Constraint Logic Programming to solve the Nurse Rostering Problem

Gil Teixeira
Faculty of Engineering
University of Porto
Porto, Portugal
gil.teixeira@fe.up.pt

***Abstract*—Staff scheduling is a common problem found in vast range of fields, such as, transportation, emergency services, retail, and healthcare. It is not possible to generate optimal solutions in polynomial time. However it is possible to generate rosters which are significantly better than those produced by an expert human planner, and in a fraction of the time. In this paper, it is presented a solution to the nurse rostering problem (NRP) using contrain logic programming. This variation of the NRP was introduced by T. Curtois and is available at http://www.schedulingbenchmarks.org/nrp/. Analysis of the obtained results and a comparison with other literature approaches to the same problem is also provided.**

***Index Terms*—CLP, Scheduling, NRP**

I. Introduction

A. Context

Every year, in hospitals all over the world there are problems assigning the nurses schedules. Countless solutions have been proposed through the last decades to different variations of the problem. However, due to the problem constrains, finding the ideal roster in not an easy task, and because the problem is NP-hard it is impossible to solve in polynomial time. Constraint Logic Programming (CLP) emerged as a merge of two different programming paradigms: Constraint programming (CP) and Logic programming (LP). CLP is mostly used in Search and Optimization problems, such as the nurse rostering problem (NRP).

In this paper it is presented a solution to a variation of the NRP introduced by T. Curtois [1] and available at http://www.schedulingbenchmarks.org/nrp/. This dataset contains twenty four instances to solve in increasing degrees of complexity. The solution uses the previous mentioned paradigm CLP to model the constrains of the problem and then it tries to generate feasible solutions.

The main contributions of this work are:

- Formulation of a variation of Nurse Rostering Problem introduced by T. Curtois and R. Qu [1] using CLP.
- Experimentation with different search options and analysis of the obtained results.

In Section Related Work, we provide an overview of the solutions proposed in the literature. In Section Problem Definition, we define the problem to be solved and its constrains. In Section Implementation, we specify our proposed solution. In Section Experiments and Results, we detail the experiments performed and its results. Finally in Conclusions and Future Work, we draw conclusions on the obtained results and mention some future research directions.

II. Related Work

In [4] were presented the results of two methods for nurse rostering problems. The first developed algorithm was Branch and Price. Branch and price is a branch and bound method in which each node of the branch and bound tree is a linear programming relaxation which is solved using column generation. The column generation consists of a restricted master problem and a pricing problem. The other algorithm was an ejection chain based approach called variable depth search (VDS). These algorithms provide optimal solutions for smaller instances, however optimal solutions for larger instances is still unknown.

Other approach, presented in [6], models the problem as a maximum Boolean satisfiability problem (maxSAT). In this approach, a propositional logic formula is built from Boolean variables using logic operators. The formula is given in conjunctive normal form (CNF), meaning that the formula is a conjunction of clauses. These clauses are partitioned into two types: hard and soft clauses. Each soft clause is given a weight. The goal is to find an assignment which satisfies the hard clauses and minimizes the sum of weights of the unsatisfied soft clauses. Even though other based on integer programming provides better results than maxSAT for most of the considered instances, maxSAT could provide optimal solutions for two of the instances and obtained solutions for two very large instances within 4 h, which could not be solved by integer programming within 1 h.

III. Problem Definition

A. Constrains

The solution to the NRP consists of assigning different shifts to nurses for a specified planning period. However, some constrains related to the nurses' contracts and preferences must be satisfied. This problem is NP-Hard which makes it computationally difficulty to solve even when the problem is relatively small. There are two types of constrains: hard and soft constrains. Hard constrains are those that must be satisfied. Soft constrains are those ideally would be satisfied but if not some penalty will be added to the solution. The hard (HC)

and soft (SC) constrains of the problem, as explained by the authors of the dataset, are described as followed:

