Constraint Logic Programming to solve the Nurse Rostering Problem

Gil Teixeira

MPE 2019/20

Context

- Staff scheduling problems are found in:
  - Transportation (airlines, train and bus drivers, airports);
  - Emergency services (police, fire and ambulance crews);
  - Factories;
  - Healthcare (physicians and nurses);
  - Retail.
- There is **no known algorithm that can guarantee optimal solutions.**
- 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.

Nurse Rostering Problem

- The **shift scheduling problem** is a widely occurring, difficult optimization problem. Nurse rostering problems are encountered in hospitals all over the world.
- The Nurse Rostering Problem (**NRP**) is defined as assigning a number of nurses to different shifts during a specified planning period, considering some regulations and preferences.
- When tackling this problem in real-world settings, the constraints of the problem are often classified as **hard** and **soft** constraints.

Problem description

- Let **D = {1, ..., |D|}** be a scheduling period consisting of **|D|** consecutive days.
- Given a set of nurses **N = {1, ..., |N|}** and a set of shifts **S = {1, ..., |S|}**, the goal is to find an assignment of shifts to nurses for every day.
- Each shift **s** has a duration **ls**, expressed in minutes and the types of shift that can follow.
- Each nurse has preference of days and shifts that they want or don't want to do.

The Dataset

- The benchmark data set captures the core features of the problem.
- Instances of varying planning length, numbers of nurses and shift types have been produced.
  - There are instances from 2 to 52 weeks.
  - The number of staff varies from 8 to 150
  - The number of different shift types varies from 1 up to 32 in the largest instances.
  - There are 24 instances in total.
  - The instances are available at http://www.schedulingbenchmarks.org/nrp/.

Literature approaches

- **Metaheuristics:**
  - Burke, E. K., Curtois, T., Qu, R., & Berghe, G. V. (2010). A scatter search methodology for the nurse rostering problem. Journal of Operational Research Society, 61, 1667–1679.
- **Mathematical programming:**
  - Bard, J. F., & Purnomo, H. W. (2005). Preference scheduling for nurses using column generation. European Journal of Operational Research, 164(2), 510–534.
- **Artificial intelligence techniques:**
  - Beddoe, G. R., & Petrovic, S. (2007). Enhancing case-based reasoning for personnel rostering with selected tabu search concepts. Journal of the Operational Research Society, 58(12), 1586–1598.

My Solution - CLP

- Constraint logic programming is a merge between: Constraint programming (CP) and Logic programming (LP). CLP is mostly used in Search and Optimization problems, such the NRP.
- Implementation:
  - 1st Step – Parse Text files to Prolog facts.
  - 2nd Step – Use **Constraint Logic Programming** to reach a solution.
  - 3rd Step – Generate XML with obtained solution
  - 4th Step – Validate solution using **RosterViewer**.

Domain

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

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

- For every nurse, for every day there is a variable from 0 (no shift that day) to the total number of different shift types.

Hard Constrains of the problem

- HC1 : **Maximum one shift per day** – nurses cannot be assigned more than one shift on a day.

- Constrain already defined by the domain.

Hard Constrains of the problem

- HC2 : **Shift rotations** – A minimum amount of rest is required after each shift. Therefore certain shifts cannot follow others.

- For every shift, $\boldsymbol{t}$, was obtained the lists of shifts that cannot follow **t**, $\boldsymbol{T_{NotNextDay}}$. Then for everyday: ($\boldsymbol{s_{n,d}}$ #= $\boldsymbol{t}$) => $\boldsymbol{s_{n,d+1}}$ not in $\boldsymbol{T_{NotNextDay}}$

Hard Constrains of the problem

- HC3: **Maximum number of shifts** – the maximum number of shift types that can be assigned to each nurse within the planning period.

- 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 Maximum number of shifts.

