SAT Report

p.soleimanybaraijany@studio.unibo.it[1],
maryam.salehi@studio.unibo.it[2], and
shadi.farzankia@studio.unibo.it[3]

[1]University of Bologna

September 2022

1 Introduction

ery-large-scale integration(VLSI) is the process of embedding numerous transistors onto a silicon semiconductor microchip technology that emerged in the late 1970s when microcomputer chips were developing. The last few decades have witnessed remarkable growth in the electronics industry thanks to the advancements in VLSI technology. Presently, cellular communication and smartphones afford unprecedented processing capabilities and portability due to technological improvements and it is forecasted that this trend will continue as there is an ever-increasing demand for state-of-the-art devices. In VLSI design, maintaining a small area and high performance have always been two conflicting constraints considered by integrated circuit designers.
In this project, we try to propose a solution to the size problem of VLSI circuits. A certain number of circuits are embedded in a plate with a fixed width and the ultimate goal is to minimize the height of the plate.
Four different approaches including constraint programming (CP), Propositional Satisfiability (SAT), linear programming (LP), and satisfiability modulo theories (SMT) have been utilized to solve the problem and the results of all approaches have been compared and analyzed to find the best solution.
This report is concerned with how propositional Satisfiability is used to solve the VLSI problem.

2 Instance Format and Solution Format

In this section, the format of input for VLSI instances as well as the output solutions are presented.

2.1 Instance Format

Input instances are defined as follows:

w
n
x_0 y_0
x_1 y_1
...

Where:

w: width of the plate
n: number of circuits to be embedded
x_i : horizontal dimension of i^{th} circuit
y_i : vertical dimension of 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 3×3, 2×4, 2×8, 3×9, and 4×12. Figure 1 shows the graphical representation of the instance.

2.2 Output format

After solving the problem using the input instances, output can be represented as:

w max_height
n
x_0 y_0 pos_{x_0} pos_{y_0}
x_1 y_1 pos_{x_1} pos_{y_1}
...

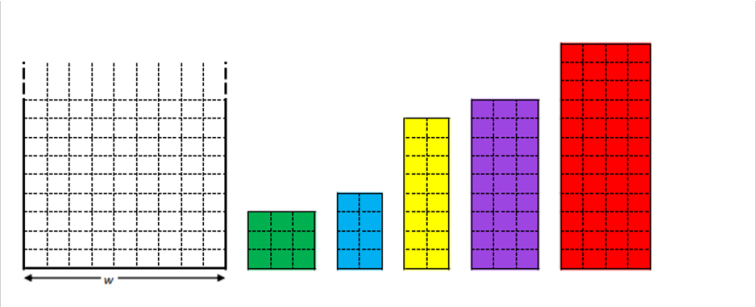

Figure 1: Graphical representation of an instance

where:

w, n, x_i, y_i: same as input instances
max_height: The optimal height of the plate
pos_{x_i} = horizontal coordinate of the bottom left point of the i^{th} circuit
pos_{y_i}= vertical coordinate of the bottom left point of the i^{th} circuit

For example, a possible solution for the instance presented in figure 1 can be as follows:

9 12
5
3 3 4 0
2 4 7 0
2 8 7 4
3 9 4 3
4 12 0 0

Which states that the maximum height is 12 and the left bottom corner of 3×3 circuit is at (4,0). The output has been depicted in figure 2.

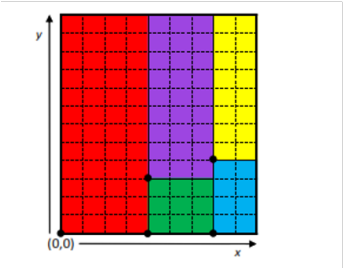

Figure 2: Graphical representation of one solution

3 Preliminary Concepts and Modeling

The Boolean Satisfiability problem (also called Propositional satisfiability problem, Satisfiability, SAT, or BSAT) is a kind of math-based problem in which it is determined whether there is an interpretation that satisfies a Boolean Formula. A formula is satisfiable if all of its variables can be replaced by True or False in such a way that the whole formula is True. However, if no assignment of variables exists that makes the formula True, it is called unsatisfiable. SAT is one of the most discussed problems in logic and computer science and it was the first problem to be proved NP-complete. In early 1970's Cook and Levin showed that NP-complete problems have deterministic polynomial time algorithms. Solving a wide range of optimization problems are as difficult as SAT since these problems can be reduced to a SAT problem. Heuristic SAT algorithms are able to solve complex problems such as circuit design and artificial intelligence problems that may contain tens of thousands of variables.
In this section as another approach, SAT is used to address the same problem. The first part of modeling includes model selection and defining the proper variables.

3.1 Variables and domains

This section includes a summary of the variables we used for each one the valid domain is described.

3.1.1 Maximum and Minimum Height

The width of the plate is fixed and is equal to w, However, the height of the plate is a variable that should satisfy two criteria:

1. The height should be determined such that the area of plate is minimized.
2. We should be able to embed all the circuits on the plate without overlapping.

