Project Work
Combinatorial Decision Making and Optimization
Module 1

Topic: Modelling and solving the present wrapping problem
Author: Zeynep Kiziltan
Date: May 29, 2020

This document describes a project work for the first module of the Combinatorial Decision Making and Optimization course, which is about modelling and solving a combinatorial decision problem with (i) Constraint Programming (CP), and (ii) propositional SATisfiability (SAT) or its extension to Satisfiability Modulo Theories (SMT). You can form a group of 2 members. You are free to propose other combinatorial decision problems, provided that you get my approval before start working on it. Good luck ☺

1 Description of the Problem

It is a common practice that a private business rewards its loyal clients with presents, which are typically wrapped in a costly corporate paper covered with the logo of the business. Imagine that you work for such a business which wants to limit the overall amount of paper that can be used for this purpose, in order to reduce the associated expenses. As the combinatorial decision and optimization expert, you are assigned to solve the Present Wrapping Problem (PWP): given a wrapping paper roll of a certain dimension and a list of presents, decide how to cut off pieces of paper so that all the presents can be wrapped. Consider that each present is described by the dimensions of the piece of paper needed to wrap it. Moreover, each necessary piece of paper cannot be rotated when cutting off, to respect the direction of the patterns in the paper.

2 Format of the Instances

This section describes the format in which the PWP instances are written, as well as the expected format of the corresponding solutions.

Instance Format An instance of PWP is a text file consisting of lines of integer values. The first line gives w and h, which are the width and the height of the paper roll. The following line gives n, which is the number of

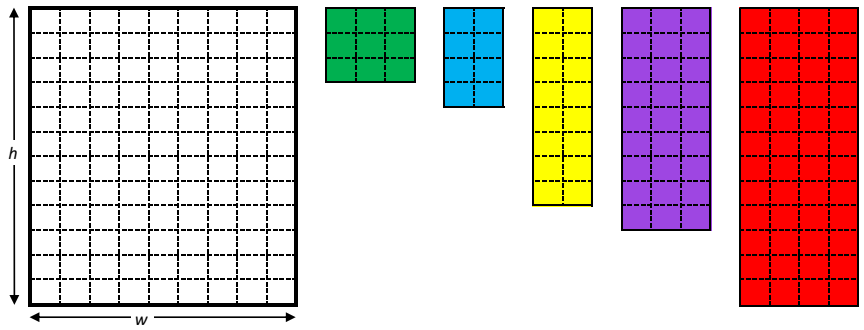

Figure 1: Graphical representation of the instance.

necessary pieces of paper to cut off. Then n lines follow, each with x_i and y_i, representing the horizontal and vertical dimensions of the i-th piece of paper. For example, a file with the following lines:

```
9 12
5
3 3
2 4
2 8
3 9
4 12
```

describes an instance in which the paper roll has the width 9 and the length 12, and we need to cut off 5 pieces of paper, with the dimensions 3×3, 2×4, 2×8, 3×9, and 4×12. Figure 1 shows the graphical representation of the instance.

Solution Format How to cut off a piece of paper i can be described by the position of i in the main paper roll. The solution should indicate the position of each i by its $\hat{x}_i$ and $\hat{y}_i$, which are the coordinates of the left-bottom corner i. This could be done by for instance adding $\hat{x}_i$ and $\hat{y}_i$ next to the x_i and y_i in the instance file. To exemplify, the solution of the instance depicted in Figure 1 could look like:

```
9 12
5
3 3   4 0
```

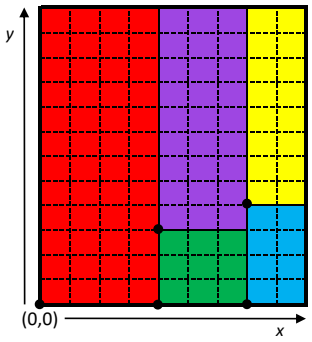

Figure 2: Graphical representation of the solution.

```
2 4   7 0
2 8   7 4
3 9   4 3
4 12  0 0
```

which says for instance that the left-bottom corner of the 3×3 piece is at $(4, 0)$. The solution can be represented graphically as in Figure 2.

