Solving the INRC-II Nurse Rostering Problem by Simulated Annealing based on Large Neighborhoods

Sara Ceschia · Andrea Schaerf

Abstract This paper proposes a local search method based on a large neighborhood to solve the static version of the problem proposed for the Second Nurse Rostering Competition (INRC-II). The search method, driven by a Simulated Annealing metaheuristic, uses a combination of neighborhoods that either changes the assignments of a nurse (working shift or day-off) or swaps the assignments of two compatible nurses, for multiple consecutive days. Computational results on the set of competition instances show that our method has been able to improve on a previous Simulated Annealing approach that uses smaller neighborhoods, and to get close to the level of the best available ones.

Keywords Nurse Rostering · INRC-II · Large Neighborhoods · Simulated Annealing

1 Introduction

The term "Nurse Rostering" denotes a class of scheduling problems where the objective is to create a plan of shifts for each nurse of the hospital, during a fixed planning horizon, considering the coverage requirements of nurses and their preferences, and satisfying several complex constraints related to job contracts. Nurse Rostering problems have been extensively studied by the research community (see [3,17]), solution techniques range from exact methods (e.g., [8,15]) which are very effective in handling complex rostering patterns, but whose application in practice is limited to medium size instances, to heuristic approaches. Among the latter, many relevant contributions adopted composite or variable neighborhood search strategies to deal with the complex problem structure, and obtain a deeper exploration of the solution space [7,12,13,16].

S. Ceschia · A. Schaerf
DPIA, University of Udine
via delle Scienze 206, I-33100, Udine, Italy
E-mail: {sara.ceschia,schaerf}@uniud.it

In this work, we tackle the nurse rostering problem proposed in the *Second International Nurse Rostering Competition* (INRC-II) [5], which takes into account several real-world constraints such as staff requirements, nurse skills, consecutive assignments, preferences, and week-end duties. Differently from the *First International Nurse Rostering Competition* (INRC-I) [9] where the planning horizon was fixed, in the INRC-II the problem has to be solved in multiple stages, each one corresponding to one planning week. In addition, in order to simulate a real situation happening in a hospital, it was considered also the *history* which contains information about the previous planning horizon (border data and counters).

The top two finalist of the competition both reformulated the problem as a flow model: Römer and Mellouli [14] proposed a multi-commodity IP, while Legrain *et al* [11] devise an IP model where the decision variables correspond to a *rotation* (i.e., a list of shifts on consecutive days separated by a day-off) assigned to a nurse, and then solved it using a branch-and-price procedure.

For the sake of brevity, we do not provide the problem formulation, which can be found in the INRC-II description [5]. In this work, we focus on the *static* version of the INRC-II problem, in which all information is known at the beginning of the planning horizon, and thus the solution is obtained in one single stage.

Our solution approach employs a composition of large neighborhoods [1], guided by the Simulated Annealing (SA) meta-heuristic. Our preliminary results on the set of INRC-II instances show that our method outperforms the same method based on smaller neighborhoods [6], and gives results reasonably close to the best available ones, obtaining also some best known solutions.

2 Solution Method

We solve the problem by means of Simulated Annealing, which is a local search-based metaheuristic, so we need to introduce the key elements of the local search paradigm.

2.1 Search Space and Initial Solution

The search space is defined by two integer-valued matrices: for each day of the planning horizon, the first one defines the shift (or day-off) assigned to each nurse and the second one specifies the skill employed.

The initial solution is built by assigning to each staffing requirement a random nurse among those that hold the proper skill; this way the Understaffing constraint (H2) is initially satisfied.

2.2 Neighborhood Relations

The basic neighborhood typically used in nurse rostering problems is Swap, in which the duties of two nurses in one day are swapped. The precondition of a swap move is that the nurses involved in the move are “skill-compatible”, in the sense that each one is able to fulfill the skill of the other nurse on that day.

