August 22, 2022

# 1 SAT

## 1.1 A simplified version of the problem

In order to model the problem with SAT, at first we consider a simplified version of the problem. Instead of minimizing the length of the plate with the given plate, we consider a fixed value $H$ for the height and check with SAT if it is possible to insert the given circuits (without being able to rotate them) in the plate within the fixed width $W$. In this case, we can think of the plate as a rectangle in $\mathbb{R}^2$, with vertices $A(0,0), B(W,0), C(W,H), D(0,H)$.

We can then introduce the atomic propositional formulas $x_{ijk}$ with the intended meaning that $x_{ijk}$ is true if and only if the bottom-left vertex of the $k$-th circuit is the point $(i,j)$. In general, if the number of circuits is $K$, this would require to introduce $W \cdot H \cdot K$ formulas, but in order to reduce the total number of formulas we choose, for each circuit $k$ with width $w(k)$ and height $h(k)$, to restrict $i$ and $j$ respectively to the sets $I_k = \{0, \ldots, W - w(k)\}$ and $J_k = \{0, \ldots, H - h(k)\}$. As a consequence the total number of atomic formulas to use becomes

$$\sum_{k=1}^{K} (W - w(k) + 1)\,(H - h(k) + 1) < WHK$$

.

### 1.1.1 Fundamental constraints

We need to model the constraints of the problem by means of several propositional formulas. A solution (if any) of the simplified problem will then be represented by an assignment of truth values to each $x_{ijk}$ which satisifies all of the formulas that model the constraints at the same time.

Our definitions of the atomic formulas $x_{ijk}$ already guarantee that every circuit will be positioned within the bounds of the outer rectangle $ABCD$. In order to guarantee the correctness of the solution we then need to define two fundamental constraints: the first one is that every rectangle must have exactly one bottom-left vertex, and the second one is that any two different circuits must not overlap.

- *Existence and uniqueness of the bottom-left vertex location:* for every circuit $k = 1 \dots K$, we need to impose the constraint *exactly_one* on the formulas $x_{ijk}$, with $i \in I_k$, $j \in J_k$. This amounts to imposing the constraints
$$\bigwedge_{k=1}^{K} \bigvee_{i \in I_k, j \in J_k} x_{ijk} \qquad \text{(at\_least\_one)}$$
and
$$\bigwedge_{k=1}^{K} \bigwedge_{\substack{i,i' \in I_k \\ j,j' \in J_k \\ (i,j) \neq (i',j')}} \neg(x_{ijk} \wedge x_{i'j'k}) \qquad \text{(at\_most\_one)}$$

- *No-overlap constraints:* Let's assume that the $k$-th circuit has the bottom-left vertex on the point of coordinates $(i, j)$. We can then observe that the $k'$-th circuit overlaps with the former if and only if its bottom-left corner lies within the rectangle with vertices:
  - $V1(\max(0, i - w(k') + 1), \max(0, j - h(k') + 1))$;
  - $V2(\min(i + w(k) - 1, W - w(k')), \max(0, j - h(k') + 1))$;
  - $V3(\min(i + w(k) - 1, W - w(k')), \min(j + h(k) - 1, H - h(k')))$;
  - $V4(\max(0, i - w(k') + 1), \min(j + h(k) - 1, H - h(k')))$.

  By defining, for each two circuits $k, k'$, the sets
$$I_{kk'} := \{\max(0, i - w(k') + 1, \dots, \min(i + w(k) - 1, W - w(k')\}$$
  and
$$J_{kk'} := \{\max(0, j - h(k') + 1), \dots, \min(j + h(k) - 1, H - h(k'))\}$$
  we can then define these constraints as:
$$\bigwedge_{1 \leq k < k' \leq K} \bigwedge_{\substack{i \in I_k \\ j \in J_k \\ i' \in I_{kk'} \\ j' \in J_{kk'}}} (\neg x_{ijk} \vee \neg x_{i'j'k'})$$

A significant drawback of this approach is that the number of constraints tends to become extremely large as the number of rectangles $K$ and the dimensions $H, W$ of $ABCD$ increase. In order to address at least part of this problem, we tried several different encodings for the *at_most_one* constraint, namely the *bitwise*, the *Heule's* and the *sequential* encodings, and the last one experimentally appeared to work better. However, keeping in mind the practical motivation of the problem, we ultimately decided to get rid entirely of the *at_most_one* constraint, and only the *at_least_one* constraint was left there: in this way, in a given solution a circuit would be given a list (if any) of possible positions of its bottom-left vertex. This approach in general increases the number of solutions, but in order to obtain a solution of the original problem it is sufficient, for each rectangle $k$, to choose arbitrarily only one of the possible positions of its bottom-left vertex returned by the the solver.

### 1.1.2 Symmetry-breaking constraints

Again, since we observed that the time required to generate all the constraints often far exceeded the time needed to actually solve the problem, we opted not to add additional implied constraints that could improve the time required to satisfy the constraints. Moreover, so as not to slow down the constraint generation too much, we decided to add only two symmetry-breaking constraints: one for the horizontal symmetry and one for the vertical symmetry.

Let $X$ be the set of all variables $x_{ijk}$. Then the permutations on $X$

$$\begin{aligned} \pi_h \colon X &\longrightarrow X \\ x_{ijk} &\longmapsto x_{(W-i-w(k))jk} \end{aligned}$$

and

$$\begin{aligned} \pi_v \colon X &\longrightarrow X \\ x_{ijk} &\longmapsto x_{i(H-j-h(k))k} \end{aligned}$$

identify respectively the horizontal and the vertical symmetries of the model.

Unlike with the previous approach with CP, we do not have global constraints at our disposal. A possible solution would be to implement the lexicographic ordering constraint in a naive way. In