A Heuristic Algorithm For Nurse Scheduling With Balanced Preference Satisfaction

Ademir A. Constantino
Department of Computer Science
State University of Maringá
Maringá, 87020-900, Brazil
Email: ademir@din.uem.br

Dario Landa-Silva
School of Computer Science
University of Nottingham
Nottingham, NG8 1BB, UK
Email: dario.landasilva@nottingham.ac.uk

Everton Luiz de Melo,
Department of Computer Science
State University of Maringá
Maringá, 87020-900, Brazil
Email: everton_lm@yahoo.com.br

Wesley Romão
Department of Computer Science
State University of Maringá
Maringá, 87020-900, Brazil
Email: wesley.uem@gmail.com

Abstract— **This paper tackles the nurse scheduling problem with balanced preference satisfaction which consists of generating an assignment of shifts to nurses over a given time horizon and ensuring that the satisfaction of nurses' personal preferences for shifts is as even as possible in order to ensure fairness. We propose a heuristic algorithm based on successive resolutions of the bottleneck assignment problem. The algorithm has two phases. In the first phase, the algorithm constructs an initial solution by solving successive bottleneck assignment problems. In the second phase, two improvement procedures based on re-assignment steps are applied. Computational tests are carried out using instances from the standard benchmark dataset NSPLib. Our experiments indicate that the proposed method is effective and efficient, reducing discrepancies (hence improving fairness) between the individual rosters.**

Keywords-component: Nurse Scheduling Problem; Bottleneck Assignment Problem; Heuristics; Combinatorial Optimization

I. INTRODUCTION

We tackle the *nurse scheduling problem* (NSP) which consists on assigning work shift patterns to teams of nurses in such way that the skills demand in each day is met and nurses' preferences for work shifts and days are satisfied as much as possible. We tackle a variant of the problem in which the total preferences satisfaction should be evenly distributed among all nurses as much as possible. Here, we call this problem: *nurse scheduling with balanced preference satisfaction* – NSPBPS. The aim is to ensure that all nurses are 'equally' satisfied with their individual roster according to the preferences that they expressed. The total preferences satisfaction for each nurse is given by the sum of the preferences 'satisfaction' considering each shift assigned to the nurse in the duty roster. In order to achieve the desired balanced preference satisfaction among all nurses, we aim to construct the rosters in such way as to maximize the minimum individual total preference satisfaction. That is, we try to avoid individual rosters with too low total preferences satisfaction, which is equivalent to seeking a 'fair' satisfaction of individual preferences.

Like other personnel scheduling problems, the NSP can be approached in three ways. One is *cyclic scheduling* where work patterns are generated and rotated among nurses (e.g. [1]). Another is *self-scheduling* where following some hierarchical order, nurses choose the shifts they want to work provided the assignment is feasible (e.g. [2]). The third way is *preference scheduling* where cyclic and self-scheduling are combined for constructing schedules (e.g. [3]). Preference scheduling is more flexible that cyclic scheduling but also fairer than self-scheduling because all nurses' preferences are considered simultaneously.

According to Petrovic and Vanden Berghe [4] the more common objectives in nurse scheduling problems include:

- minimising the number of constraint violations,
- minimising the number of nurses
- minimising overtime,
- maximising coverage,
- maximising satisfaction of personal preferences.

We argue that the specific objective of maximizing the minimum individual total preference satisfaction is a new way to address the overall satisfaction of personal preferences. Then, we think that the NSPBPS is an important version of the NSP, particularly in scenarios where every nurse is assigned a similar workload in the given time horizon.

Nurse scheduling is a complex combinatorial optimisation problem classified by Osogami and Imai [5] as NP-hard. In the literature, we find a huge variety of problem descriptions and models due to the different characteristics and policies of each particular hospital. Consequently, there is a wide variety of different problem instances and solutions procedures and therefore, a fair comparison between the proposed procedures seems to be an idealist target [6].