For the INRC-II problem, given that the coverage requirements are not fixed, but they can vary from the minimum ones to the optimal ones (and above, theoretically), swap moves alone are not sufficient, as they never change the total working duties. For this reason, for example in our previous work [6], the solver uses a set-union of Swap and Change moves. A Change move assigns to one single nurse a new pair shift/skill, including the possibility to assign or remove a day-off. The selected skill must always belong to the skill-set of the nurse.

In this work, in order to try to explore the search space more effectively, we propose a generalization of the Swap and Change neighborhoods, to the case of multiple consecutive days. In detail, we propose the following two neighborhoods:

MultiSwap: The MultiSwap move generalized the Swap move by swapping the shifts of two nurses for k consecutive days. The nurses must be skill-compatible for all the duties fulfilled in all k days.

MultiChange: The MultiChange move changes the shift and/or the skill assigned to a nurse for k consecutive days. It includes also the possibility to put a nurse in day-off or remove a day of rest, independently for each day.

The MultiSwap and MultiChange neighborhoods have respectively dimensions $n \times n \times k \times (h-k)$ and $n \times s \times sk \times k \times (h-k)$, where n is the number of nurses, s in the number of shifts, sk is number of skills, k is the length of the move, and h is the number of days in the horizon.

Given that we use SA as search method, we need to draw a random move at each iteration. Due to the heterogeneity of the neighborhood, we decided to do not select the move in a uniform way, but to guide the selection using ad-hoc probabilities. In this sense, the search method can be considered a hyper-heuristic with fixed selection probability of choosing an operator (i.e., a neighborhood) [4].

First of all, the type of move (MultiSwap or MultiChange) is obtained by a weighted random selection. That is, the MultiSwap move is selected with probability p_{MS}, while MultiChange with $1 - p_{\mathsf{MS}}$, where p_{MS} is a parameter of the method. Secondly, the length k of the move is uniformly selected between 1 and a specific maximum, equal to $kmax_{\mathsf{MS}}$ and $kmax_{\mathsf{MC}}$, for MultiSwap or MultiChange, respectively.

Finally, for the MultiChange neighborhood only, in order to avoid excessively disrupting moves, while on the first day the new shift and/or skill are selected uniformly, on the subsequent days, we define three parameters for guiding the shift selection, so as to try to boost consecutive assignments to the same shift. In detail, depending on the shift assigned by the move on the previous day, we considered the probability to go on day-off p_{off}, to change the shift respect to the one of the previous day p_{change} or to keep the same shift of the previous day p_{stay}.

All the above probability-related parameters must be tuned properly, as explained in the experimental analysis.

2.3 Cost Function

The hard cost components Single assignment per day (H1) and Missing required skill (H4) are always satisfied by construction, whereas we permit moves that lead to a state that is not feasible respect to the Understaffing (H2) or Shift type succession (H3) constraints. As a consequence, the cost function adds to the objective function, that is the sum of all the soft constraints multiplied by the corresponding weight, the violations of H2 and H3 multiplied by a constant high value (500).

2.4 Simulated Annealing

In our implementation of the SA algorithm [10] the initial temperature T_0 goes down to the minimum temperature T_{min} using a geometric cooling scheme $T_n = \alpha \cdot T_{n-1}$, where α is the cooling rate whose value is between 0 and 1. For reproducibility of experiments, we fixed the maximum number of iterations I_{max}, and we compute for each temperature level the number of sampled solutions n_s using the following formula:

$$n_s = I_{max} \Bigg/ \left(\frac{-\log (T_0/T_{min})}{\log \alpha} \right). \tag{1}$$

In addition, in order to speed up the early stages of the SA procedure, we have slightly modified the cooling scheme by introducing a *cutoff* mechanism that allows to decrease the temperature prematurely if a number n_a ($n_a \leq n_s$) of the sampled solutions has been accepted already.