- HC1 : **Shift Per Day** - Employees cannot be assigned more than one shift on a day.
- HC2 : **Shift Rotation** - A minimum amount of rest is required after each shift. Therefore certain shifts cannot follow others. For example, an early shift cannot follow a late shift.
- HC3 : **Maximum Number of Shifts** - The maximum numbers of shifts of each type that can be assigned to employees. For example, some employees will have contracts which do not allow them to work night shifts or only a maximum number of night shifts.
- HC4 : **Maximum Total Minutes** - The maximum amount of total time in minutes that can be assigned to each nurse within the planning period.
- HC5 : **Minimum Total Minutes** - The minimum amount of total time in minutes that can be allocated to each nurse within the planning period.
- HC6 : **Maximum Consecutive Shifts** - The maximum number of shifts an employee can work without a day off. For example, part-time employees sometimes do not work as many consecutive shifts as full-time staff.
- HC7 : **Minimum Consecutive Shifts** - The minimum number of shifts that must be worked before having a day off. This constraint always assumes that there are an infinite number of consecutive shifts assigned at the end of the previous planning period and at the start of the next planning period.
- HC8 : **Minimum Consecutive Days Off** - The minimum number of consecutive days off that must be assigned before assigning a shift. This constraint always assumes that there are an infinite number of consecutive days off assigned at the end of the previous planning period and at the start of the next planning period.
- HC9 : **Maximum Number of Weekends** - The maximum number of worked weekends (a weekend is defined as being worked if there is a shift on Saturday or Sunday) within the planning period.
- HC10 : **Days off** - Shifts must not be assigned to the specified employee on the specified days.
- SC1 : **Shift on requests** - If the specified shift is not assigned to the specified employee on the specified day then the solution's penalty is the specified weight value.
- SC2 : **Shift off requests** - If the specified shift is assigned to the specified employee on the specified day then the solution's penalty is the weight value.
- SC3 : **Cover** - If the required number of staff on the specified day for the specified shift is not assigned then it is a soft constraint violation. The penalty associated with this constraint is equal to the total amount of violated coverage multiplied by the specified relevant under- or over-weight defined in the problem data.

The objective function models the requirement to maximise the allocation of employee shift requests and minimise under and over staffing. For example, an employee may request to work a certain shift type on a particular day. The higher the weight, the more important the request is to the employee. If there is no request then the parameter has the value zero. The objective function:

$$min \sum_{i \in I} \sum_{d \in D} \sum_{t \in T} v_{idt} + \sum_{i \in I} \sum_{d \in D} w_{dt}^{min} y_{dt} + \sum_{d \in D} \sum_{t \in T} w_{dt}^{max} z_{dt}$$

where:

I	is the set of nurses
D	is the set of days in the planning horizon
T	is the set of shift types
v_{idt}	total incurred penalty relevant to shift on/off requests of nurse $i \in I$ for shift type $t \in T$ on day $d \in D$
w_{dt}^{min},w_{dt}^{max}	under-weight and over-weight relevant to the total coverage of shift type $t \in T$ on day $d \in D$
y_{dt}	total number of nurses below the preferred coverage for shift type $t \in T$ on day $d \in D$
z_{dt}	total number of nurses above the preferred coverage for shift type $t \in T$ on day $d \in D$

B. Inputs

The dataset provides 24 instance in total with an increasing level of complexity. The instances can go from 2 to 52 weeks, from 8 to 150 employees and from 1 to 32 shift types. These instances are describe in straightforward text files where it is described the number of days of the planning horizon, the shifts, the staff, the nurses' mandatory days off, the shift off and shift on requests and finally the nurse requirements for every day.

C. Solution format and validation

The validation of solutions can be performed using Roster-Viewer [2], by submiting an XML file. This file must have the nurse schedule for each day and the type of shift they will perform. The program GUI will display the total penalty and whether the solution is feasible.

IV. Implementation

The implementation was develop on the system SICStus. SICStus is a state-of-the-art, ISO standard compliant, Prolog development system. Within this system the library CLP(FD): Constraint Logic Programming over Finite Domain [3] was also used.

A. Domain

To model the problem, we create for every nurse, and for every day a variable that ranges from 0 (if a nurse does not work that day) to the total number of different shift types. These variables were aggregate in a lists of lists as follows:

$$S_{n,d} = \begin{bmatrix} \begin{bmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,d} \end{bmatrix} \\ \begin{bmatrix} s_{2,1} & s_{2,2} & \cdots & s_{2,d} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} s_{n,1} & s_{n,2} & \cdots & s_{n,d} \end{bmatrix} \end{bmatrix}$$

where:

d is the number of days of the planning horizon
n is the number of nurses
$s_{n,d}$ is the shift attributed to that nurse on that day

B. Hard Constrains

1) Shift Per Day: Every nurse has a list of shifts that (s)he can do. Therefore, every nurse schedule was constrained to that list of shifts.

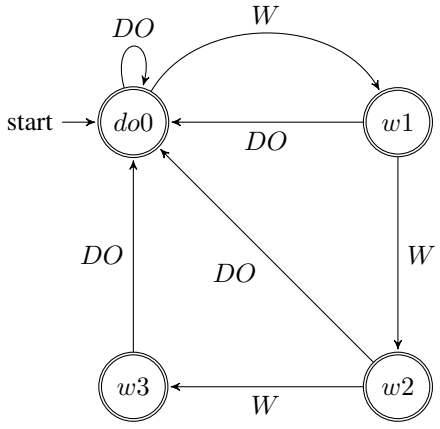
2) Shift Rotation: For every shift, t, was obtained the lists of shifts that can follow t, $T_{NextDay}$. Then for every day d, if $S_{n,d} = t$, then $S_{n,d+1}$ must be in $T_{NextDay}$.

