# Project Work
# Combinatorial Decision Making and Optimization
# Module 1

**Topic**: Modelling and solving the VLSI problem
**Authors**: Diego Mazzieri, Zeynep Kiziltan
**Date**: June 1, 2021

This document describes a project work for the first module of the Combinatorial Decision Making and Optimization course of the academic year 2020/2021, which is about modelling and solving a combinatorial decision problem with (i) Constraint Programming (CP), and (ii) propositional SATisfiability (SAT) and/or its extension to Satisfiability Modulo Theories (SMT). You can work alone or form a group of 2 members, and it suffices for point (ii) to develop a SAT or an SMT solution. If you do both, you obtain bonus points. You can also form a group of 3 members, but in that case you are required to develop both SAT and SMT solutions. You are free to propose other combinatorial decision problems, provided that you get our approval before start working on it. For any questions, please contact us. Good luck ☺

## 1 Description of the Problem

VLSI (Very Large Scale Integration) refers to the trend of integrating circuits into silicon chips. A typical example is the smartphone. The modern trend of shrinking transistor sizes, allowing engineers to fit more and more transistors into the same area of silicon, has pushed the integration of more and more functions of cellphone circuitry into a single silicon die (i.e. plate). This enabled the modern cellphone to mature into a powerful tool that shrank from the size of a large brick-sized unit to a device small enough to comfortably carry in a pocket or purse, with a video camera, touchscreen, and other advanced features.

As the combinatorial decision and optimization expert, you are assigned to design the VLSI of the circuits defining your electrical device: given a fixed-width plate and a list of rectangular circuits, decide how to place them on the plate so that the length of the final device is minimized (improving its portability). In order for the device to work properly, each circuit must be placed in a fixed orientation with respect to the others, therefore it cannot be rotated.

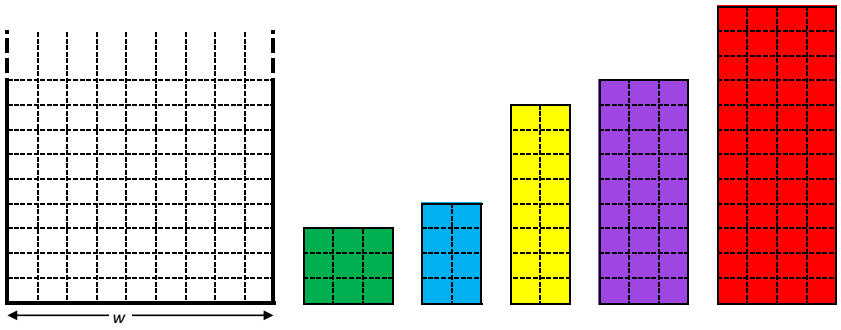


Figure 1: Graphical representation of the instance.

## 2 Format of the Instances

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

**Instance Format** An instance of VLSI is a text file consisting of lines of integer values. The first line gives $w$, which is the width of the silicon plate. The following line gives $n$, which is the number of necessary circuits to place inside the plate. Then $n$ lines follow, each with $x_i$ and $y_i$, representing the horizontal and vertical dimensions of the $i$-th circuit. For example, a file with the following lines:

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

describes an instance in which the silicon plate has the width 9, and we need to place 5 circuits, with the dimensions $3 \times 3$, $2 \times 4$, $2 \times 8$, $3 \times 9$, and $4 \times 12$. Figure 1 shows the graphical representation of the instance.

**Solution Format** Where to place a circuit $i$ can be described by the position of $i$ in the silicon plate. The solution should indicate the length of the plate $l$, as well as the position of each $i$ by its $\hat{x}_i$ and $\hat{y}_i$, which are the

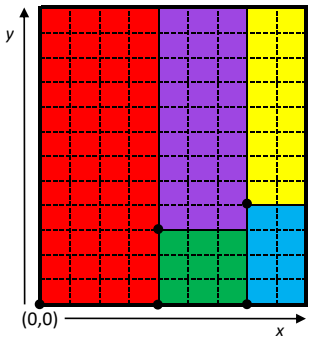


Figure 2: Graphical representation of the solution.

coordinates of the left-bottom corner $i$. This could be done by for instance adding $l$ next to $w$, and adding $\hat{x}_i$ and $\hat{y}_i$ next to $x_i$ 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
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 $3 \times 3$ circuit is at $(4, 0)$. The solution can be represented graphically as in Figure 2.

## 3 Project Work

The purpose of this project is to model and solve VLSI problem with (i) Constraint Programming (CP), and (ii) propositional SATisfiability (SAT) and/or its extension to Satisfiability Modulo Theories (SMT), using the MiniZinc CP solver and the Z3 SAT/SMT solver. You will decide yourselves the problem constraints around the problem description, and build the CP model and the SAT and/or SMT encoding accordingly. A suite of problem instances are provided in the format specified in Section 2 for experimental purposes. You should not make any assumptions based on these instances. Your approaches should work for any random instance.

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, the main problem constraints and the objective function.

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

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

4. Observe which symmetries are present in the problem and/or in your model, and improve the CP model and the SAT and/or SMT encoding with symmetry breaking constraints. Note that when you post multiple symmetry breaking constraints, they should not conflict with each other (i.e., there must be at least one solution satisfying all the symmetry breaking constraints).

5. Investigate the best way to search for solutions in CP which does not conflict with the symmetry breaking constraints. Your symmetry breaking constraints should not conflict with the SAT and/or SMT search either.

6. The problem requirements do not allow the rotation of the circuits. This means that, an $n \times m$ circuit cannot be positioned as an $m \times n$ circuit in the silicon plate. Let us think of a general case, in which the rotation is allowed. How would you modify the CP model and the SAT and/or SMT encoding?

7. Investigate the best way to search for solutions in CP also when rotation is allowed.

## 4 Project Report

You should describe clearly in a report how you tackled each of the points above by using a mathematical notation. While a MiniZinc notation is acceptable to describe your CP model and search annotations, please use propositional logic and first-order logic for your SAT and SMT models. Do not copy and paste Z3Py code.

The report should also describe your experimental results obtained by the provided instances. Show experimental results of your best combination of model and search in CP, and of your best encoding in SAT and/or SMT. As a reference, solving processes that exceed a time limit of over 5 minutes (300 secs) should be aborted. Take into account that, depending on the solving technology, on the machine, etc., some of the instances may be too difficult to solve within the time limit. For this reason, it is not strictly necessary to succeed in solving all instances to pass the project. However, you are encouraged to solve as many instances as possible.

## 5 Submission

The project will be submitted via Virtuale. If you are working as a group, it suffices that only one student submits the project, provided that the members' names and e-mail addresses 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 a separate folder for each technology, (named CP, SAT, and/or SMT) and add the following files in each folder:

- a document in PDF describing your work, including the points 1–7, 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 $ins - i$.txt should be named $out - i$.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 the surnames of the group members.

## 6 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. We will be available to evaluate the projects in the second half of July and late August/early September. **Attention: if you have any specific deadline by which you need to register the final mark (for instance due to a scholarship), please contact us now!**

Project evaluation will be based on the students' ability to model and solve a problem using CP and SAT and/or SMT, and to use tools like MiniZinc and Z3, as well as adherence to the points 1–7 described previously. While it suffices to produce either a SAT or an SMT encoding for groups of max 2 students, bonus points will be given to those who produce both. Groups of max 3 students are required to produce both.

A project evaluation will result in a mark $v_1$ between 9 and 16, with 9 being the minimum requirement to pass within the scale $0 \dots 16$. 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.

## 7 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.