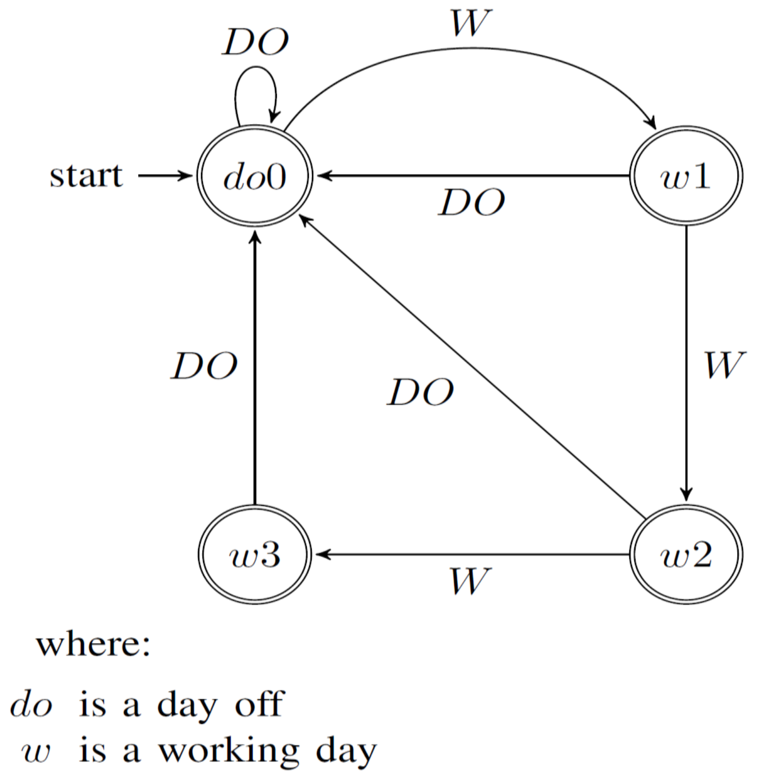
Hard Constrains of the problem

- 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.

- 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 in the specified interval.

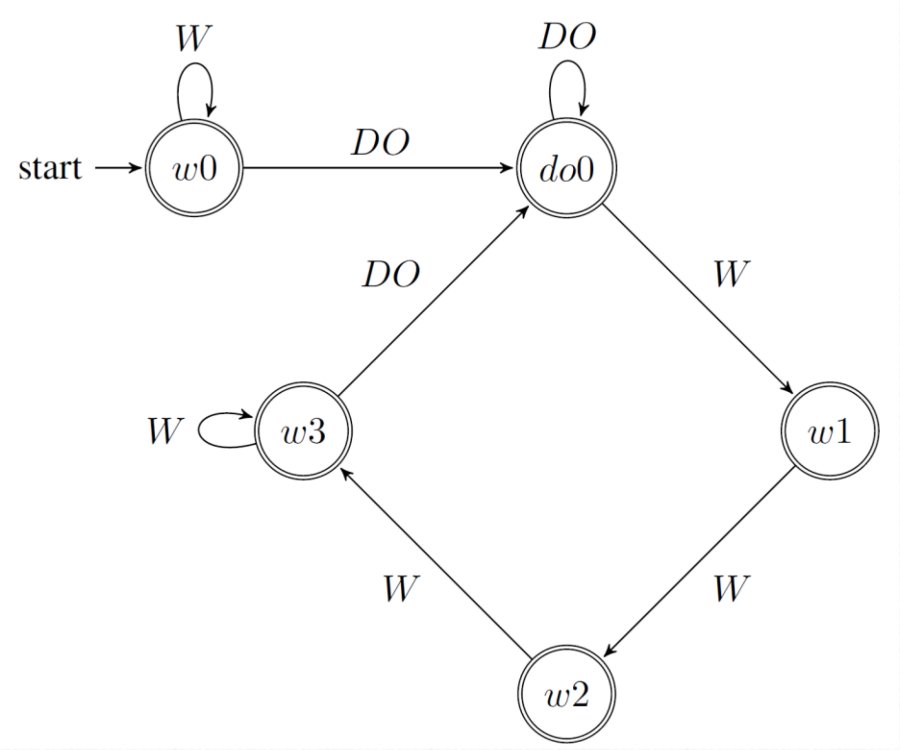
Hard Constrains of the problem

- HC6 : **Maximum consecutive shifts** – the maximum number of consecutive shifts, which can be worked within the planning period.