3) Maximum Number of Shifts: For every nurse schedule was obtained the cardinality of every shift. Then the sum of the cardinality of all shifts is constrained to be equal or less than the specified.

4) Maximum Total Minutes: For every nurse schedule was obtained the cardinality of every shift. Then the duration of each shift was multiplicated by the cardinality of each shift and that value had to be equal or less than the specified Maximum Total Minutes.

5) Minimum Total Minutes: This constrain is similar to the previous, expect the value calculated had to be more or less than the specified Minimum Total Minutes.

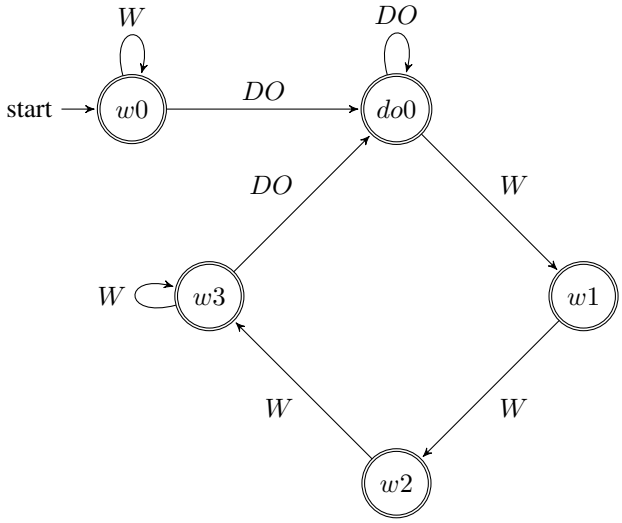
6) Maximum Consecutive Shifts: To add this constrain, an automaton was applied to the schedule of every nurse. This constrain assumes that the last day of the previous planning period was a day off. The first node acts a reset node, then it has w nodes according to the number of Maximum Consecutive Shifts. An example when the number of maximum consective days is three can be seen below.



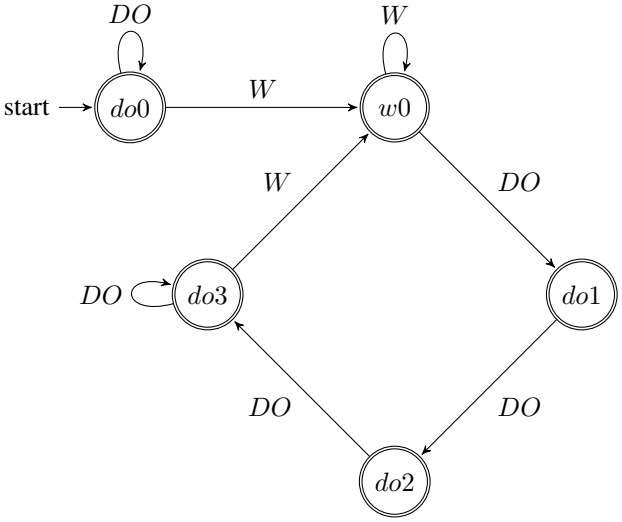
where:

do is a day off
w is a working day

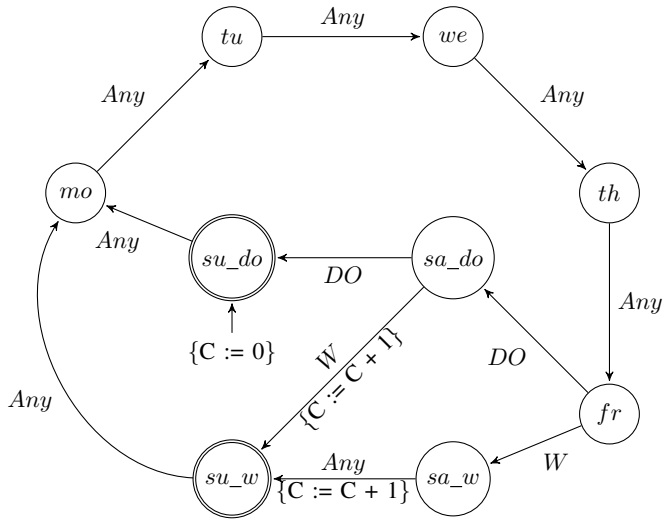
7) Minimum Consecutive Shifts: To add this constrain, an automaton was applied to the schedule of every nurse. Since this constraint always assumes that the last day of the previous planning period was a day off, it is assumed that on the beginning of this planning period the nurse already worked the minimum consecutive shifts. An example when the number of Minimum Consecutive Shifts is three can be seen below.



