A GRASP Approach using a Constructive Heuristic Approach to the Nurse Scheduling Problem

António Ramadas[1] and Pedro Castro[1]

[1] Master in Informatics and Computing Engineering at the Faculty of Engineering of the University of Porto, Rua Dr. Roberto Frias, s/n 4200-465, Porto, Portugal
`{antonio.ramadas, up201305337}@fe.up.pt`

Abstract. As a solution for the Second International Nurse Rostering Competition (INRC-II) [1], we introduce in this paper an approach based on the Greedy randomized adaptive search procedure (GRASP) with a Constructive Heuristic (CH) for the initial solution. The local search iteration is implemented with Hill Climbing. The proposed heuristic was adapted to INRC-II specification and the results obtained as the heuristic implementation are detailed in the present document. The solution obtained is also compared with the finalists of the competition.

Keywords: Scheduling, GRASP, Local Search, Heuristics, Nurse Rostering.

1 Introduction

The need for fast, automated and efficient services has been increasing over the last decade and it does not show any signs of slowing down. Computational processing power has also seen its share of growth however it has not keep up with the tremendous rise in data quantity or with customers' needs. The need for a fast solver with valid and good solutions is an object of study for several years.

Nurse rostering (or scheduling) is a particular subset of employee scheduling that has been the focus of many researches for many years. The domain includes staffing, budgeting and short-term scheduling problems, but different staff needs is what makes nurse rostering such a challenging endeavor [2]. Especially in an area where the scheduling of the workers (nurses in this case) has a big impact in their well-being and satisfaction [3] which can in turn affect the working environment.

The nurse rostering problem (NRP) consists in assigning nurses to specific shifts in a way that maximizes staff requirements and nurse satisfaction and at the same time minimizes costs and nurse burndown. The satisfaction may be attributed to shift off requests granted or abide by the maximum number of consecutive working days specified in the contract. Although we count this as satisfaction, it can also be said that the lack of satisfaction is the cost of a specific scheduling.

The proposed heuristic was a consequence of the research done to study the state of art of the nurse rostering problem. As previously mentioned, this problem is facing a lot of contributions to its solution. This paper aims to improve the only available paper implementing GRASP to this problem [4]. Its results are very promising and led us to create this proposal which aims to study a different initial approach.

We propose a variation of GRASP as the metaheuristic used by our solver to this problem implemented in C++ programming language with no framework or external library except for handling JSON syntax. GRASP can be roughly summarized as a two iteration process where the first generates an initial solution and the next iteration improves the solution through local search. Therefore, there are multiple possibilities to implement this heuristic. The proposed solution implements a constructive heuristic followed by the Hill Climbing algorithm as the local searcher.

The rest of the paper is structured as follows. Section 2 describes the Second International Nurse Rostering Competition, including the problem inputs, its constraints and solution validation. Section 3 specifies the theory behind our proposed solution, including formulation, heuristics for initial solution construction and local search. Section 4 details the results of our evaluation methodology, by comparing different constructive heuristics. Section 5 includes a personal appreciation on the achieved results, suggestions for improvements and possible next steps.

2 The Second International Nurse Rostering Competition

2.1 Problem and Complexity

Real life problems are object of an intense study by the academic community. Moreover, nurse rostering is one of the main problems where more attention is devoted. The same problem may be contextualized differently simplifying it from NP complete. As an example, Osogami and Imai [5] converted it to Boolean Programming proving that this is way it can be simplified to NP-Hard. Its definition is that this kind of problems are not solvable in a reasonable amount of time, leading it to the use of heuristics.