Comprehensive literature surveys of the NSP are given by Cheang et al. [7] and Burke et al. [8]. They discussed

resolution methods and different points of view from which the problem can be analysed. Besides, both survey articles report the need for a set of benchmark problem instances in order to compare the many algorithms proposed to solve the NSP. Towards this, Maenhout and Vanhoucke [6] proposed a large dataset called NSPLib in which six indicators were introduced for measuring the complexity of each problem instance. That benchmark dataset includes a good variety of nurse scheduling problem instances that allow researchers to compare their algorithms. Maenhout and Vanhoucke [6], [9], [10], [11] proposed new challenges related to nurse scheduling by introducing the NSPLib and new results.

In addition to the surveys by Cheang et al. [7] and Burke et al. [8], there are more recent publications addressing the NSP using different solution methods and benchmark datasets. Many of these methods use meta-heuristic approaches, like the Electromagnetic method [10], Scatter Search [9], various Genetic Algorithms [11], [12], [13], [14], Tabu Search [15], Variable Neighbourhood Search (VNS) [16], GRASP [17], Memetic Algorithms [18], and Simulated Annealing [19].

There is also a number of references applying mathematical programming to tackle the NSP. Examples include Integer Programming [20] and Column Generation (using a model based on the Set Covering Problem) [21]. Some Artificial Intelligence techniques have also been applied to tackle the NSP including Case-Based Reasoning combined with a Memetic Algorithm [22] and Fuzzy Set Theory [23].

According to Cordeau et al. [24] a good heuristic must satisfy some criteria like *simplicity* and *flexibility*, besides *accuracy* and *speed*. Cordeau et al. [24] also say: "algorithms that contain too many parameters are difficult to understand and unlikely to be used". Our algorithm uses only two easy to set-up parameters. Also, the algorithm is flexible in that is well suited for tackling different constraints (hard and soft) without the need for major modifications. Accuracy and speed are also observed features in our approach as shown in our experiments and results. Our algorithm uses the well-known bottleneck assignment problem solved to optimality by a polynomial-time algorithm.

The remainder of this paper is structured as follows. The problem description is given in Section II, a high-level description of the proposed algorithm is provided in Section III while the solution method is described in detail in Section IV. Experiments are presented and discussed in Section V while Section VI concludes the paper.

II. DESCRIPTION OF THE NURSE SHCEDULING PROBLEM

The nurse scheduling problem addressed in this paper follows the definition given by Maenhout and Vanhoucke [10]. However, as we explained above, our specific objective is to achieve *balanced preference satisfaction* instead of maximum overall preference satisfaction. The problem involves a number of hard and soft constraints to be considered when elaborating each individual schedule. The hard constraints are related to labour law impositions and the soft constraints represent desirable aspects of a workday as expressed by nurses.

Hard Constraints: There are two hard constraints. One of them prohibits that an afternoon shift is assigned after or before a morning shift. The other constraint prohibits that a night shift is assigned before a morning or an afternoon shift.

Soft Constraints: There are four requirements expressed by nurses. The first one is for a minimum and maximum number of working days within the scheduling period. The second one is for a minimum and maximum number of consecutive working assignments. The third one is for a minimum and maximum number of assignments of each shift type in the scheduling period. The fourth soft constraint is for a minimum and maximum number of consecutive shift assignments of the same type.

Nurses Preferences: These refer to specific shifts types that nurses wish to work in each day. These preferences are declared in advance and then a cost (inversely proportional to the preference) is associated to each shift. Every soft constraint violation incurs a penalty added to the solution total cost.

More formally, the NSP can be defined as the problem of scheduling a set N of nurses within a period of D days. Given a set of possible shifts S for each day, each nurse has to be assigned a shift for each day in the scheduling period, while taking into account the above requirements and constraints. The goal is to optimize the overall satisfaction of nurses' preferences. Then we have:

N: set of nurses, index n (n=1,...,n_{max}), n_{max}=|N|;

D: set of days in the scheduling period, index d (d=1, ..., d_{max}), d_{max}=|D|.