8) Minimum Consecutive Days Off: This constrain is similar to the previous, with roles inverted for shifts and days off.



9) Maximum Number of Weekends: To add this constrain, an automaton was applied to the schedule of every nurse. It is used a counter to count the number of working weekends. This counter is incremented whether the nurse works on Saturday or Sunday.



where:

Any is either a day off or a working day
C is the Counter being incremented
mo is monday
tu is tuesday
we is wednesday
th is thursday
fr is friday
sa_do is non-working saturday
sa_w is working saturday
su_do is non-working sunday
su_w is working sunday

10) Days off: If a nurse n requests a day d off then $S_{n,d}$ must be equal to 0.

C. Soft Constrains

1) Shift On Requests: For every shift requested, if it is not satisfied then a penalty is added to the total penalty, if it is satisfied the total penalty remains unchanged.

2) Shift Off Requests: This constrain is similar to the previous, expect the nurses request days off.

3) Cover: The matrix $S_{n,d}$ is transposed to $S_{d,n}$ in order to be able to get the cardinality of every shift for every day. If that cardinality is lower or higher than specified, the amount of violated coverage is multiplied by a weight and added to the Total Penalty.

V. Experiments and Results

A. Search Options

After applying the constrains it is necessary to search for solutions. SICStus provides multiple search options. To find the optimal search options several 20 minutes tests were performed. The options are divided into three groups.

1) Options that control the order in which the next variable is selected for assignment:

leftmost : The leftmost variable is selected
min : The leftmost variable with the smallest lower bound is selected.
max : The leftmost variable with the greatest upper bound is selected.
ff : The first-fail principle is used: the leftmost variable with the smallest domain is selected.
ffc : The most constrained heuristic is used: a variable with the smallest domain is selected, breaking ties by (a) selecting the variable that has the most constraints suspended on it and (b) selecting the leftmost one.

Figure 1 reveals that *ffc* option revealed the best results until the Instance 4, with the option *ff* obtaining the best results from Instance 5. From Instance 5, *ffc* was not able to find solutions. Options *min* and *down* were also not able to find solution from Instance 7 onwards, with two exceptions in Instance 9 and 10.

2) Options that control the way in which choices are made for the selected variable X:

step : Makes a binary choice between X #= B and X #\= B, where B is the lower or upper bound of X.
enum : Makes a multiple choice for X corresponding to the values in its domain.
bisect : Makes a binary choice between X #=< M and X #> M, where M is the middle of the domain of X, i.e. the mean of min(X) and max(X) rounded down to the nearest integer. This strategy is also known as domain splitting.

Figure 2 reveals that all search options have the same results, therefore the default option *step* will be used for future tests.

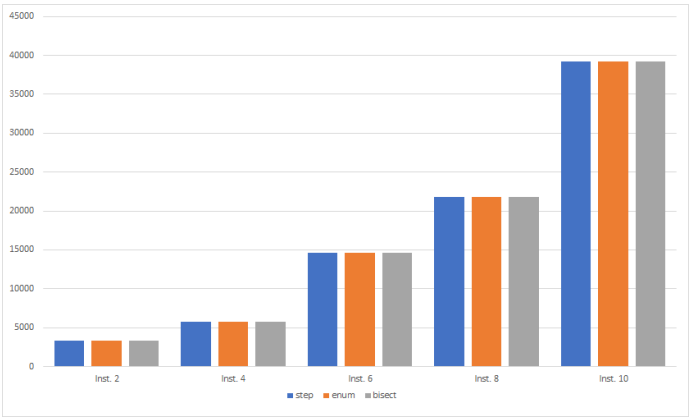


Fig. 2. Branching Strategy

3) Options that control the order in which the choices are made for the selected variable X:

up : The domain is explored in ascending order. This is the default.
down : The domain is explored in descending order.

Figure 3 reveals that *down* option revealed significantly better results in instances 2 and 4. With the search option *up* it was not even possible to find a solution for instances 8 and 12. The reason for this behaviour is because if the option *down* it tries to assign shifts first and only then it tries to assign a day off, while the option *up* tries to assign first a day off.