In order to share the results of the community in solving the nurse rostering problem, it has been created 2 editions of a competition creating the same scenario for every participant so their solvers may be evenly compared. The current paper proposes a solution under the specification of The Second International Nurse Rostering Competition. Therefore, our proposal may be later compared with the finalists. This competition defines a variant of the nurse scheduling problem as a multistage problem with 4 to 8 weeks, where a week is considered a separate and individual stage. For each week, the solver must assign a fixed number of nurses to a set of shifts, such as morning, day or night. When scheduling a single week, constraints regarding the global scope of 4 to 8 weeks must be taken into account. Some information, called history, is carried out between consecutive weeks, so that the one coming from the previous week has to be taken into account by the solver.

2.2 Inputs

The competition's organization has specified that the information provided is passed on the form of files regarding the scenario, week data and history.

The scenario represents the general data common to all stages of the overall process. It the total number of weeks of the planning horizon, the list of skills a nurse may have (a nurse has at least one skill), a set of contracts that describes the minimum and maximum total number of assignments that the nurse should have, consecutive working days as well as consecutive days off, the maximum number of working weekends and if the weekends have to be full or not. It also holds a list of the shifts with its required skill, minimum and maximum consecutive assignments of the type and a matrix of forbidden successions, that state that some shifts types cannot be followed by other certain types.

Each week will have a data file which will include all the information about that week. It is detailed for each shift, skill and weekday the optimal and minimum number of the nurses necessary for working duties and also the nurses' requests for shift or days off.

A history data file is transferred between weeks and it contains information about the history of each nurse. So, in this file, for each nurse it will be detailed the type of shift last worked and in what day, the number of current consecutive working days and the number of consecutive days off. Global counters depicting the total number of working shifts and working weekends are also presented.

2.3 Constraints

This competition provides a set of hard constraints, soft and global constraints. Hard constraints are rules that cannot be violated, rendering the solution invalid if they are. Soft constraints dictate the quality of the solution and define the weight that should penalize it. Some of these analyse the global state of the solution which means they take into account all the previous week data. Each constraint is briefly described. A mathematical formulation is supplied by the organization of the competition [13]. The following tables of constraints contain a brief description extracted from a paper also proposing a solution for this problem [14].

Table 1. Hard constraints of the INRC-II

Identification	Title	Description
H1	Single assignment per day	Each nurse can only work once per day
H2	Under-staffing	There are minimum staff needs for each shift and skill
H3	Shift type successions	Assigning two consecutive days must be legal according to the scenario
H4	Missing required skill	Each nurse must have the required skill for a shift

Table 2. Soft constraints of the INRC-II

Identification	Title	Description
S1	Insufficient staffing for optimal coverage	There is an optimal coverage per shift and skill, each missing nurse yields 30 points
S2	Consecutive assignments	There is a limit of consecutive assignments: each consecutive assignments or consecutive days of work above or below the limit yield respectively 15 points and 30 points
S3	Consecutive days off	There is a limit of consecutive days off work: each day above or below the limit yields 30 points
S4	Preferences	Nurses have shift preferences. Undesired assignments yield 10 points
S5	Complete week-end	If specified by the contract, nurses should work both weekend days, otherwise 30 points

Table 3. Soft global constraints of the INRC-II

Identification	Title	Description
S6	Total assignments	Nurse contracts enforce minimum and maximum limit of total assignments (4, 8 weeks); Each assignment above or below the limits yields 20 points
S7	Total working week-ends	Nurse contracts enforce maximum limit of working weekends; Each working weekend (1 or 2 days) above the limit yields 30 points

The goal is to minimize the total weight of the solution, i.e. its penalization. A constraint penalisation is the multiplication of the number of times it is violated and its weight.

2.4 Solution format and Validation

The competition provides a set of tools that are used to run the multi-stage simulation. A simulator takes as input the history file, the current week data and the scenario. It invokes the competitor's solver for each week iteratively and outputs a file when it terminates, describing the schedule of that particular week in addition to updating the history file. After processing every stage, a validator tool is also provided which reviews the solution given by the solver and calculates the solution's penalty regarding the soft constraints as well as it checks if the solution is feasible (no hard constraints are violated).