S_d: set of shifts for day d, index s (s=1,...,s_d), s_d=|S|.

The term *shift* 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_{max} shifts assigned to a nurse. In this paper s_d is the minimum number of nurses required for day d.

Let pf_n be a sum of the costs (preferences non-satisfaction) for nurse n regarding his/her individual roster. Thus, if we wish to optimize the overall nurses' satisfaction, we should construct and assign rosters to nurses in such way that the sum of pf_n for all n (n=1,...,n_{max}) is minimized. However, to aim for rosters with balanced preference satisfaction, we have to minimize the maximum pf_n for all n (n=1,...,n_{max}). This objective is used in the well-known bottleneck assignment problem as shown next.

III. 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. 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=(T, A), where T is the set of vertices and A is the set of edges. The set T is composed by the subsets (partitions) of vertices T_0, T_1, T_2,..., T_{dmax}, where T_0 is the set of vertices representing the nurses and T_d (d from 1 to d_{max}) is the set of vertices representing day d of the scheduling period. The problem is to find n paths going from the first to

the last partition while minimizing 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 is to make a one-to-one assignment between two sets of the same size in such way that the maximum of the assignment costs is minimized. Then, solving each successive BAP in our approach corresponds to minimizing the maximum cost due to the non-satisfaction of individual preferences. The bottleneck assignment problem (BAP) is formulated as follows:

$$\text{Minimize:} \quad Z$$

$$\text{Subject to:} \quad \sum_{i=1}^{n} x_{ij} = 1; \quad j = 1,\ldots,n; \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = 1; \quad i = 1,\ldots,n; \tag{2}$$

$$c_{ij} \cdot x_{ij} \leq Z; \tag{3}$$

$$x_{ij} \in \{0,1\}; \quad i, j = 1,\ldots,n; \tag{4}$$

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 NRP in this way is that the bottleneck assignment problem can be solved in polynomial time using the algorithm proposed by Carpaneto and Toth [25].

IV. The Proposed Heuristic Algorithm

The heuristic algorithm proposed here is divided in two phases, both based on successive resolutions of a bottleneck assignment problem between two consecutive partitions as outlined above. In the first phase, an initial solution is built. In the second phase two procedures are employed in order to improve the initial solution by modifying the assignment between the partitions.

A. Construction Phase

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

$$f(i,j,d) = pc(i,j,d) + P_h \cdot nHCV + P_s \cdot nSVC \tag{5}$$

where $pc(i,j,d)$ is the preference cost associated to assigning 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. So, matrix $[c_{ij}^d]$ may be divided into two blocks (Figure 1). 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). Figure 1 illustrates how to construct this cost matrix $[c_{ij}^d]$.

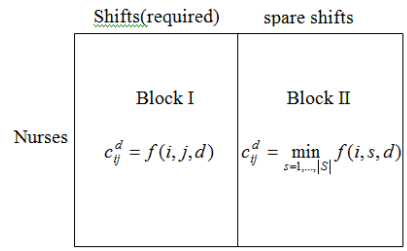

Figure 1. Cost matrix structure.