3 Project Work

The purpose of this project is to solve PWP with (i) Constraint Programming (CP), and (ii) propositional SATisfiability (SAT) or its extension to Satisfiability Modulo Theories (SMT), using the MiniZinc CP solver and the Z3 SAT/SMT solver. A suite of problem instances are provided in the format specified in Section 2. You will decide yourselves the problem constraints around the problem description, and build the CP model and the SAT/SMT encoding accordingly. For correctness check, you are advised to visualize the solutions as in Figure 2. This can be done manually by hand, or automatically by a program which takes as input the solution of the problem in the format described in Section 2.

While a trivial model/encoding is a good starting point, you should come up with the **best** model/encoding to tackle the relatively more difficult instances of the problem in the most efficient way. Proceed as follows:

1. Start with the variables and the main problem constraints.

2. In any solution, if we draw a vertical line and sum the vertical sides of the traversed pieces, the sum can be at most l. A similar property holds if we draw a horizontal line. Use these implied constraints in both of your CP model and SAT/SMT encoding.

3. Use global constraints to impose the main problem constraints and the implied constraints in your CP model.

4. Investigate the best way to search for solutions in CP.

The problem requirements do not allow the rotation of the pieces. This means that, an $n \times m$ piece cannot be positioned as an $m \times n$ piece in the main paper roll. Moreover, the provided problem instances do not have pieces of identical dimension. Let us think of a general case, in which:

5. the rotation is allowed: which of the CP model and the SAT/SMT encoding is easier to modify to take this into account? How would you modify that model/encoding?

6. there can be multiple pieces of the same dimension: how would you improve the CP model and the SAT/SMT encoding?

Points 1 — 4 will be part of your MiniZinc and Z3 programs. If you prefer, you could express points 5 and 6 using a mathematical notation, without necessarily inserting them in your programs.

4 Submission

The project will be submitted via IOL. If you are working as a group, it suffices that only one student submits the project, provided that the members' names are indicated. While there is no strict deadline for submission, the students are encouraged to finish the project over the summer, before the start of the second year.

For submission, create 2 folders, named CP and SAT/SMT and add the following files in each folder:

- a document in PDF describing the model, including the points 1–6, as well as any remarks or comments you consider appropriate.

- a directory `out` with the output files of those instances that could be solved successfully. The output file corresponding to an instance $w \times l$.txt should be named $w \times l$–out.txt.

- a directory `src` with all the source code used to carry out the project, together with a `README` file with basic instructions for execution (so that results can be reproduced).

Upload a `tgz` or `zip` compressed archive after naming it with your surname(s).

5 Examination, Evaluation and Grading

The examination will take place in the form of an oral exam on Teams. Oral exam dates will be set and announced, as the students submit their projects. Project evaluation will be based on the students' ability to model and solve a problem using CP and SAT/SMT and to use tools like MiniZinc and Z3, as well as adherence to the points 1–6 described previously. While it suffices to produce either a SAT or an SMT encoding, bonus points will be given to those who produce both.

A project evaluation will result in a mark v_1 between 0 and 6, with the following meaning in a scale of 30: $0 \to 9, 1 \to 11, 2 \to 12, 3 \to 13, 4 \to 14, 5 \to 15, 6 \to 16$. An insufficient project will not get a mark. The final mark of the course v_f will be $v_1 + v_2$ where v_2 is the mark obtained from Module 2 of Vittorio Maniezzo. Those who have $v_f \geq 31$ will obtain a lode.

6 Academic Integrity

Intellectual development requires honesty, responsibility, and doing your own work. Plagiarism is the use of someone else's work, words, or ideas as if they were your own. Any idea taken from any person or resource should be cited appropriately. Plagiarised work will not be evaluated.