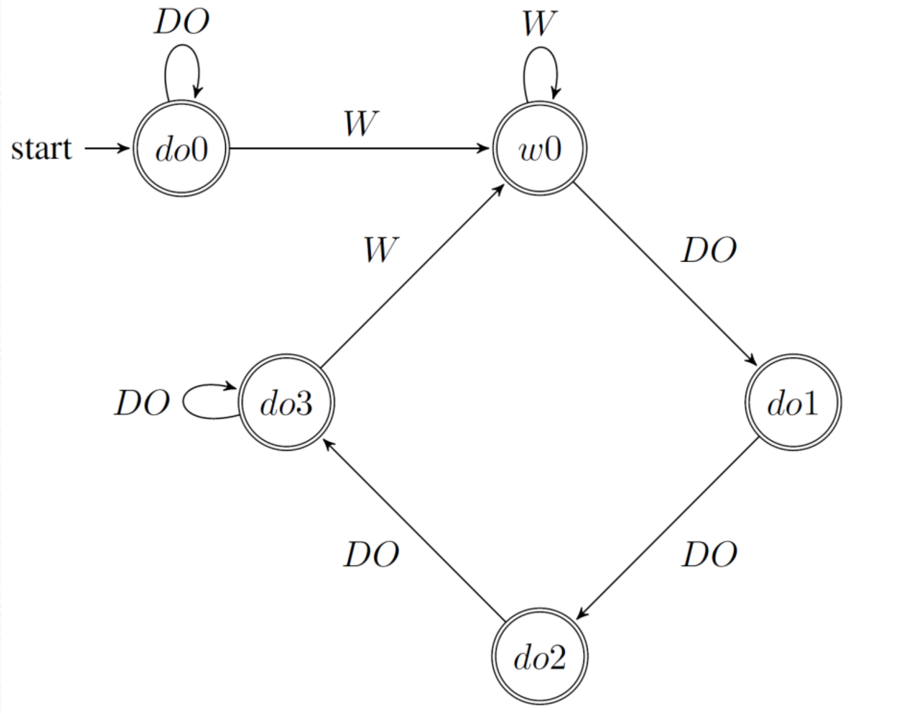
Hard Constrains of the problem

- HC7 : **Minimum consecutive shifts** – the minimum number of consecutive shifts, which can be worked within the planning period.