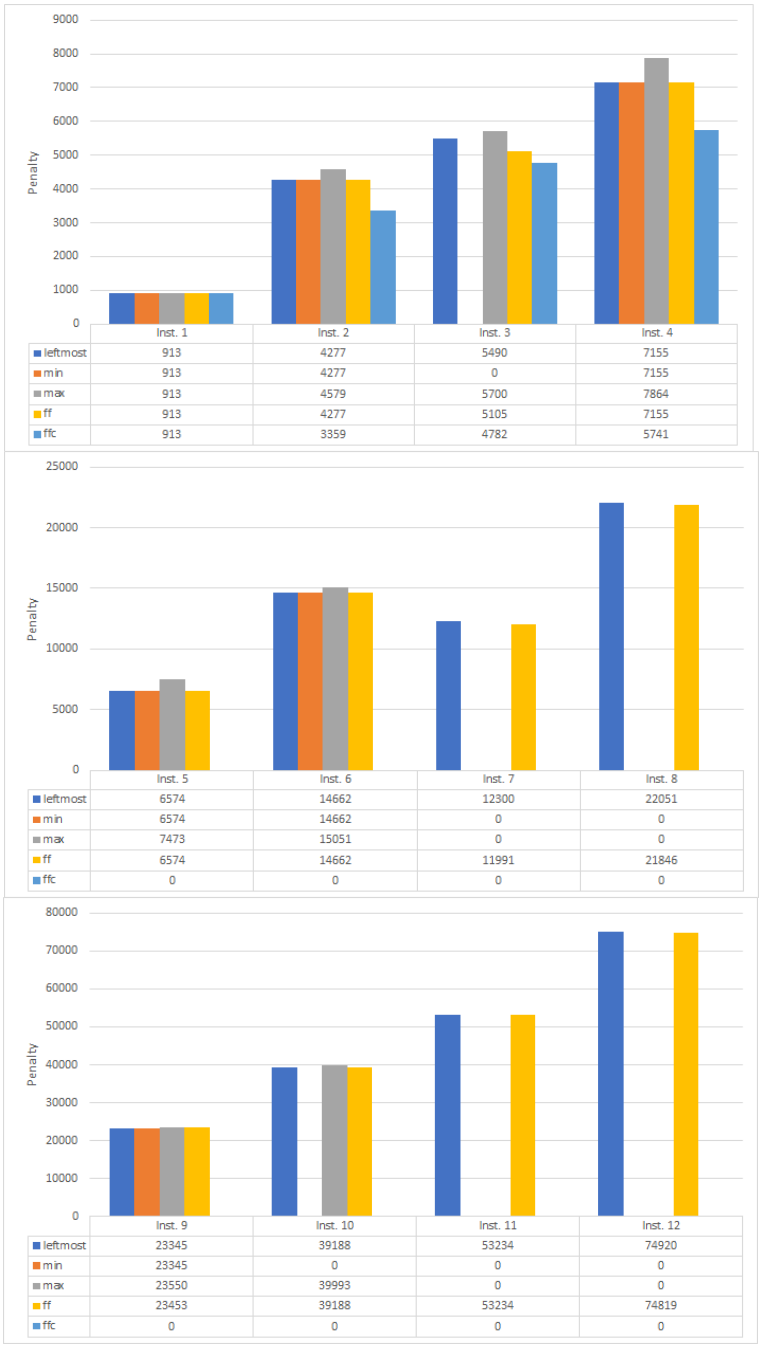


Fig. 1. Variable Selection Strategy

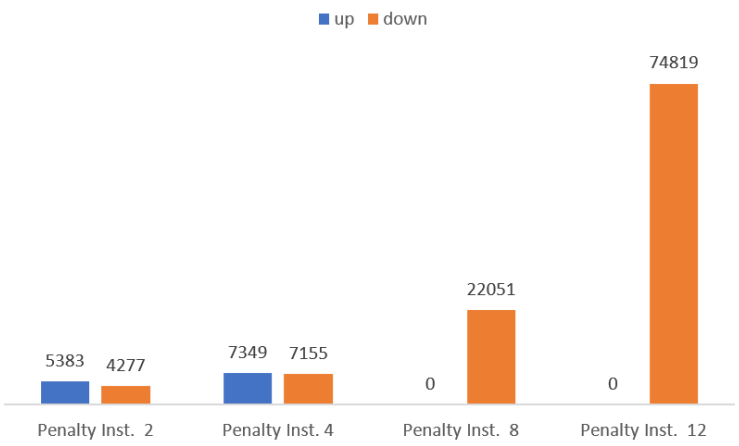


Fig. 3. Variable Order Strategy

B. Results

To evaluate the results were perform 10 minutes and 1 hour tests. The flags used were [ffc,step,down] for the first 4 instances and [ff,step,down] for the remainder. For every tests was saved the value of the penalty, as well as, several execution statistics. The full results can be seen in tables II and III. The statistics recorded were:

- resumptions - The number of times a constraint was resumed.
- entailments - The number of times a (dis)entailment was detected by a constraint.
- prunings - The number of times a domain was pruned.
- backtracks - The number of times a contradiction was found by a domain being wiped out, or by a global constraint signalling failure.
- constraints - The number of constraints created.