3 GRASP Approach using CH

3.1 Context

Has it has been said before, a lot of research has been done on the topic of Nurse Rostering. As part of the Methodologies for Planning and Scheduling it was encouraged to search for previous work done on the topic and develop a new strategy to solving this problem. Reinventing the wheel or developing our own implementation of a studied method was not the purpose of our project. The initial phase of this project consisted in researching the state of the art and analysing the current methods used and how they fare against each other [15].

Simpler methods using tradition heuristics such as Shuffling or Greedy Shuffling. These are both based on exchanging parts of the schedules of two particular nurses in order to improve the solution. The difference between the two is that the first picks the nurse with worst schedule and tries to change it with another. The greedy approach tries every shuffle, calculates the best one and applies the switch. Although slightly more complex, instead of using parts of the schedule, Goodman, Dowsland and Thompson [4] used shift patterns, that need to be pre-processed before running the search, which can apply either of the previous heuristics.

There has also been extensive study of metaheuristics applied to the nurse rostering problem. Tabu Search(TS) is widely used to solve many combinatorial problems [6]. TS as the name indicates a search heuristic that iterates from one solution to another within a neighbourhood space of solutions. What sets TS from a simple Hill Climbing search is that it uses memory structures that describe the visited solutions. Although it may iterate to a worse solution, it will not visit solutions already processed. This will help the algorithm reduce the chance of being stuck in a local minimum and the "prohibited" solutions discourages the search on previously visited solutions. A great example of an implementation of TS on a NRP was is an implementation of a hybrid Tabu search on a real system (Plane), a software used by Belgian hospitals [16].

Simulated Annealing is another derivation of the Hill Climbing algorithm. It performs the search the same way but when a Hill Climbing algorithm would chose to avoid a worse, SA probabilistically decides whether or not to visit. Gary M. Thompson presented a SA heuristic for shift-scheduling using non-continuously available employees [7] to tackle the NRP.

Genetic algorithms are stochastic metaheuristics are also the focus of studies on solving the NRP. The algorithm is based on the natural evolution of species where the strongest thrive and the weak die. The same is applied here. Several solutions are created and evaluated (which for a generation). Each solution has a probability attached to it, proportional to its value. So, the better the solution, the more chance it has of advance to the next generation. For each is solution that does not advance to the new generation a new solution is formed based on the merging of 2 other solutions therefore creating a new solution(evolution). There can also be mutations, where change occurs in a given solution given a pre-determined probability. Larger the probability, larger the chance of mutating new and old solutions. After a few iterations, or generations, the best solution till date is presented as the final solution. There are several examples of these implementations on [8], [9] and [10].

3.2 GRASP

In 1989 Resende and Feo coined the term GRASP [11]. It will be the name of an approach to solving computationally difficult set covering problems and stands for Greedy Randomized Adaptive Search Procedures. GRASP consists of a number of independent repetitions of two phases: Construct and Improve. Each GRASP iteration consists of constructing a random initial solution and then applying a local search algorithm until a local optimum is found (i.e., the final solution for that iteration). It is an iterative method because the search is applied to one element at a time. (This does not mean that parallel process cannot be used. The local search algorithm has to run each initial solution and because they share no information between them, they can be ran in parallel). It is greedy because the addition of each element to the solution is guided by a greedy function. It is adaptive because the element chosen at any iteration in a construction is a function of those previously chosen. The Improvement phase consists on running a local search algorithm on top of each initial solution. Running several GRASP iterations allows for a wider search of the solution space.

Although any local search algorithm can be used in the Improvement phase, Laguna and Martí [12] showed that using metaheuristic algorithms does not result in better end results than if a simple local search algorithm, like Hill Climbing, if only a limited time is allowed to compute the solution. So, they concluded that given a short amount of time, such as in the International Nurse Rostering Competition, there is no clear advantage of using SA or TS versus using a simpler Hill Climbing. It's is the trade-off between doing more GRASP iterations or spending more time in the improvement phase that result in this similarity. Goodman, Dowsland and Thompson [4] successfully used GRASP to solve the NRP.