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_{max} BAPs, we have constructed an initial solution (duty roster) for the set of n_{max} 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.
```

B. Improvement Phase

The improvement phase is composed of two procedures and aims to improve the initial solution obtained in the construction phase. The first procedure, called *Cutting and Recombination Procedure* (CRP), performs successive 'cuts' before each day d, dividing the duty roster in two parts (left and right-hand side) and performs a recombination of each part as shown in Figure 2. After this, a new assignment problem is formulated and a new recombination is made (see Figure 3).

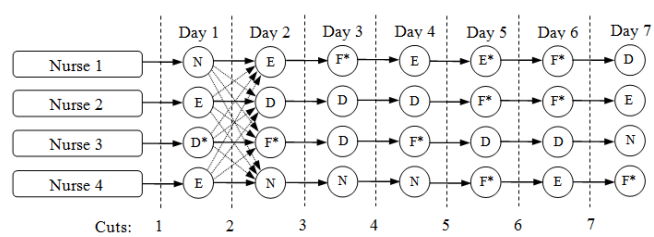

Figure 2. Example of a multipartite graph for 4 nurses, 7 days, showing a duty roster with a cut before day 2 and possible recombinations (dashed lines) of partial rosters. The letters E,D,N,F mean Early, Day, Night and Free shift, respectively, and * means spare shift.

For each cut, a new square cost matrix is built, but now c_{ij}^{d} represents the cost of assigning roster j from day d onwards (right-hand side of cut) to nurse i while considering the shifts already assigned This assignment cost is given by equation (5). Note that the cost matrix is calculated going through the duty roster from day d onwards (right-hand side) for each nurse i. During this process, the algorithm verifies which spare shift may be replaced so that the recombination has a reduced cost. This replacement occurs due the different nurses' preferences. That is, a shift previously assigned to a nurse may have different preference cost when it is assigned to another nurse.

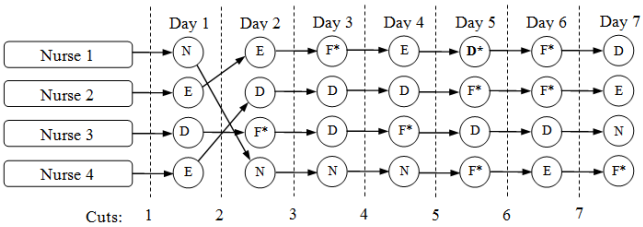

Figure 3. Result of a possible recombination after solving the corresponding assignment problem. Note that in day 5 a spare shift was changed for reducing the cost corresponding to nurse 2 - because we have assumed that nurse 2 prefers shift D instead of shift E.

The pseudo code of the CRP procedure is given below:

```
Procedure CRP
Begin
   For d=1 to d_max do:
      Construct the cost matrix for day d;
      Solve the BAP for the cost matrix;
      Recombine the rosters according to the
         BAP solution;
End.
```

The second improvement procedure, called *Shift Redistribution Proceeding* (SRP), aims to decrease the total cost of the solution by redistributing tasks among nurses on each day (see Figure 4).

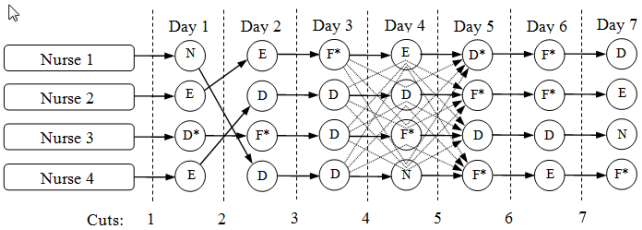

Figure 4. New possibilities of shift association to Day 4 are represented by dashed arrows.

Since the NSP tackled here involves costs related to individual preferences, the same shift assigned to different nurses may incur different costs. The SRP procedure consists of selecting a day (partition) and associating each of the n shifts of this day to each of the n rosters. The cost of each association is an element of the matrix $[c_{ij}^{d}]$, where c_{ij}^{d} is the cost of replacing shift j in the roster for nurse i. Analogous to the CRP procedure, the calculation of such costs involves the minimum cost spare shift as well as penalties due to constraints violation. Then some spare shift may be replaced as well.

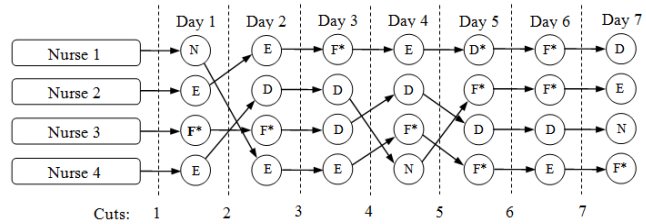

Figure 5. Rosters after shift redistribution in one day, including the change of a spare shift (from D to F) on day 1 for nurse 3.

Once the cost matrix is generated and the related assignment problem is solved, the solution is altered through shift exchanges. Figure 5 shows an example of such alteration.

This SRP procedure is repeated for all partitions according to the following pseudo code:

```
Procedure SRP
Begin
   For d=1 to d_max do:
      Construct the cost matrix to the day d;
      Solve BAP to the cost matrix;
      Replace the shift in the rosters
         according to the BAP solution;
End.
```

The improvement procedures CRP and SRP are executed in an intercalated manner, covering partitions in both directions, *forward* (d=1 up to d_{max}) and *backward* (d= d_{max} downto 1), until there is no improvement for a certain number of iterations (NumIt). Based on this idea we created four procedures: RCP_Forward(s), SRP_Forward(s), RCP_Backward(s), SRP_Backward(s), where s represents a solution or full roster. Each of the above procedures returns an improved solution given the current solution s. Let $Val(s)$ be the objective function value of solution s, i.e. the objective function value of the last assignment problem solved. Then, the algorithm for the general improvement procedure is given next.

```
Procedure Improvement(s)
Begin
   count:=0;
   Repeat
      s':=s;
      s':=RCP_Forward(s');
      s':=SRP_Forward(s');
      s':=RCP_Backward(s');
      s':=SRP_Backward(s');
     if Val(s')=Val(s) then
        count:=count + 1
      else
        count:=0;
   until count=MaxIt;
end.
```

V. Performance Analysis of the Algorithm

The proposed algorithm was implemented in Pascal and tests were performed on a PC with two 3.2 GHz Xeon Quad Core processors and 16GB of RAM running the operating system Windows XP. The instances used in these experiments were obtained from the NSPLib [6]. As this computer had a total of 8 cores, we divided the data base in 8 parts and

executed 8 simultaneous experiments. These experiments took more than 182 hours of computation time.

The NSPLib library has instances that involve weekly schedules. Four problem sizes are considered: 25, 50, 75, and 100 nurses. For each of the 4 problem sizes there are 7,290 files with different set of requirements per day and different preference costs making a total of 29,160 files. There are 8 different cases with different preferences and coverage constraints, numbered from 1 to 8, with which each one of the files may be combined. Each case has different set of coverage constraints. Then, a total of 233,280 instances of the problem can be created. The other group of instances in the NSPLib involves monthly schedules (28 days) and two problem sizes: 30 and 60 nurses. There are 960 files for each problem size and 8 cases, numbered from 9 to 16 with which each of the files may be combined. Therefore, in this second group there are 15,360 possible problem instances. Taking both groups into account we got a total of 248,640 instances.

The values for constraint violation penalties used here are the same to the ones used in [6] and [9], that is, $P_h = P_s = 100$. Besides, we fixed $MaxIt = 3$ in procedure Improvement(s).

Although the minimum coverage constraint is always satisfied by our algorithm, this is not guaranteed in some infeasible solutions reported in the NSPLib. Then, in order to compare our solutions to those infeasible solutions reported in previous results, we made some adjustments. At the end of the improvement phase, infeasible solutions are modified so that all remaining constraints are satisfied even if it provokes coverage constraints violations. Of course, each coverage constraint violation implies an additional penalty cost to the solution.

A. Results and Discussion

The results are divided in two groups, weekly and monthly scheduling. Our algorithm is called *MBAP* (Multi-Bottleneck-Assignment-Problem-based Algorithm).