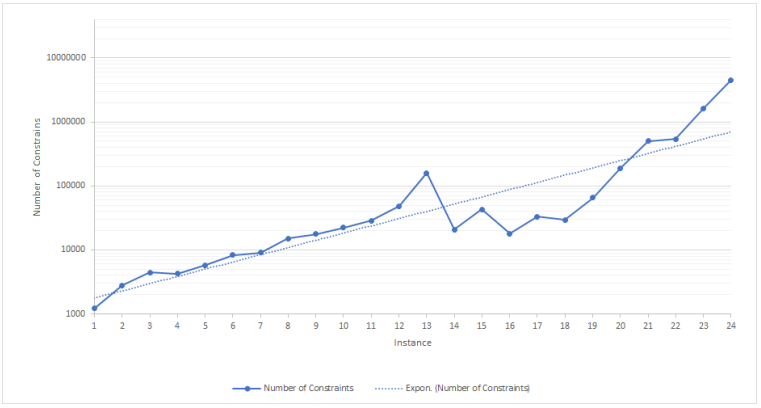


Fig. 4. Constrains per Instance

Figure 4 shows the number of constrain per Instance. Since the scale y-axis is logarithm, the chart appears to have an exponential growth in the number of constrains applied between instances 1 and 12 and 18 and 24. The peak in complexity at instance 13 is due to a high number of staff and shift compared to the previous and next instances.

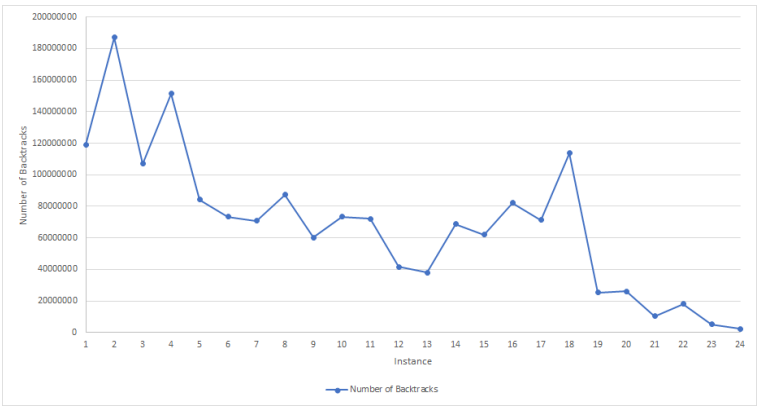


Fig. 5. Backtracks per Instance

Figure 5 shows the number of backtracks for the 1 hour tests. It is possible to observe that there is a tendency for the number of backtracks to decrease with the increase of complexity. This happens because, since the search space is higher, the generator takes more time to find a contradiction by the current domain.