3.3 CH

We came up with the idea that a good improvement to the already successful was getting rid of the semi-random initial solutions that GRASP offered and instead use constructive heuristics to build a better initial solution. Our purpose was to study the impact of the spending more time in the Construction phase with the cost of spending less time on the Improvement phase.
Our implementation of GRASP allows to choose which heuristic to use in the Construction phase. We took it upon ourselves to implement the GRASP's original Construction phase as well as two other that will be explained below. The Construction phase always produces solutions that respect the hard constraints of the competition. This means that every solution is feasible and is an answer to the NRP in question.

3.4 Local Search

Request First. This heuristic focuses on minimizing the cost of the shift request soft constraint that dictates that for each appeal not granted the solver will penalize the solution by a certain weight. It maintains the random property of the GRASP Construction phase meaning that will fill each day with a random nurse. The only difference is that when it chooses a nurse that has asked not to work in that specific shift or day, it will skip this nurse and try to allocate another. Therefore, the cost associated with the soft constraint will be 0.

More Skilled. This heuristic focuses on prioritizing the shifts with less available nurse skill resources. More Skilled, before assigning nurses, pre-processes the scenario by storing the total sum of each skill available by the nurses. Therefore, instead of sorting the shifts in a chronological order, the shifts are prioritized by first filling the ones where the skill required is the less available by the nurses. Whenever possible, this heuristic also tries to optimize the complete week-end soft constraint. This heuristic may be considered as a more complex approach than a naive First Assign, because it often led to the violation of hard constraints by poorly assigning nurses.

Finally, this heuristic does also maintain the random property of the GRASP Construction phase by randomly choosing a nurse who has the skill required by the shift. The next algorithm illustrates the implementation done.

```
sortShiftsBySkillsAvailable()
FOR EACH shift IN shifts
  WHILE shift requires more nurses DO
    assignValidRandomNurse(shift)
```

4 Experiments and Results

In this section, we describe the tests performed on the solver in order to experiment with different heuristics, as well as the results obtained. As referred, it is used 2 different constructive heuristics and 1 random generation heuristic. The results were obtained following the specification of the competition and compared to the overall winner.

4.1 Test Data Sets and Run Times

The testing of the solver was performed using datasets provided by the competition's organization (see Section 2), consisting on planning horizons of 4 up to 8 weeks, each week requiring a total of a pre-established number of nurses to be scheduled.

Each dataset was used with the different heuristics described (see section 3) in order to benchmark them and compare their efficiency with each other and the INRC-II winner. All the solutions provided do not violate any hard constraints and aim to improve the soft ones (enhance the scheduling so they don't penalise).

The competition provided up to 4 different datasets: test, competition, late and hidden. Each dataset may have different sets of nurses (30, 40, 50, 60, 80, 100 and 120) and a planning horizon of 4 or 8 weeks. We used the hidden dataset to better compare against the finalists whose solutions were evaluated by the competition's judges in order to rank the finalists of the INRC-II.

Table 1 describes the values provided by the competition's benchmark for maximum solver runtime (for a single planning week), in accordance to the hardware at use during the tests. The benchmark is an application provided by the competition that when run determines the maximum run times for the computer the application runs at.