3 Experimental Analysis

All code is written in standard C++11 and compiled using GNU g++ (v. 5.4.0). All experiments ran on an Ubuntu Linux 16.04 machine with 32GB of RAM memory and 16 Intel® Xeon E5-2660 (2.20 GHz) cores, hyper-threaded to 32 virtual cores. A single virtual core has been dedicated to each experiment.

We test our solver on the datasets of the INRC-II competition, in particular we select the complete set of *late* instances and a subset of *hidden* dataset, as already done by Legrain *et al* [11].

Table 1 Best parameter values.

Name	Description	Value
T_0	Start temperature	110
α	Cooling rate	0.95
T_{min}	Minimum temperature	2.13
n_s	Moves sampled at each temperature	2600843
n_a	Moves accepted at each temperature	260084
p_{MS}	Probability to draw a the MultiSwap move	0.45
$kmax_{\mathsf{MS}}$	Maximum number of consecutive days for MultiSwap move	20
$kmax_{\mathsf{MC}}$	Maximum number of consecutive days for MultiChange move	2

The parameters of our solver are the ones of the SA algorithm (initial temperature, minimum temperature, cooling rate, number of sampled and accepted solutions and total iterations) and the probabilities related to the random move selection.

For the MultiChange move, we decided to give the same probability to each type of assignment (day-off, change shift, keep the same shift), and so we set the values of these three parameters all equal: $p_{\mathsf{off}} = p_{\mathsf{change}} = p_{\mathsf{stay}} = 0.33$. All the other parameters have been tuned using the F-Race procedure [2] with a confidence level of 99% (p-value = 0.01).

The total number of iterations was set to $2 \cdot 10^8$, corresponding to an average running time of 2750 seconds on our machine, which is comparable to the timeout used by Legrain *et al* [11] for the static solver. The parameter values of the winning configuration for our solution method are shown in Table 1.

From the tuning phase, it turns out that the best performing configuration is the union of the MultiSwap move that swaps the assignments of two nurse up to $kmax_{\mathsf{MS}} = 20$ consecutive days and MultiChange, that changes the shift and/or the skill of a nurse only on one day or two consecutive days ($kmax_{\mathsf{MC}} = 2$).

This big disparity between $kmax_{\mathsf{MS}}$ and $kmax_{\mathsf{MC}}$ can be explained by the fact that MultiSwap moves do not alter substantially the current shift patterns, but simply pass them from one nurse to another, whereas MultiChange moves destroy the pattern to recreate a new one.

Given that $kmax_{\mathsf{MC}} = 2$, we thus decided to split the general neighborhood MultiChange, into two smaller ones: Change and DoubleChange, with the aim of having a finer grain control and making a more efficient implementation.

The new neighborhood relation of the SA solver is thus a joint combination of Change, DoubleChange and MultiSwap. The share of probability (1 - p_{MS}) assigned to MultiChange moves is divided between the two sub-neighborhood is a non-uniform way, based on subsequent tuning. The final outcome is $p_{\mathsf{MS}} = 0.45$, $p_{\mathsf{C}} = 0.5$, and $p_{\mathsf{DC}} = 0.05$.

Our results, in comparison with the available previous ones for the static problem, are shown in Table 2. Looking at the table, we observe that our solution approach improves all the results obtained in our previous work [6], but the average values are still outperformed in all but two cases by Legrain *et al* [11] (we have obtained several best known solutions, though). Nevertheless,

Table 2 Average and best results on the benchmark.