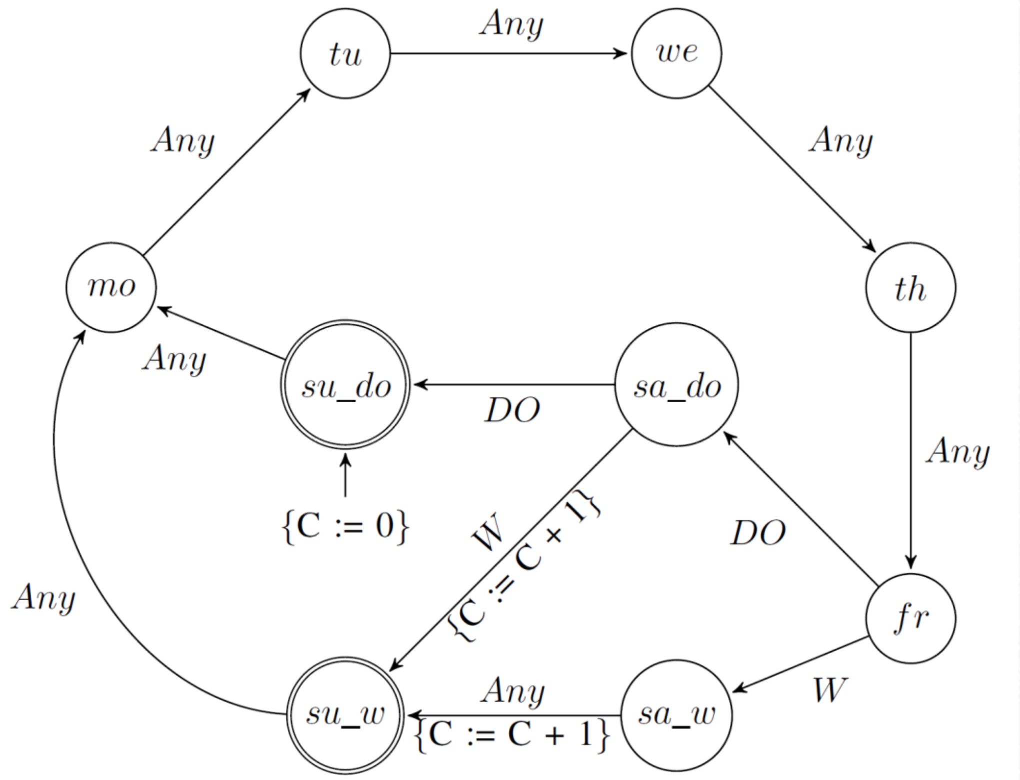
Hard Constrains of the problem

- HC8 : **Minimum consecutive days off** – the minimum number of consecutive days off, which can be assigned within the planning period.



Hard Constrains of the problem

- 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.



Hard Constrains of the problem

- HC10 : **Requested days off** – shifts must not be assigned to a specified nurse on some specified days.

- If a nurse n requests a day, ***d*** , off then $\boldsymbol{s_{n,d}}$ must be equal to 0.

Soft Constrains of the problem

- 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.

- For every shift on and shift off requested, if it is not satisfied then a penalty is added to the total penalty, if it is satisfied the total penalty remains unchanged.

Soft Constrains of the problem

- SC3 : **Coverage** – the required number of nurses assigned to a specified day for a specified shift should be within a range. 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.

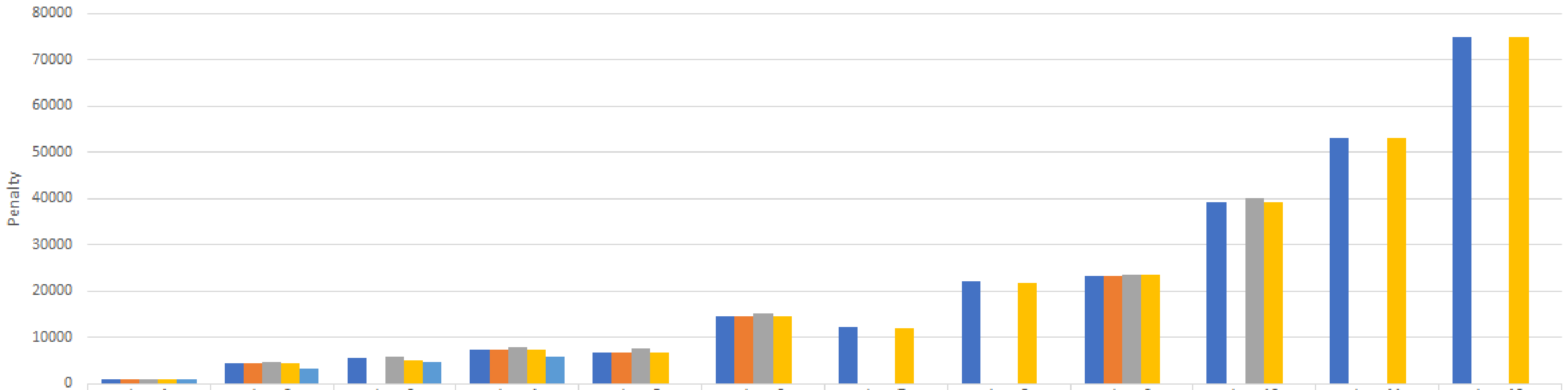
- The matrix $\boldsymbol{S_{n,d}}$ is transposed to $\mathbf{S_{d,n}}$ in order to be able to get the cardinality of every shift for every day. If that cardinality is lower or higher than specified, the amount of violated coverage is multiplied by a weight and added to the Total Penalty.