Table 4. Maximum allowed time

Nurses	Solver Time (s)
30	62.05
35	85.32
40	108.58
50	155.12
60	291.65
70	248.19
80	294.73
100	387.80
110	434.87
120	480.87

4.2 Results and Comparison

The solutions were computed following the specifications of the competition. Also, each dataset was executed 10 times for each heuristic, including the one from the competition's winner. The following table summarizes the results obtained:

Dataset	Solver	Heuristic	Fsb. (%)	Avg.	Min.	Max	Std. Dev.
35n (0) 4w (1-7-1-8)	GRASP	Random	100	10100	8855	11295	796
		Request First	100	10160	9055	11200	641
		More Skilled	100	12476	11470	14180	1116
	Winner INRC-II	---	100	1630	1630	1630	0
35n (0) 8w (6-2-9-8-7-7-9-8)	GRASP	Random	100	24127	23295	25225	689
		Request First	100	24243	22310	26520	1334
		More Skilled	100	28464	27600	29630	911
	Winner INRC-II	---	0	99999	99999	99999	0
70n (0)	GRASP	Random	100	43973	42640	45920	1037

8w (3-3-9-2-3-7-5-2)		Request First	100	44553	42285	46410	1539
		More Skilled	100	53031	52155	53940	703
	Winner INRC-II	---	100	5164	5115	5230	35
110n (0) 8w (8-8-2-2-3-2-0-8)	GRASP	Random	0	99999	99999	99999	0
		Request First	0	99999	99999	99999	0
		More Skilled	80	93921	91050	97970	2174
	Winner INRC-II	---	100	5511	5435	5610	67

The syntax of the description of the dataset column is as follows: number of nurses, number of the scenario, number of weeks and the sequence of weeks the simulation was done. As an example, the first instance specifies the scenario numbered 0 with 35 nurses over 4 weeks starting by the week 1 and finishing at 8. Lastly, 99999 indicates the maximum and it is an indicator of infeasibility by the solver.

The results are compared by its average and it is worth noting the standard deviation is small in all the cases. To provide more information to the user, it was included the minimum and maximum values of the results.

Except for a single instance, the INRC-II winner got better results than GRASP independently of the heuristic chosen. The winner's heuristic only deteriorates on a single instance, but as the implementation has not been disclosed, we can't study it. However, our proposal provided valid solutions for all the scenarios.

Focusing on GRASP results, a random heuristic got better (equals lower) results than the other heuristics. However, this approach does not guarantee good results, but, by giving it sufficient time to try the different combinations without a logical validation, this heuristic showed very competitive results compared to more elaborate ones. Looking closely, it is possible to see the competitive results when compared against the Request First heuristic. The differences are small by roughly less or equal to 1%. Finally, it is worth noting that despite the More Skilled heuristic consistently produces the worst results, it was the only one to efficiently output 80% of the times a valid solution on time.

5 Conclusions

The GRASP random heuristic showed competitive results against CH. Perhaps more complex CH would outperform it. It is worth pointing out that random does not guarantee a solution feasibility in a reasonable amount of time, nor does CH, but these last increase the odds as proven by the results (see section Results and Comparison).
Additionally, it is worth pointing out that the GRASP's random heuristic is a stochastic method and if the solver was to be re-ran, the results may be optimal, infeasible or

simply be outperformed by the CH. A real-life implementation should have this into consideration.

This implementation of GRASP with HC in the Construction phase demonstrates that it does not benefit from spending more time during the construction phase than in the improvement phase. Like Laguna and Martí [12], we found that the multiple iterations of GRASP compensate for the fact that the value of the initial solution may be less than desirable. It is also possible to conclude that GRASP does not improve significantly when using HC versus simply using the original GRASP random construction.

It was not referred previously, but the results, when compared to the candidates (i.e. not the hidden dataset) of the competition, were very competitive. This solution outperformed some official candidates of the early stages of the competition.

To enforce the goal of our proposal, more complex heuristics for building a solution may not compensate against random one in limited time, but they do increase the odds of offering a good solution. As future work, a strong starting point would be to implement CH that create solutions satisfying the optimal requirements. However, there is a trade-off of local search enhancement (as we did) for CH output optimized (it should take more CPU cycles than our implementation).