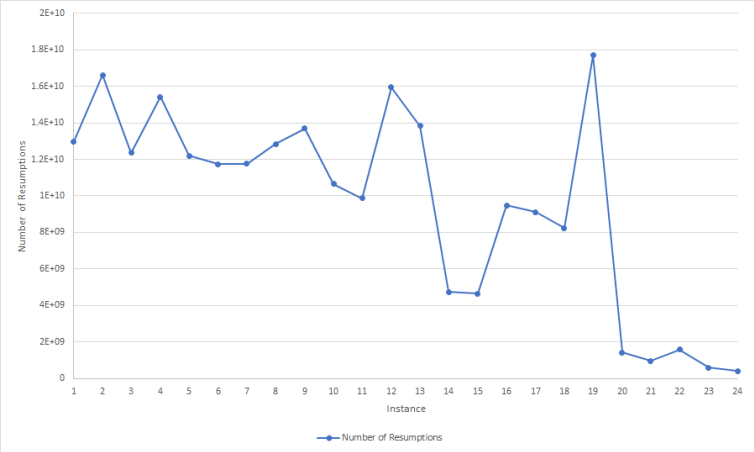


Fig. 6. Resumptions per Instance

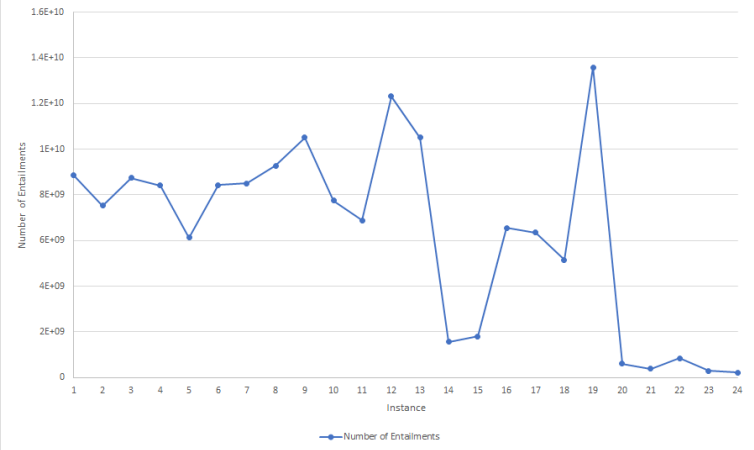


Fig. 7. Entailments per Instance

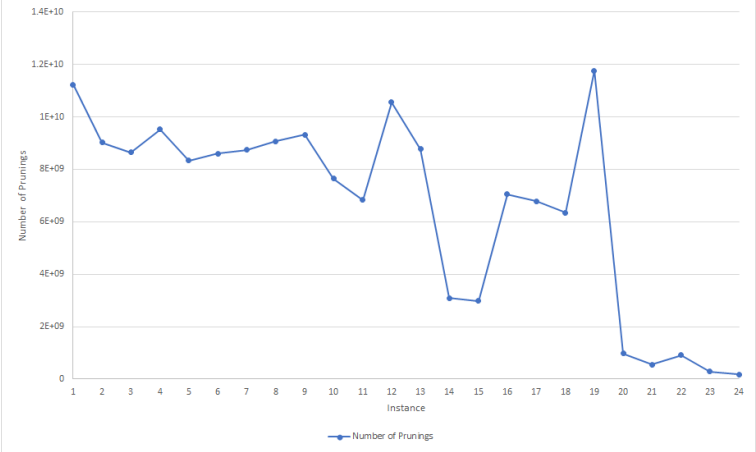


Fig. 8. Prunings per Instance

Figures 6, 7, 8 show the number of resumptions, entailments and prunings for the 1 hour tests, respectively. It is important to denote that the instances with lower values (Instances 14, 15, 20, 21, 22, 23, 24) were the ones that it wasn't possible to find a feasible solution.

Table I presents the comparison of the proposed solution with other results published by other researchers. The Ejection Chain and Gurobi tests were limited to a maximum time of 1 hour. Branch and Price results were all obtained in less than 1 hour except for instance 8 which took 3 hours.

The presented approach obtained underwhelming results being surpassed by other approaches. The Gurobi solver has generally the best results, but it could not solve the 5 more complex problems. The Branch and Price Algorithm had excellent results for simple instances, however it couldn't find solutions for almost half the instances. Ejection chain even though it does not produce the best results it can find a solution for every instance. The CLP approach couldn't find a solution for 7 instances and it got much worse solutions than the other algorithms. Finally, the difference between running 10 or 60 minutes is residual, confirming that CLP approaches are good at find any solutions, but not at finding the best solution available.

VI. Conclusions and Future Work

The presented paper aimed to present an alternative solution to the current NRP. This solution differs from current heavily studied integer programming solutions by using a different paradigm, CLP. However the results were underwhelming compared to other solutions. Nevertheless some conclusions can be drawn: CLP approach is good for finding any solution, but not at finding the best solution; the search options have a great influence in the end result. There are options for future research such as the optimization of some constrains, namely the shift rotation constrain could be applied by using an automaton; the development of a test suite to better compare different search options; it would also be interesting to investigate if it would be possible to break symmetries in our model.

REFERENCES

[1] Shift Scheduling Benchmark Data Sets. Nurse Rostering Benchmark Instances. Available at http://www.schedulingbenchmarks.org/nrp/

[2] https://www.staffrostersolutions.com/downloads.php

[3] Carlsson M., Ottosson G., Carlson B. An Open-Ended Finite Domain Constraint Solver, Proc. Programming Languages: Implementations, Logics, and Programs, 1997.

[4] Burke, Edmund & Curtois, Timothy. (2014). New approaches to nurse rostering benchmark instances. European Journal of Operational Research. 237. 71–81. 10.1016/j.ejor.2014.01.039.

[5] Gurobi Optimization, Inc. Gurobi Optimizer Reference Manual. 2014; Available from: http://www.gurobi.com.