Search Options

- After applying the constrains it is necessary to search for solutions. SICStus provides multiple search options. The search strategy can have great impact on the efficiency of the final solver. To find the optimal search options several 20 minutes tests were performed.

Search Options

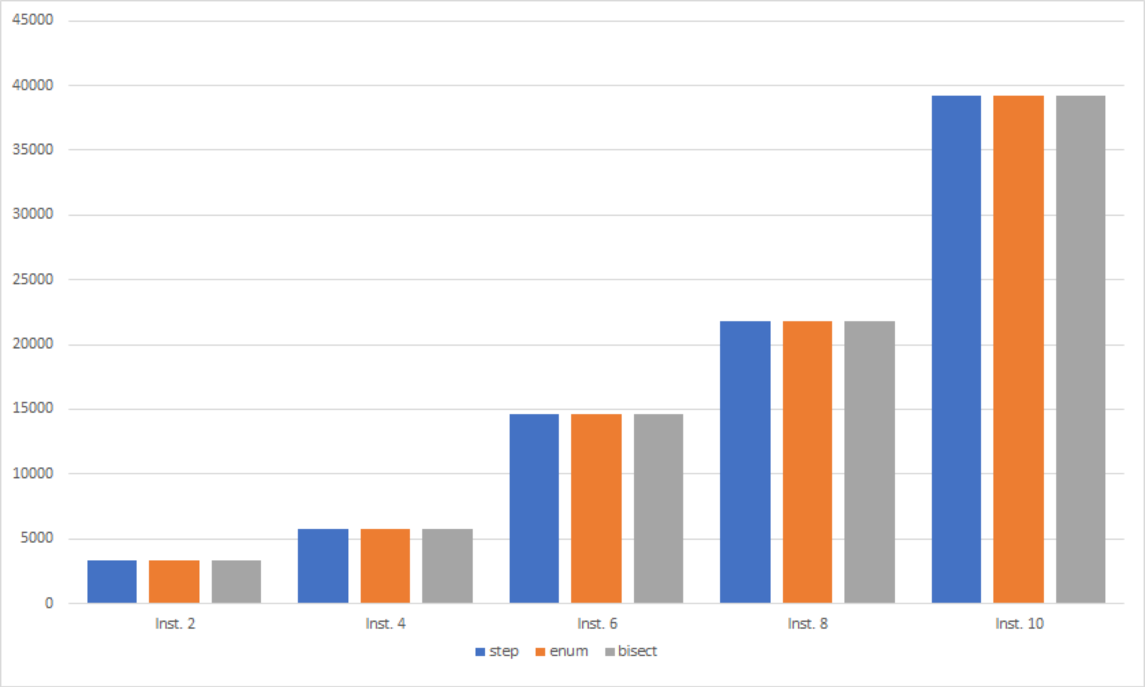
- 1) Options that control the order in which the next variable is selected for assignment:
  - **ff** : The first-fail principle is used: the leftmost variable with the smallest domain is selected.
  - **ffc** : The most constrained heuristic is used



	Inst. 1	Inst. 2	Inst. 3	Inst. 4	Inst. 5	Inst. 6	Inst. 7	Inst. 8	Inst. 9	Inst. 10	Inst. 11	Inst. 12
leftmost	913	4277	5490	7155	6574	14662	12300	22051	23345	39188	53234	74920
min	913	4277	0	7155	6574	14662	0	0	23345	0	0	0
max	913	4579	5700	7864	7473	15051	0	0	23550	39993	0	0
ff	913	4277	5105	7155	6574	14662	11991	21846	23453	39188	53234	74819
ffc	913	3359	4782	5741	0	0	0	0	0	0	0	0