All results reported so far in the NSPLib correspond to the problem of minimizing the total cost (preferences plus penalties). As explained above, our objective is to minimize the maximum individual cost (including preferences and penalties as well). The NSPLib results report the total cost for each instance, but does not report the cost for each individual roster assigned to a nurse. In order to facilitate the comparison of results, we created a new version of our algorithm (called *MAP*) using the classical assignment problem instead of the bottleneck assignment problem. The *MAP* approach aims to archive results similar to that reported by NSPLib focused on minimizing the total cost for the whole roster.

Results obtained by *MBAP* and *MAP* on 248,640 instances are shown in Table I. For each group of instances and each algorithm, this table shows the total number of solved instances (#Inst), the average minimum (*min*) cost, the average maximum (*max*) cost and the average difference between *min* and *max*, i.e. *variation*=*max* – *min*. For instance, the first line summarizes the results of 58,320 instances with 25 nurses and 7 days, where 11.187 is the average of the minimum cost rosters, 19.778 is the average of the maximum cost rosters and 8.591 (19.778 – 11.187) is the variation. The results in Table I show that *MBAP* performed well on all instances and it got smaller average variation between rosters with minimum and maximum total preference satisfaction. A smaller variation means that the individual nurse roster with lower preferences satisfaction is closer in cost to the roster with more satisfaction.

TABLE I. ROSTERS COST BY *MAP* AND *MBAP*.

|N|	|D|	Case	#Inst	*MAP*			*MBAP*		
				Min	max	variation	min	max	variation
25	7	1-8	58,320	11.187	19.778	8.591	12.994	19.208	6.214
50	7	1-8	58,320	11.972	20.541	8.570	13.563	19.965	6.401
75	7	1-8	58,320	12.602	20.351	7.748	13.960	19.685	5.724
100	7	1-8	58,320	10.162	20.531	10.368	11.677	19.583	7.906
30	28	9-16	7,680	44.612	78.966	34.353	56.263	77.545	21.281
60	28	9-16	7,680	43.165	80.170	37.005	55.151	78.056	22.905

In order to get a relative comparison we used the *%GAP* value calculated as follows:

$$\%Gap=(Z_{MBAP} - Z_{MAP})/ Z_{MAP} \times 100 \quad (6)$$

where Z_{MBAP} and Z_{MAP} are the value of the solution reached by *MBAP* and *MAP* respectively.

Table II summarizes the results. Note that a negative value indicates that the value achieved by *MBPA* is smaller than the one achieved by *MAP*, meaning that the *MBPA* approach achieved an improvement.

TABLE II. RELATIVE PERCENTAGE BETWEEN *MAP* AND *MBAP*.

|N|	|D|	Case	#Inst	%Gap		
				min	*max*	*variation*
25	7	1-8	58,320	16.15%	-2.88%	-27.67%
50	7	1-8	58,320	13.29%	-2.81%	-25.31%
75	7	1-8	58,320	10.78%	-3.27%	-26.13%
100	7	1-8	58,320	14.91%	-4.63%	-23.75%
30	28	9-16	7,680	26.12%	-1.80%	-38.05%
60	28	9-16	7,680	27.77%	-2.64%	-38.10%

The values in Table II show that *MBPA* got duty rosters with better distribution of preferences satisfaction among the nurses. The average relative variation goes from 23.75%, with 100 nurses, to 38.10%, with 60 nurses.

B. Performance of the Improvement Procedures

We now show how much each procedure CRP and SRP can improve a given solution. Table III shows results from tests with some instances. On the table we show the initial solution cost (InitSol), the cost obtained after applying CRP to the initial solution (CRP-Sol), the percentage reduction achieved by CRP (%CRP), the cost obtained after applying SRP to the initial solution (SRP-Sol), the percentage reduction achieved by SRP (%SRP), the cost obtained by applying both CRP and SRP (CRP&SRP) to the initial solution, and the percentage reduction after applying both CRP and SRT (%CRP&SRP).