Considering the criteria above, the plate's minimum height is the maximum value between the maximum height among all the circuits embedded on the plate and the sum of the areas of all the circuits divided by width of the plate.

$$min\_height = max(y_{max}, \frac{\sum_{i=1}^{n} x_i * y_i}{w})$$

The height of the plate cannot be less than the maximum value of the heights of all circuits embedded on the plate. This is because in the best case, all circuits can be placed in only one row. However, in most cases we need to have more than one row and a good estimation of the minimum height can be achieved by dividing the sum of the areas of all the circuits by width of the plate which is fixed.
In order to determine the maximum height, one needs to consider the worst case

in which all of the circuits have the maximum width and only one circuit can be embedded in each row. As a result, the maximum height is equal to the sum of the heights of all circuits.

$$max\_height = \sum_{i=1}^{n} y_i$$

Thus, the variable height ranges from min_height to max_height. Since we seek to minimize the height, it was decided to start solving the problem using the minimum height. If the solver is able to satisfy the two criteria above, we have found the optimal solution. Otherwise, we need to increment the height by one in each iteration until the optimal solution is found. If the solver reaches a height more than max_height or if solving of the problem takes more than 300 seconds the solver fails.

$$min\_height \leq height \leq max\_height$$

3.1.2 Position of the Circuits

To determine where each circuit is located on the plate, only the coordinates of its left-bottom corner are kept. Variables pos_x and pos_y denote the horizontal and vertical coordinates of the left-bottom corner of the circuits respectively.
pos_x varies between 0 and width minus $min(x)$, where min(x) is the minimum of the lengths of the circuits.
pos_y varies between zero and maximum height minus $min(y)$, where $min(y)$ is the minimum of the heights of the circuits.

$$0 \leq pos_x \leq w - x_{min}$$

$$0 \leq pos_y \leq max\_height - y_{min}$$

3.1.3 Plate

The whole plate has been considered a three-dimensional array. The first two dimensions of this array represent the coordinates of the plate. The first coordinate denoted by i, varies between zero and w (width of the plate). The second dimension shown by j ranges from zero to height. The third coordinate c is representative of where each circuit is located on the plate.

$$0 \leq i \leq w$$

$$0 \leq j \leq height$$

$$1 \leq c \leq n$$

For example if the coordinates (0,0,0), (0,1,0), (1,0,0), (1,1,0) in our plate have the value True, it means that first circuit has been put in these coordinates. As it is shown in figure 3:

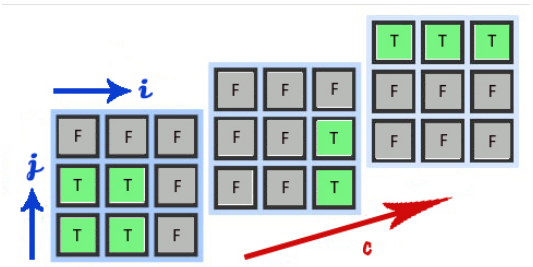

Figure 3: Three-dimensional representation of the plate

4 Constraints

Three constraints were used to solve the VLSI problem using SAT approach. For the sake of simplicity in the logical representation, we decided to start the indexes from 1 instead of 0.

4.1 The Biggest Rectangle Constraint

The biggest rectangle constraint has been used to break the symmetry. By putting the circuit with the biggest area at coordinates (0,0), vertical, horizontal, and 180 degrees symmetry can be broken. At the beginning, circuits are ordered based on their areas and the one with the biggest value will have the index one. This circuit with index one will be put on (0,0).

$$\bigwedge_{i=1}^{x[1]} \bigwedge_{j=1}^{y[1]} plate_{i,j,0}$$

4.2 at-most-one Constraint

at-most-one constraint is necessary to avoid overlapping. To ensure that only one circuit has been put on each cell of the plate the following constraint was used:

$$\bigwedge_{i=1}^{w} \bigwedge_{j=1}^{height} \bigwedge_{c_1=1}^{n-1} \bigwedge_{c_2=c_1+1}^{n} \neg(plate_{i,j,c_1}, plate_{i,j,c_2})$$

4.3 at-least-one Constraint

at-least-one constraint is used for two purposes:

1. On each position of the plate exactly one circuit should be put. Using at-least-one constraint in conjunction with at-most-one constraint means

that exactly one circuit can be put in each position.

$$\bigvee_{i=1}^{w} \bigvee_{j=1}^{height} \bigvee_{c=1}^{n} plate_{i,j,c}$$

Exactly one circuit in each position:

$$\bigwedge_{i=1}^{w} \bigwedge_{j=1}^{height} \bigwedge_{c_1=1}^{n-1} \bigwedge_{c_2=c_1+1}^{n} \neg(plate_{i,j,c_1}, plate_{i,j,c_2}) \quad \wedge \quad \bigvee_{i=1}^{w} \bigvee_{j=1}^{height} \bigvee_{c=1}^{n} plate_{i,j,c}$$

2. The second purpose of using at least one constraint is to ensure that each circuit has been put on one of its possible positions:
   For each circuit there might be several positions where it can be placed. Moving on the plate (like a sliding window) we find all the positions on the plate where it is possible to put that circuit. After finding all the possible positions for each circuit, we use at least one constraint to make sure that the circuit has been put in one the positions we had found for it.

$$\bigwedge_{c=1}^{n} \bigvee_{i=1}^{w-x[c]} \bigvee_{j=1}^{height-y[c]} \bigwedge_{i_2=i}^{i+x[c]} \bigwedge_{j_2=j}^{j+y[c]} plate_{i_2,j_2,c}$$

5 Results

The implemented SAT model could solve 18 out of 40 instances within the given time constraint. We were interested in developing the SAT approach (as an extra project in addition to SMT) to compare its results with all the other solvers we developed to determine which approach performs the best for VLSI problem.
It appears that the SAT approach has a weaker performance in comparison with CP, SMT and LP as it could solve less instances and it took much more time.