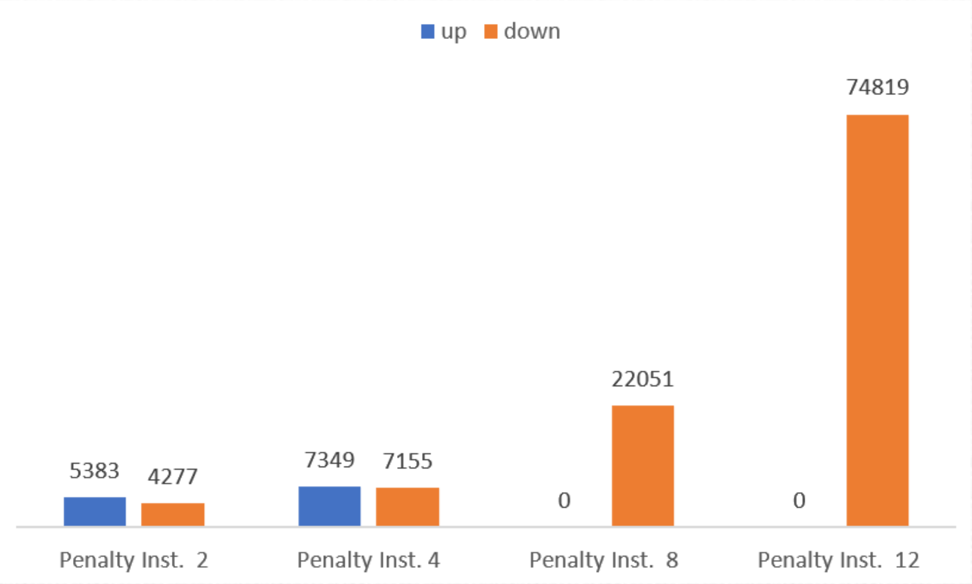
Search Options

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



Search Options

- 3) Options that control the order in which the choices are made for the selected variable X:
  - **down** : The domain is explored in descending order.