instance	[6]		SA		[11]
	avg	best	avg	best	
030-4-1-6-2-9-1	1924.33	1830	1730.67	1700	**1695**
030-4-1-6-7-5-3	2091.00	2025	1892.17	1860	**1890**
035-4-0-1-7-1-8	1702.50	1635	1518.83	1460	**1425**
035-4-2-8-8-7-5	1419.33	1340	1215.67	1155	**1155**
040-4-0-2-0-6-1	1930.83	1850	1721.00	1655	**1685**
040-4-2-6-1-0-6	2082.50	1965	1911.50	1850	**1890**
050-4-0-0-4-8-7	1843.17	1760	1525.83	1465	**1505**
050-4-0-7-2-7-2	1828.17	1700	1503.17	1440	**1500**
060-4-1-6-1-1-5	2961.50	2850	2712.00	2640	**2505**
060-4-1-9-6-3-8	3185.67	3060	2919.33	2800	**2750**
070-4-0-3-6-5-1	2895.67	2780	2565.00	2465	**2435**
070-4-0-4-9-6-7	2601.67	2475	2313.67	2230	**2175**
080-4-2-4-3-3-3	3888.33	3740	3581.50	3495	**3340**
080-4-2-6-0-4-8	3837.00	3695	3473.83	3380	**3260**
100-4-0-1-1-0-8	1958.50	1860	1480.50	1360	**1245**
100-4-2-0-6-4-6	2517.33	2320	2073.50	1985	**1950**
110-4-0-1-4-2-8	2962.67	2905	2549.00	2455	**2440**
110-4-0-1-9-3-5	3170.17	3010	2725.33	2655	**2560**
120-4-1-4-6-2-6	2457.50	2355	**1992.67**	1880	2170
120-4-1-5-6-9-8	2558.17	2410	**2088.17**	2015	2220
Average 4 weeks	2490.80	2378.25	2174.67	2097.25	2089.75
030-8-1-2-7-0-9-3-6-0-6	2617.17	2445	2167.50	2110	**2070**
030-8-1-6-7-5-3-5-6-2-9	2281.83	2170	1855.83	1795	**1735**
035-8-0-6-2-9-8-7-7-9-8	3275.17	3170	2898.67	2755	**2555**
035-8-1-0-8-1-6-1-7-2-0	3124.33	2965	2684.00	2515	**2305**
040-8-0-0-6-8-9-2-6-6-4	3432.67	3285	2968.00	2865	**2620**
040-8-2-5-0-4-8-7-1-7-2	3178.50	2970	2759.50	2675	**2420**
050-8-1-1-7-8-5-7-4-1-8	5776.00	5610	5225.17	5095	**4900**
050-8-1-9-7-5-3-8-8-3-1	5673.00	5535	5159.67	5075	**4925**
060-8-0-6-2-9-9-0-8-1-3	3446.83	3285	2746.00	2625	**2345**
060-8-2-1-0-3-4-0-3-9-1	3697.67	3485	2982.67	2870	**2590**
070-8-0-3-3-9-2-3-7-5-2	5795.17	5515	5163.17	5035	**4595**
070-8-0-9-3-0-7-2-1-1-0	5874.00	5600	5278.67	5195	**4760**
080-8-1-4-4-9-9-3-6-0-5	5747.17	5495	4938.83	4795	**4180**
080-8-2-0-4-0-9-1-9-6-2	6056.33	5800	5229.33	5050	**4450**
100-8-0-0-1-7-8-9-1-5-4	4010.67	3655	2878.33	2765	**2125**
100-8-1-2-4-7-9-3-9-2-8	4330.50	4095	3088.00	2935	**2210**
110-8-0-2-1-1-7-2-6-4-7	5688.33	5465	4630.50	4485	**4010**
110-8-0-3-2-4-9-4-1-3-7	5271.50	5010	4169.17	4015	**3560**
120-8-0-0-9-9-4-5-1-0-3	4503.00	4265	3394.50	3240	**2600**
120-8-1-7-2-6-4-5-2-0-2	4719.17	4385	3764.33	3615	**3095**
Average 8 weeks	4424.95	4210.25	3699.09	3575.5	**3202.5**

we believe that these results are encouraging given that the solver is still preliminary and there are many possible direction for improvement. We noticed that our results are more competitive for instances with a 4 week horizon; this fact suggests that we should probably use different parameter configurations depending on the horizon.

