290	14415	19944	5529	15156	19688	**4532**
75	7	6	7290	13148	19179	6031	15452	19502	**4050**
75	7	7	7290	15163	20642	**5479**	13952	19955	6002
75	7	8	7290	13955	19735	5780	14669	19643	**4974**
100	7	1	7290	11464	19425	7961	15026	20509	**5483**
100	7	2	7290	10545	19067	8522	15001	20468	**5467**
100	7	3	7290	12749	19862	7113	14660	20855	**6195**
100	7	4	7290	10966	19287	8321	15032	20524	**5492**
100	7	5	7290	12179	19831	7653	14770	20771	**6001**
100	7	6	7290	10589	19081	8491	15015	20478	**5463**
100	7	7	7290	13229	20529	**7301**	13386	21501	8115
100	7	8	7290	11699	19587	7888	14188	21051	**6863**

penalty for violating a soft constraint to be $P_s = 100$.

Table 1 summarises the results. The column n indicates the number of nurses considered. The column $|D|$ represents the number of days. The column *Case* indicate the preferences file used. The following column indicates the number of instances evaluated. The column *Best* indicates the average cost of the route of the most favoured nurse in the duty roster, i.e. the nurse whose evaluation cost is lower. Analogously, the column *Worst* represent the average cost of the route for the least favoured nurse, thus, the nurse whose evaluation cost is higher. The column *Range* presents the difference between the best and the worst columns values. A high value in the column *Range* means that there exists a higher discrepancy between the preferences considered for the nurses, hence some have many preferences considered in detriment of attending other nurses. Therefore, a low value in that column means a fairer rostering regarding the nurses preferences.

The results show that for almost all test scenarios, the proposed VNS algorithm provided fairer solutions when comparing the route costs of the most favoured nurse against the least favoured nurse. An interesting finding is that the reduction in the gap between the routes with highest and lowest costs, causes an increase in the average costs for the most favoured nurses, and sometimes also it increases the cost of the roster for the least favoured nurses. The proposed algorithm minimises the gap of nurses preferences by reducing the overall satisfaction of the nurses, hence raising the overall assignment costs. The reason is that the normalisation is applied over the preferences of all nurses. This process, aside from maintaining a fixed priority for all work shifts, alters the costs of the original preferences, approximating the value to 3. Therefore, when the normalisation is applied, the algorithm bias towards a solution that is close to a central preference value, and does not necessarily minimises the costs.

Table 2: Number of Violations.

n	$\|D\|$	Case	# of Instances	Constantino's Algorithm		Proposed Algorithm			
				Range	Avg. # of Violations	Range	Avg. # of Hard Viol.	Avg. # of Soft Viol.	Avg. # of Violations
50	7	1	7290	6383	**1031**	3929	987	102	1090
50	7	2	7290	7009	**937**	3857	983	9	992
50	7	3	7290	5735	**2364**	4655	949	2216	3165
50	7	4	7290	6754	**1053**	3912	978	135	1112
50	7	5	7290	6091	2744	4493	1406	1188	**2594**
50	7	6	7290	6974	**955**	3890	985	60	1045
50	7	7	7290	5864	**6091**	6142	1510	6824	8333
50	7	8	7290	6404	**2890**	5135	1555	3106	4661

Table 3: Comparison – average increase of violations $\times$ average decrease of range.

n	$\|D\|$	Case	# of Instances	Increase on the number of Evaluations	Decrease on the Range
50	7	1	7290	5.72%	**38.45%**
50	7	2	7290	5.87%	**44.97%**
50	7	3	7290	33.88%	18.83%
50	7	4	7290	5.60%	**42.08%**
50	7	5	7290	-5.47%	**26.24%**
50	7	6	7290	9.42%	**44.22%**
50	7	7	7290	36.81%	-4.74%
50	7	8	7290	61.28%	19.82%

Table 2 presents the average number of constraints violations for 50 nurses over 7 days. In order to simplify the visualisation, we chose only one set of instances, however it is important to emphasise that all other instances presented similar results. We compare our results with the results obtained by Constantino et al. (2011). As it can be seen, the proposed algorithm violated more constraints than the compared algorithm, except for preferences case 5. Nonetheless, when comparing the increase of the number of violations against the decrease of the range (Table 3), we can verify that, for the majority of the test cases, the decrease of the range highly surpasses the increase of violations. Notably, in case 7 the proposed algorithm was not able to improve the range, while on case 5 it was able to improve the range and the number of violations.

It is interesting to observe that the proposed algorithm could provide better results regarding the fairness of the routes but was unable to attend a same ammount of overall preferences. This gives some indication that there might be a trade-off in the sense that increasing fairness may necessarily decrease the overall preferences satisfaction.

Therefore, for the majority of the test instances, the proposed algorithm was able to provide fairer results regarding nurses preferences, hence providing better results. In general, there was an increase in the number of constraint violations, but for most cases the fairness of the comparison surpassed this increase. Nonetheless, the ultimate goal of the NSPBPS is to obtain a duty roster that evenly distributes the preference satisfaction among all nurses and for that the proposed algorithm achieved better results.

6 CONCLUSION

This work proposed an hybrid algorithm that uses an exact method (for solving bottleneck assignment problems) and a VNS metaheuristic, to solve the nurse scheduling problem with balanced preferences satisfaction (NSPBPS), a variant of the nurse scheduling problem (NSP). In the NSPBPS the aim is to satisfy nurses' preferences in such a way that the individual satisfaction of all nurses is evenly attended. Therefore, the final timetable should be fair regarding the satisfaction of individual preferences.