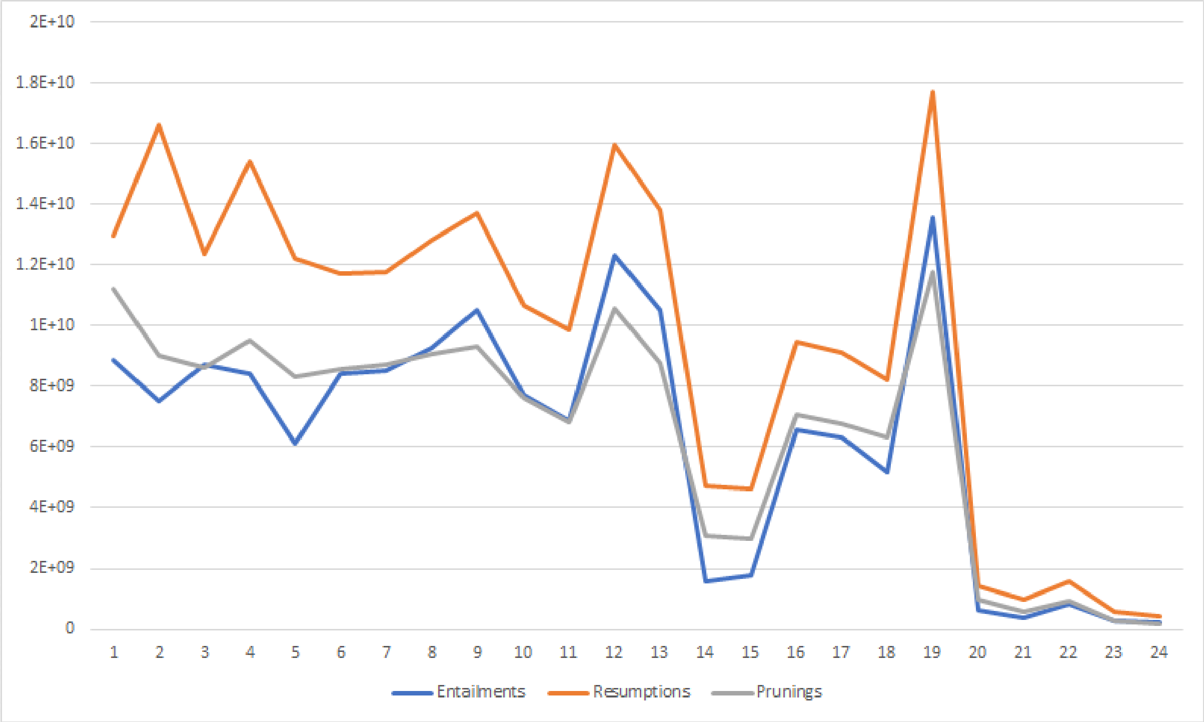


Results

- To evaluate the results were perform 10 minutes and 1-hour tests.
- For every tests was saved the value of the penalty, as well as, several execution statistics:
  - **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.

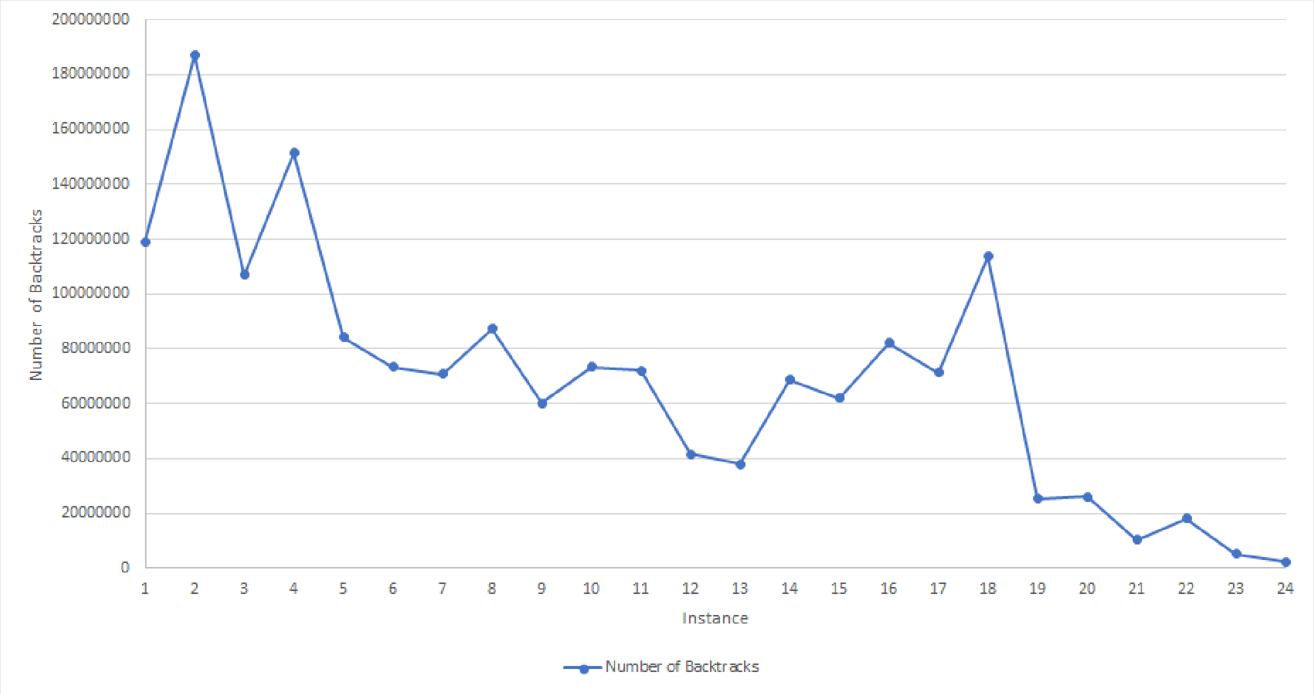
Results

- It is important to denote that the instances with lower values were the ones that it wasn't possible to find a feasible solution.