4 Conclusions and Future Work

We proposed a SA method based on a composition of large neighborhoods. The current results are reasonably good, and can be the base for further improvements.

For the future, we plan to investigate new neighborhoods (e.g. swap between three nurses on a day) and different combinations, to deepen the impact of each cost component on the solution quality by varying its weights and to implement an iterated SA with adaptive neighborhood probabilities. In addition, we would like to test our approach on larger instances, both in terms of horizon and number of nurses, in order to determine how it scales.

References

1. Ahuja, R.K., Ergun, O., Orlin, J.B., Punnen, A.P.: A survey of very large-scale neighborhood search techniques. Discrete Applied Mathematics **123**(1), 75 – 102 (2002)
2. Birattari, M., Yuan, Z., Balaprakash, P., Stützle, T.: F-race and iterated F-race: An overview. In: Experimental methods for the analysis of optimization algorithms, pp. 311–336. Springer, Berlin (2010)
3. Burke, E.K., De Causmaecker, P., Vanden Berghe, G., Van Landeghem, H.: The state of the art of nurse rostering. Journal of Scheduling **7**(6), 441–499 (2004)
4. Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Qu, R.: Hyper-heuristics: a survey of the state of the art. Journal of the Operational Research Society **64**(12), 1695–1724 (2013)
5. Ceschia, S., Dang, N.T.T., De Causmaecker, P., Haspeslagh, S., Schaerf, A.: The second international nurse rostering competition. Annals of Operations Research pp. 1–16 (2018). DOI 10.1007/s10479-018-2816-0. Online first
6. Dang, N.T.T., Ceschia, S., Schaerf, A., De Causmaecker, P., Haspeslagh, S.: Solving the multi-stage nurse rostering problem. In: Proc. of the 11th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT-2016), pp. 473–475 (2016)
7. Della Croce, F., Salassa, F.: A variable neighborhood search based matheuristic for nurse rostering problems. Annals of Operations Research **218**(1), 185–199 (2014)
8. Glass, C.A., Knight, R.A.: The nurse rostering problem: A critical appraisal of the problem structure. European Journal of Operational Research **202**(2), 379 – 389 (2010)
9. Haspeslagh, S., De Causmaecker, P., Schaerf, A., Stølevik, M.: The first international nurse rostering competition 2010. Annals of Operations Research **218**, 221–236 (2014)
10. Kirkpatrick, S., Gelatt, D., Vecchi, M.: Optimization by simulated annealing. Science **220**, 671–680 (1983)
11. Legrain, A., Omer, J., Rosat, S.: A rotation-based branch-and-price approach for the nurse scheduling problem (2017). Working paper, available at https://hal.archives-ouvertes.fr/hal-01545421
12. Lü, Z., Hao, J.K.: Adaptive neighborhood search for nurse rostering. European Journal of Operational Research **218**(3), 865 – 876 (2012)
13. Rahimian, E., Akartunali, K., Levine, J.: A hybrid integer programming and variable neighbourhood search algorithm to solve nurse rostering problems. European Journal of Operational Research **258**(2), 411 – 423 (2017)
14. Römer, M., Mellouli, T.: A direct MILP approach based on state-expanded network flows and anticipation for multi-stage nurse rostering under uncertainty. In: Proc. of the 11th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT-2016), pp. 549–551 (2016)
15. Smet, P., Salassa, F., Vanden Berghe, G.: Local and global constraint consistency in personnel rostering. International Transactions in Operational Research **24**(5), 1099–1117 (2017)

16. Tassopoulos, I.X., Solos, I.P., Beligiannis, G.N.: A two-phase adaptive variable neighborhood approach for nurse rostering. Computers & Operations Research **60**, 150 – 169 (2015)
17. Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L.: Personnel scheduling: A literature review. European Journal of Operational Research **226**(3), 367 – 385 (2013)