Table III shows that CRP is able to achieve more cost reductions than SRP. On some instances, CRP alone could achieve the same improvement than the one obtained by using

both procedures. However, overall, the table shows that the combined procedures reached better results than CRP or SRP working separately.

TABLE III. CONTRIBUTION OF EACH IMPROVEMENT PROCEEDING ON COST REDUCTION.

|N|	|D|	*File*	InitSol	CRP-Sol	%CRP	SRP-Sol	%SRP	CRP&SRP	%CRP&SRP
25	7	1	343	309	9.91	313	8.74	307	10.49
50	7	1	1123	580	48.35	584	47.99	580	48.35
75	7	1	939	880	6.28	882	6.07	880	6.28
100	7	1	2,476	1,289	47.94	1292	47.81	1,289	47.94
30	28	1	3,998	1,583	60.40	2149	46.24	1,573	60.65
60	28	1	6,267	3,186	49.16	3364	46.32	3,184	49.19

VI. CONCLUSIONS

This work proposed a hybrid (mathematic programming and local search) heuristic algorithm for tackling the nurse scheduling problem with balanced preference satisfaction. The aim in this problem is to distribute the total preference satisfaction as evenly as possible over all nurses in order to improve fairness. The proposed approach is based on exact assignment procedures with polynomial time complexity that solve a series of individual nurse sub-problems (modelled as bottleneck assignment problems) of the overall NSP.

The rosters obtained by the proposed algorithm *MBAP* (Multi-Bottleneck-Assignment-Problem-based Algorithm) are better with respect to the balanced preference satisfaction than the solutions reported in the NSPLib data base. Thus, with this work we also propose a new nurse scheduling challenge by introducing new results for the NSPLib data base which may be used by other researchers.

It is common for heuristic algorithms to use randomization, particularly meta-heuristic approaches. However, the algorithm proposed in this paper is deterministic and hence multiple executions always generate the same results for the same input data. Other advantages of our algorithm are that it does not involve much parameter tuning and it is flexible to incorporate new constraints by just introducing new values on the cost matrix for the bottleneck assignment problems.

Nurse re-scheduling occurs in many health institutions due to changes in workload demand, staff availability, etc. Another interesting aspect of the proposed method is the possibility of using it for nurse re-scheduling. The improving algorithms can be applied from the day the change happened while previous days are just treated as historic records but not modified.

Future research work includes investigating extensions to our method by considering new improvement procedures, investigating the combination of these procedures with meta-heuristics in order to develop hybrid versions as suggested by Burke et al. [8]. Once we have other improvement procedures, we can use each procedure for neighbourhood search within a VNS meta-heuristic for example.

ACKNOWLEDGMENT

This work was partly supported by the CNPq (Brazilian National Council for Scientific and Technological Development) and CAPES (Brazilian Ministry of Education).

REFERENCES

[1] Howell, J.P., “Cyclical scheduling of nursing personnel”, Hospitals, vol. 40, pp. 77-85, 1966.

[2] Miller, M.L., “Implementing self-scheduling”, The Journal of Nursing Administration, vol. 14, pp. 33-36, 1984.

[3] Warner, D.M. “Scheduling Nursing Personnel According to Nursing Preference: A Mathematical Programming Approach”, Operations Research, vol. 24, pp. 842-856, 1976.

[4] Petrovic. S.. Vanden Berghe. G., “Comparison of Algorithms for Nurse Rostering Problems”, In: Proceedings of The 7th International Conference on the Practice and Theory of Automated Timetabling, 2008, pp. 1-18.