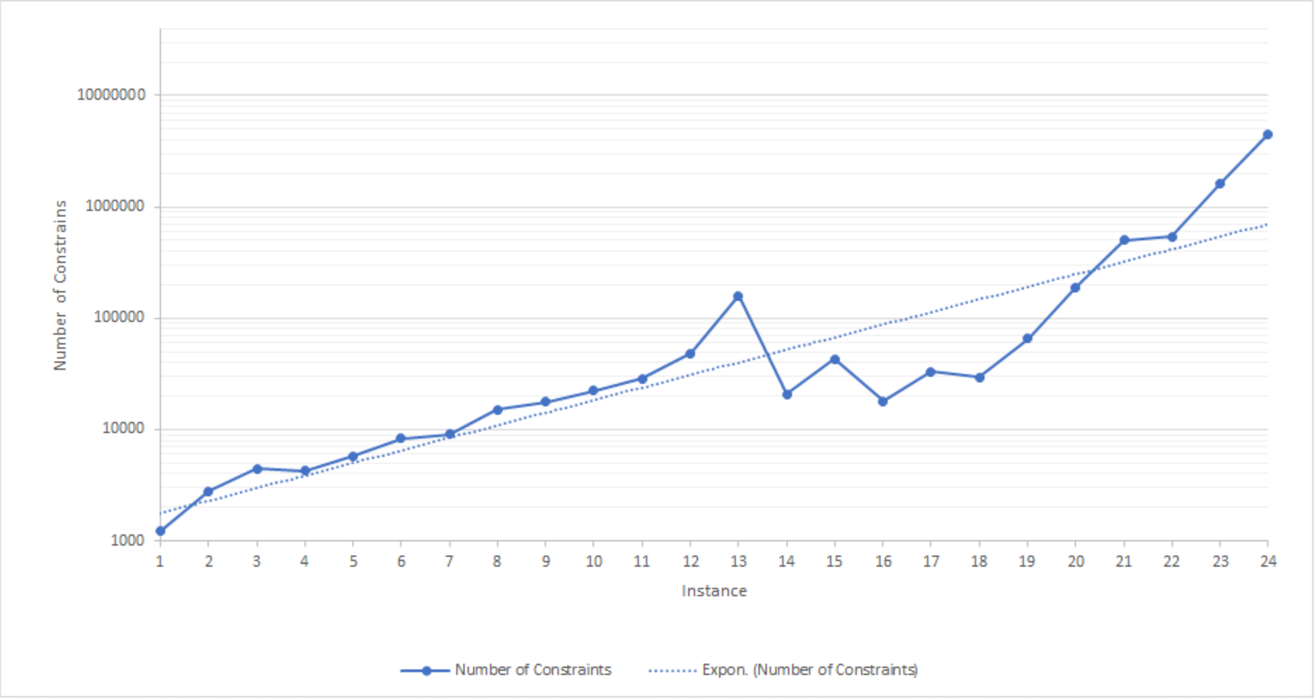
Results - Backtracks

- It is possible to observe that there is a tendency for the number of backtracks to decrease with the increase of complexity.



Results - Constrains

- 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.



Results - Comparison

- The presented approach obtained underwhelming results being surpassed by other approaches.
- The CLP approach couldn't find a solution for 7 instances and it got much worse solutions than the other algorithms.
- The difference between running 10 or 60 minutes is residual.

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

Conclusions & Future Work

- The presented work aimed to present an alternative solution to the current NRP.
- The results were underwhelming compared to other solutions.
- CLP approach is good for finding any solution, but not at finding the best solution.
- The search options have a great influence in the final result.
- There are options for future research:
  - development of a test suite to better compare different search options;
  - investigate if it would be possible to break symmetries in our model.

References

- Schedulingbenchmarks.org. 2020. *Nurse Rostering Benchmark Instances*. [online] Available at: <http://www.schedulingbenchmarks.org/nurse.html> [Accessed 22 April 2020].
- Smet, P. (2018). Constraint reformulation for nurse rostering problems. In Proceedings of the 12th international conference of the practice and theory of automated timetabling (pp 69–80).
- Curtois, T., & Qu, R. (2014). Computational results on new staff scheduling benchmark instances. Technical report, ASAP Research Group, School of Computer Science, University of Nottingham, NG8 1BB, Nottingham, UK

Any Questions?