[6] Demirović, E., Musliu, N. & Winter, F. Modeling and solving staff scheduling with partial weighted maxSAT. Ann Oper Res 275, 79–99 (2019). https://doi.org/10.1007/s10479-017-2693-y

TABLE I
COMPARISON OF RESULTS

Inst.	Weeks	Staff	Shifts	Branch and Price [4]	Ejection Chain [4]	Gurobi [5]	CLP 10 min	CLP 60 min
1	2	8	1	607	607	607	913	812
2	2	14	2	828	837	828	3560	3359
3	2	20	3	1001	1003	1001	4782	4480
4	4	10	2	1716	1718	1716	5942	5640
5	4	16	2	1160	1358	1143	6574	6574
6	4	18	3	1952	2258	1950	14761	14662
7	4	20	3	1058	1269	1056	12092	11991
8	4	30	4	1308	2260	1323	21846	21745
9	4	36	4	439	463	439	23552	23351
10	4	40	5	4631	4797	4631	39188	39087
11	4	50	6	3443	3661	3443	53335	53134
12	4	60	10	4046	5211	4040	74920	74819
13	4	120	18	-	3037	3109	137748	137748
14	6	32	4	-	1847	1280	-	-
15	6	45	6	-	5935	4964	-	-
16	8	20	3	3223	4048	3323	34555	34251
17	8	32	4	-	7835	5851	58497	58191
18	12	22	3	-	6404	4760	56178	55876
19	12	40	5	-	5531	5420	80532	80431
20	26	50	6	-	9750	-	-	-
21	26	100	8	-	36688	-	-	-
22	52	50	10	-	516686	-	-	-
23	52	100	16	-	54384	-	-	-
24	52	150	32	-	156858	-	-	-

TABLE II
RESULTS OF 10 MINUTES TESTS

intance	penalty	resumptions	entailments	prunings	backtracks	constraints
1	913	2165555049	1479286767	1876692016	19960738	1220
2	3560	2709614755	1206667754	1468704770	30208180	2787
3	4782	2024797589	1432015461	1413518421	17595431	4422
4	5942	2596659221	1382279858	1590578334	26093821	4234
5	6574	2041806991	1015064351	1390366139	14200008	5746
6	14761	1965615953	1417178651	1440965817	12240895	8345
7	12092	2035151060	1487903337	1506072698	11905839	9138
8	21846	2188312613	1588526116	1538526966	14135727	15195
9	23552	2364957377	1838138473	1609894198	9374862	17842
10	39188	1823312781	1348535571	1306804194	12159826	22315
11	53335	1617354294	1121545428	1122853582	12302270	28582
12	74920	2375950759	1793759958	1554872157	7646706	48430
13	137748	2347395039	1786108691	1483857596	6138289	158700
14	-	826426045	294193833	557061504	11661403	20676
15	-	831544657	306204963	516786972	9437087	42942
16	34555	1600538857	1106493906	1185209809	13691961	18061
17	58497	1446142368	970230440	1067002418	10940601	33197
18	56178	1358001775	843531117	1039212678	17748613	29332
19	80532	2930415093	2247493735	1944155773	4632067	65526
20	-	248754582	104782637	167363846	4461616	188991
21	-	156074708	59778986	90998129	1605041	501794
22	-	281115443	143931580	162657119	3114482	543925
23	-	122677154	52647736	60512591	825403	1626729
24	-	120993500	47579287	54673696	388285	4554857

TABLE III
RESULTS OF 1 HOUR TESTS

Intance	Penalty	Resumptions	Entailments	Prunings	Backtracks	Constraints
1	812	12973551454	8863065460	11229164894	119117917	1220
2	3359	16605635306	7524712562	9022481461	187002694	2787
3	4480	12352977041	8742904370	8640616155	106870201	4422
4	5640	15416681297	8406950814	9518791520	151488364	4234
5	6574	12203027523	6126493391	8331764491	84149363	5746
6	14662	11740461899	8428617306	8594762143	73413363	8345
7	11991	11760016300	8511179674	8738334331	70828526	9138
8	21745	12836695964	9284915598	9064228907	87408445	15195
9	23351	13698709229	10508920131	9320098466	60148281	17842
10	39087	10651609732	7744407809	7642336115	73458821	22315
11	53134	9883269474	6890692609	6831294568	72203220	28582
12	74819	15949010642	12323425132	10553205354	41630530	48430
13	137748	13831380409	10516195108	8769451967	38094266	158700
14	-	4739412564	1559764054	3097093319	68647954	20676
15	-	4641817560	1801093007	2984958092	61954145	42942
16	34251	9475017762	6549982690	7047591067	82220457	18061
17	58191	9116562661	6343588993	6785509494	71284453	33197
18	55876	8243685979	5148266082	6343792260	113799307	29332
19	80431	17689541034	13583622742	11766603627	25437682	65526
20	-	1437932070	602877819	973784958	26111034	188991
21	-	962365875	380341597	562430796	10285992	501794
22	-	1582594506	842013860	919608904	18096577	543925
23	-	592611919	300980221	289179135	5097465	1626729
24	-	412151051	215631457	181825265	2418408	4554857