[5] Osogami, T., Imai, H., “Classification of Various Neighborhood Operations for the Nurse Scheduling Problem”, Lecture Notes in Computer Science, vol. 1969, pp. 72-83, 2000.

[6] Maenhout, B., Vanhoucke, M., “An electromagnetic meta-heuristic for the nurse scheduling problem”, 2005, Working Paper - Faculteit Economie en Bedrijfskunde, Gent, 2005.

[7] Cheang, B., Li, H., Lim, A., Rodrigues, B., “Nurse rostering problems - a bibliographic survey”, Eur. J. Oper. Res., vol. 151, pp. 447-460, 2003.

[8] Burke, E.K., Causmaecker, P., Vanden Berghe, G., Landeghem, H.: The state of the art of nurse rostering. J. Sched, vol.7, pp. 441-499, 2004.

[9] Maenhout, B., Vanhoucke, M., “New Computational Results for the Nurse Scheduling Problem: A Scatter Search Algorithm”, Lecture Notes in Computer Science, vol. 3906, pp. 159-170, 2006.

[10] Maenhout, B., Vanhoucke, M., “An electromagnetic meta-heuristic for the nurse scheduling problem”, Journal of Heuristics, vol. 13, pp. 359-385, 2007.

[11] Maenhout, B., Vanhoucke, M., “Comparison and hybridization of crossover operators for the nurse scheduling problem”, Ann. Oper. Res, vol. 159, pp. 333-353, 2008.

[12] Ohki, M., Morimoto, A., Miyake, K., “Nurse scheduling by using cooperative GA with efficient mutation and mountain-climbing operators”, In: Proceedings of the Third International IEEE Conference on Intelligent Systems, 2006, pp. 164-169.

[13] Ohki, M., Uneme, S., Kawano, H., “Parallel processing of cooperative genetic algorithm for nurse scheduling”, In: Proceedings of the Fourth International IEEE Conference on Intelligent Systems, 2008, pp. 36-41.

[14] Tsai, C., Li, S., “A two-stage modeling with genetic algorithms for the nurse scheduling problem”, Expert Systems with Applications, vol. 36, pp. 9506-9512, 2009.

[15] Oughalime, A., Ismail, W., Yeun, L., “A tabu search approach to the nurse scheduling problem”, In: Proceedings of International Symposium on Information Techonology, 2008, pp. 1-7.

[16] Burke, E.K., Curtois,T., Post, G., Qu, R., Veltman, B., “A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem”, Eur. J. Oper. Res., vol. 188, pp. 330-341, 2008.

[17] Goodman, M., Downsland, K., Thompson, J., “A GRASP-knapsack hybrid for a nurse-scheduling problem”, Journal of Heuristics, vol. 15, pp. 351-379, 2009.

[18] Ozcan, E., “Memetic Algorithms for Nurse Rostering”, In: 20th International Symposium on Computer and Information Sciences, Lecture Notes in Computer Science, 2005, pp. 482-492.

[19] Cheng, M., Ozaku, H. I., Kuwahara, N., Kogure, K. and Ota, J., “Simulated Annealing Algorithm for Daily Nursing Care Scheduling Problem”, In: Proceedings of the 3rd Annual IEEE Conference on Automation Science, 2007, pp. 507-512.

[20] Yilmaz, E., “A Mathematical Programming Model for Scheduling of Nurses’ Labor Shifts”, J. of Medical Systems, vol. 1, pp. 1-6 2010.

[21] Bard, J. , Purnomo, H., “Preference scheduling for nurses using column generation”, Eur. J. Oper. Res., vol. 164, pp. 510-534, 2005.

[22] Beddoe, G., Petrovic, S., Li, J., "A hybrid metaheuristic case-based reasoning system for nurse rostering", Journal of Heuristics, vol. 12, pp. 99-119, 2009.

[23] Topaloglu, S., Selim, H., "Nurse scheduling using fuzzy modeling approach", Fuzzy Sets and Systems Archive, vol. 161, pp.1543-1563, 2010.

[24] Cordeau, J.F., Gendreau, M., Laporte, G. Potvin, J.Y., Semet, F., "A guide to vehicle routing heuristics", J. Oper. Res. Soc., vol. 53, pp. 512-522, 2002.

[25] Carpaneto G., Toth P., "Algorithm for the solution of the Bottleneck Assignment Problem", Computing, vol. 27, pp. 179-187, 1981.