In conclusion, if the time is not a constraint, then a CH is a good solution for large datasets by offering more confidence about delivering a feasible solution. For small to medium datasets where time is not the essence, then a random implementation shorts the time of implementation and eases the implementation process. From our experience in the implementation process, a CH takes too much time building a solution where time in this competition may be considered a luxury. As a next step, more complex heuristics are a good starting point to study, but without the time constraints of the competition. Finally, the proposed heuristics to build a solution aims to fill only the minimum requirements leaving the enhancement to local search.

6 References

1. S. Ceschia, N. Dang, P. De Causmaecker, S. Haspeslagh, A. Schaerf: The second International Nurse Rostering Competition. PATAT 2014: Proceedings of the 10th International Conference of the Practice and Theory of Automated Timetabling. PATAT 2014, 554–556 (2014)
2. Edmund K. Burke, Patrick De Causmaecker, Greet Vanden Berghe, Hendrik Van Landeghem: The State of the Art of Nurse Rostering. Journal of Scheduling 7: 441–499 (2004)
3. CW Mueller, JC McCloskey: Nurses' job satisfaction: a proposed measure,1990 Mar-Apr;39(2):113-7
4. M.D. Goodman, K.A. Dowsland, J.M. Thompson, J. Heuristics 15: 351. doi:10.1007/s10732-007-9066-7 (2009)
5. T. Osogami, H. Imai: Classification of Various Neighborhood Operations for the Nurse Scheduling Problem. In Goos, Gerhard and Hartmanis, Juris and van Leeuwen, Jan and Lee, D.T. and Teng, Shang-Hua (ed.), Algorithms and Computation, Lecture Notes in Computer Science, Vol. 1969. Springer Berlin Heidelberg (2000)

6. Fred Glover and Manuel Laguna. Tabu Search. Kluwer Academic Publishers, Norwell, MA, USA. (1997)
7. Gary M. Thompson, A simulated-annealing heuristic for shift scheduling using non-continuously available employees, Computers & Operations Research, Volume 23, Issue 3, 1996, Pages 275-288, ISSN 0305-0548
8. U. Aickelin, K. Dowsland, An indirect genetic algorithm for a nurse scheduling problem, Computers and Operations Research, submitted for publication. Available from<www.inf.brad.ac.uk/~uaickeli/papers/02COR_indirect.pdf>, 2000, p. 26
9. U. Aickelin, K. Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem. Journal of Scheduling, 3 (3) (2001), pp. 139–153
10. U. Aickelin, P. White, Building better nurse scheduling algorithms, Annals of OR, submitted for publication
11. Thomas A Feo and Mauricio G. C Resende. 1989. A probabilistic heuristic for a computationally difficult set covering problem. Oper. Res. Lett. 8, 2 (April 1989), 67-71. DOI=http://dx.doi.org/10.1016/0167-6377(89)90002-3
12. M. Laguna, R. Martí, A GRASP for coloring sparse graphs, Computational optimization and applications 19 (2) (2001) 165–178.
13. Second International Nurse Rostering Competition: mathematical model: http://mobiz.vives.be/inrc2/wp-content/uploads/2014/10/math_model.pdf [last visited June 7 2017]
14. Diogo Santos, Pedro Fernandes, Henrique Lopes Cardoso, Eugénio Oliveira: A Weighted Constraint Optimization Approach to the Nurse Scheduling Problem 2015 IEEE 978-1-4673-8297-7/15 DOI 10.1109/CSE.2015.46
15. CHEANG, Brenda; LI, Haibing; LIM, Andrew; and RODRIGUES, Brian. Nurse Rostering Problems: A Bibliographic Survey. (2003). European Journal of Operational Research. 151, (3), 447-460. Research Collection Lee Kong Chian School Of Business
16. Burke E., De Causmaecker P., Vanden Berghe G. (1999) A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. In: McKay B., Yao X., Newton C.S., Kim JH., Furuhashi T. (eds) Simulated Evolution and Learning. SEAL 1998. Lecture Notes in Computer Science, vol 1585. Springer, Berlin, Heidelberg