The proposed algorithm employs an exact method to solve a series of bottleneck assignment problems in order to generate an initial feasible solution. Then, a VNS metahueristic was employed to improve the initial solution using two types of neighbourhoods, the CRP (cutting and recombination procedure) suggested by Constantino et al. (2011) and a new neighbourhood structure k-Swap. Experimental results showed that the proposed algorithm managed to achieve a better balance of the individual preferences satisfaction. However, the overall number of constraint violations raised. Nonetheless, as the main goal in solving the NSPBPS is to present a duty roster that is more balanced in terms of individual preferences satisfaction, the proposed algorithm achieved a

significant improvement over the algorithm by Constantino et al. (2011).

After evaluating the experimental results, it became evident that providing a more balanced roster means that the algorithm looses ability to minimise the overall number of constraint violations. Hence, we can conjecture that a trade-off exists between balancing the solution against minimising constraint violations. Hence, future work could investigate the multi-objective aspect of the NSPBPS and evaluate the feasibility of applying a multi-objective technique to solve the problem. Additionally, we aim to provide results for the monthly scenarios available in the NSPLib.

ACKNOWLEDGEMENTS

We thank CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazilian Ministry of Education), Araucária Foundation (Fundação Araucária do Paraná) and CNPQ (National Council for Scientific and Technological Development) for their financial support of this research.

REFERENCES

Bard, J. F. and Purnomo, H. W. (2005). A column generation-based approach to solve the preference scheduling problem for nurses with downgrading. *Socio-Economic Planning Sciences*, 39(3):193 – 213.

Beddoe, G., Petrovic, S., and Li, J. (2009). A hybrid metaheuristic case-based reasoning system forÂănurseÂărostering. *Journal of Scheduling*, 12(2):99–119.

Burke, E., De Causmaecker, P., Berghe, G., and Van Landeghem, H. (2004). The state of the art of nurse rostering. *Journal of Scheduling*, 7(6):441–499.

Burke, E. K., Curtois, T., Post, G., Qu, R., and Veltman, B. (2008). A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem. *European Journal of Operational Research*, 188(2):330 – 341.

Carpaneto, G. and Toth, P. (1981). Algorithm for the solution of the bottleneck assignment problem. *Computing*, 27(2):179–187.

Cheang, B., Li, H., Lim, A., and Rodrigues, B. (2003). Nurse rostering problemsâĂŞâĂŞa bibliographic survey. *European Journal of Operational Research*, 151(3):447 – 460.

Cheng, M., Ozaku, H., Kuwahara, N., Kogure, K., and Ota, J. (2007). Simulated annealing algorithm for daily nursing care scheduling problem. In *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*, pages 507–512.

Constantino, A., Landa-silva, D., de Melo, E., and Romão, W. (2011). A heuristic algorithm for nurse scheduling with balanced preference satisfaction. In *Computational Intelligence in Scheduling (SCIS), 2011 IEEE Symposium on*, pages 39–45.

Constantino, A., Landa-Silva, D., Melo, E. L., Mendonça, C. F. X., Rizzato, D. B., and Romão, W. (2013). A heuristic algorithm based on multi-assignment procedures for nurse scheduling. *Annals of Operations Research*, pages 1–19.

Goodman, M., Dowsland, K., and Thompson, J. (2009). A grasp-knapsack hybrid for a nurse-scheduling problem. *Journal of Heuristics*, 15(4):351–379.

Maenhout, B. and Vanhoucke, M. (2005). Nsplib – a nurse scheduling problem library: a tool to evaluate (meta-)heuristic procedures. In *O.R. in health*, pages 151–165. Elsevier.

Maenhout, B. and Vanhoucke, M. (2006). New computational results for the nurse scheduling problem: A scatter search algorithm. In Gottlieb, J. and Raidl, G., editors, *Evolutionary Computation in Combinatorial Optimization*, volume 3906 of *Lecture Notes in Computer Science*, pages 159–170. Springer Berlin Heidelberg.

Maenhout, B. and Vanhoucke, M. (2007). An electromagnetic meta-heuristic for the nurse scheduling problem. *Journal of Heuristics*, 13(4):359–385.

Maenhout, B. and Vanhoucke, M. (2008). Comparison and hybridization of crossover operators for the nurse scheduling problem. *Annals of Operations Research*, 159(1):333–353.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097 – 1100.

Ohki, M., Uneme, S., and Kawano, H. (2008). Parallel processing of cooperative genetic algorithm for nurse scheduling. In *Intelligent Systems, 2008. IS '08. 4th International IEEE Conference*, volume 2, pages 10–36–10–41.

Osogami, T. and Imai, H. (2000). Classification of various neighborhood operations for the nurse scheduling problem. In Goos, G., Hartmanis, J., Leeuwen, J., Lee, D., and Teng, S.-H., editors, *Algorithms and Computation*, volume 1969 of *Lecture Notes in Computer Science*, pages 72–83. Springer Berlin Heidelberg.

Oughalime, A., Ismail, W., and Yeun, L. C. (2008). A tabu search approach to the nurse scheduling problem. In *Information Technology, 2008. ITSim 2008. International Symposium on*, volume 1, pages 1–7.

Petrovic, S. and Vanden Berghe, G. (2008). Comparison of algorithms for nurse rostering problems. In *Proceedings of The 7th International Conference on the Practice and Theory of Automated Timetabling*, pages 1–18.

Tsai, C.-C. and Li, S. H. (2009). A two-stage modeling with genetic algorithms for the nurse scheduling problem. *Expert Systems with Applications*, 36(5):9506 – 9512.

Yilmaz, E. (2012). A mathematical programming model for scheduling of nurses' labor shifts. *Journal of Medical Systems*, 36(2):491–496.