CS4210-A: Algorithms for Intelligent Decision Making
Part I: SAT and Constraint Programming
Modelling Assignment

E. Demirović e.demirovic@tudelft.nl
N. Yorke-Smith n.yorke-smith@tudelft.nl

February 2022

This assignment is required. You may perform it in groups of two students. Assignment prepared by N. Yorke-Smith and E. Demirović.

Read This First

Read the source code of the relevant demo report on Brightspace before tackling this assignment: it contains problem-independent instructions on what to hand in, as well as useful problem-independent indications on how to proceed. It is also strongly recommended to read the Submission Instructions and Grading Rules at the end of this document before attempting to address its tasks. Good luck!

Purpose of This Assignment

The purpose is to learn by doing, trying mathematical modelling, model improvement, and different solvers. This assignment will take some time, as it involves understanding and tackling a real-world combinatorial optimisation problem. However, it is not meant to be unduly time-consuming. We would rather read about what you tried and learned, rather than you feel stuck because you cannot solve difficult instances of the problem to optimality.

Your Task

Consider the Rotating Workforce Scheduling (RWS) problem [1]. Your task is to build a constraint programming model for this problem in MiniZinc, and solve it on current hardware with current solving technology. For this, we have provided 20 instances on which you can test your solutions. You will need to carefully read and understand the problem specification and other information in the paper linked.

We start off with the RWS problem without extensions, which is a satisfiability problem. Please perform the following steps:

A. Design and evaluate a CP model for the RWS problem.

B. Design and evaluate a second CP model for the RWS problem: for example, with a different viewpoint.

C. Experiment with at least two different solving backends and your models, on the data instances provided with this assignment.

Kletzander et al. [1] then extend the RWS problem with constraints to check for unsatisfiability and two optimization conditions: maximizing the number of free weekends and optimizing the distribution of weekends. Your goal next goal is as follows:

D. Solve the data instances for optimality by **maximizing the number of free weekends**. For this, use a timeout of 3600 seconds and present the optimization in a figure similar to Figure 2 in the paper. Do this for both your models you developed in A. and B. above.

After performing all the experiments, answer the following sequence of questions:

E. What was your modelling strategy or strategies?

F. Which improvements of your models have you performed to render them more efficient? Quantify the effect of each improvement, without necessarily running all versions of a model on all the instances under all the chosen backends: use your best judgement.

G. For each of the chosen technologies and backends, which model is more efficient?

H. Which combination of model, technology, and backend would you recommend to the manager for solving future instances of the problem? Why? Factor in your answer to Question E, as one may also want to consider the maintainability of the chosen model.

Output Instructions

Your model should present the output as follows: on the first line, print the working schedule solution found by your model as an array, followed by the number of free weekends of that schedule. Moreover, the output array should be shifted with regard to the offset in such a way that the first element corresponds to a Monday (e.g. a solution with offset = 4 should be shifted 4 places to the right). For example, a working schedule for four employees and three shift types (with 1=Day, 2=Afternoon, 3=Night and -=Day off) should be output as:

```
[1,1,1,1,3,3,-,-,-,2,2,2,2,3,3,3,-,-,1,1,1,2,2,3,3,-,-,-]
1
```

Submission Instructions

All task answers, other than source code, must be in a single report in PDF format. Further:

- Identify the team members and state the group number inside the report. Your group number is assigned when you self-enroll to a group under category 'Assignment groups' on Brightspace.
- Address each task of each problem, using the numbering and the ordering in which they appear in this assignment statement.
- Take the instructions of the relevant demo report on Brightspace as a strict guideline for the structure and content of a model description, model evaluation, and task answer in the report, as well as an indication of the expected quality of the report.
- Write clear task answers, source code, and comments.
- Justify all task answers, except where explicitly not required.
- State any assumptions you make that are not in this document.
- Please thoroughly proof-read, spell-check, and grammar-check your report.
- Please adhere exactly to the output instructions. Your solutions will be checked automatically.
- Upload all model files to Brightspace in a single ZIP file without folder structure.
- Name one MiniZinc model file (within your ZIP) as `CSP_solution.mzn`: this is the file we will run at our discretion (see below).
- Match exactly the uppercase, lowercase, and layout conventions of any filenames and I/O texts imposed by the tasks, as we will process your models automatically.
- Write a paragraph, which will not be graded, describing your experience with this assignment: Which aspects were too difficult or too easy? Which aspects were interesting or boring? This will help us improve the course in the coming years.
- Remember that when submitting you implicitly certify (a) that your report and all its uploaded attachments were produced solely by your team, except where explicitly stated otherwise and clearly referenced, (b) that each teammate can individually explain any part starting from the moment of submitting your report, and (c) that your report and attachments are not freely accessible on a public repository.

Please submit two files on Brightspace: one PDF file which is your report, and one ZIP file which contains (in the root of the ZIP file) all your model files. Submit once only, as a group on Brightspace. For any Brightspace questions, please contact `brightspace@tudelft.nl`.

Grading Rubric

If all tasks have been seriously attempted, and all requested models exist in files with the imposed names and the comments exemplified in the relevant demo report, and your models produce correct outputs for some instances in reasonable time, then you score at least 5 points (read on), otherwise your final score is 0 points. Further:

- If you have designed two different correct CP models for the RWS problem for two different solvers, you score 1 more point.
- If one of your models with one solving backend can find a solution for most of the instances within reasonable time, you score 1 more point.
- If one of your models with one solving backend can solve to optimality for most of the instances within reasonable time, you score 1 more point.
- Base on the quality of the report, you can score 2 more points.
- If the Teaching Assistant figures out a minor fix that is needed to make a model run as per our instructions above, then, instead of giving 0 points, the TA may deduct 2 points at her discretion.

A random sample of teams will be invited to explain their model and its development to a TA in person.

References

[1] Lucas Kletzander et al. "Exact methods for extended rotating workforce scheduling problems". In: *Proceedings of the International Conference on Automated Planning and Scheduling*. Vol. 29. 2019, pp. 519–527.