Solving Nonogram with SAT Solver

B08901192 楊登堡

December 27, 2023

1 Introduction

Nonograms are picture logic puzzles in which cells in a grid have to be colored or left blank according to numbers given at the side of the grid, called *clues*, to reveal a hidden picture. In this puzzle, the clues measure the length of blocks in each row or column. For example, a clue of "2 3" means there are a 2-length block and a 3-length block in that row or column, and in that order, there exist at least one blank between any two blocks. The Figure 1. shows an example of nonogram.

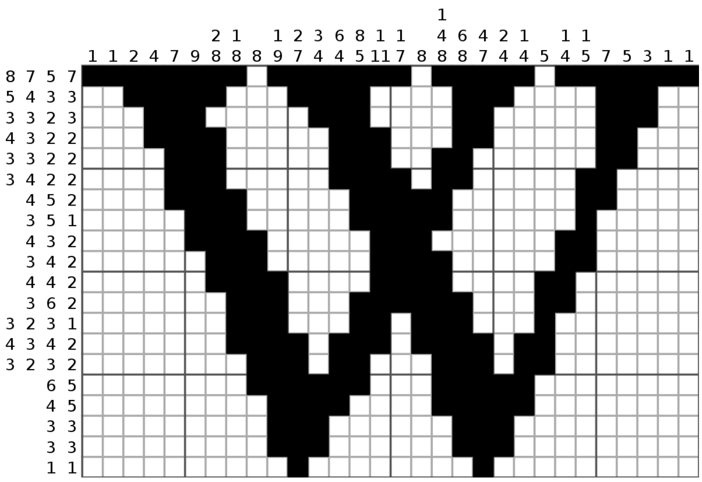

Figure 1example of nonogram

2 Modeling

In this project, we focus on solving nonogram with SAT solvers. In this section, we will elaborate on how to use a SAT solver to solve a Nonogram.

2.1 Input

The inputs of a nonogram including the number of rows and columns, and the clues at the side of the grid. We will use the notation as follows:

$$
\begin{aligned}
m &: \text{ the number of rows} \\
n &: \text{ the number of columns} \\
r_{i,k} &: \text{ length of } k\text{-th block in the } i\text{-th row } (i = 0, 1, ..., m-1, k = 0, 1, ..., a_i - 1) \\
c_{j,k} &: \text{ length of } k\text{-th block in the } j\text{-th column } (j = 0, 1, ..., n-1, k = 0, 1, ..., b_j - 1),
\end{aligned}
$$

where a_i and b_j is the number of clues in the i-th row and j-th column, respectively.

2.2 Variables

To use SAT solvers to solve a nonogram, we may set the variables as follows:

$$\begin{aligned} x_{i,j} &: \text{ color of } (i,j) \text{ cell} \\ l_{i,k,l} &: \text{ left-most position of the } k\text{-th block in the } i\text{-th row is } l \\ t_{j,k,l} &: \text{ top-most position of the } k\text{-th block in the } j\text{-th row is } l \end{aligned}$$

2.3 Constraints

With the inputs and variables set as previous, we may generate constraints as follows:

1. To reserve enough space for the last block of each row, the left-most position of it must be no more than $n - r_{i,a_i-1}$ in i-th row, and so are the other blocks in the same row. Therefore, we have the constraint
$$f_{1,1} = \bigwedge_{i=0}^{m-1} \bigwedge_{k=0}^{a_i-1} (l_{i,k,0} + l_{i,k,1} + \dots + l_{i,k,n-r_{i,a_i-1}})$$
Similarly, the top-most position of any block in j-th column must be between 0 and $m - c_{j,b_j-1}$.
$$f_{1,2} = \bigwedge_{j=0}^{n-1} \bigwedge_{k=0}^{b_j-1} (t_{j,k,0} + t_{j,k,1} + \dots + t_{j,k,m-c_{j,b_j-1}})$$
Denote $f_1 = f_{1,1} \wedge f_{1,2}$.

2. For each block in each row (column), the left-most (top-most) position of each block can only has one value.
$$f_2 = \left(\bigwedge_{i=0}^{m-1} \bigwedge_{k=0}^{a_i-1} \bigwedge_{l=0}^{n-r_{i,a_i-1}-1} \bigwedge_{m=l+1}^{n-r_{i,a_i-1}} (\neg l_{i,k,l} \vee \neg l_{i,k,m}) \right) \wedge \left(\bigwedge_{j=0}^{n-1} \bigwedge_{k=0}^{b_j-1} \bigwedge_{l=0}^{m-c_{j,b_j-1}-1} \bigwedge_{m=l+1}^{m-c_{j,b_j-1}} (\neg t_{j,k,l} \vee \neg t_{j,k,m}) \right)$$

3. Any two blocks cannot overlapped each other. That is, for any block in each row or column, it cannot overlap with previous block and next block. Then, $f_{3,1}$ describes the constraint of non-overlap of any block and its next block in each row,
$$\begin{aligned} f_{3,1} &= \bigwedge_{i=0}^{m-1} \bigwedge_{k=0}^{a_i-2} \bigwedge_{l=0}^{n-r_{i,a_i-1}} [l_{i,k,l} \Rightarrow (l_{i,k+1,r_{i,k}+l+1} + l_{i,k+1,r_{i,k}+l+2} + \dots + l_{i,k+1,n-r_{i,a_i-1}})] \\ &= \bigwedge_{i=0}^{m-1} \bigwedge_{k=0}^{a_i-2} \bigwedge_{l=0}^{n-r_{i,a_i-1}} [(\neg l_{i,k,l} + l_{i,k+1,r_{i,k}+l+1} + l_{i,k+1,r_{i,k}+l+2} + \dots + l_{i,k+1,n-r_{i,a_i-1}})], \end{aligned}$$
and $f_{3,2}$ shows the constraint of non-overlap of any block and its next block in each column,
$$\begin{aligned} f_{3,2} &= \bigwedge_{j=0}^{n-1} \bigwedge_{k=0}^{b_j-2} \bigwedge_{l=0}^{m-c_{j,b_j-1}} [t_{j,k,l} \Rightarrow (t_{j,k+1,c_{j,k}+l+1} + t_{j,k+1,c_{j,k}+l+2} + \dots + t_{j,k+1,m-c_{j,b_j-1}})] \\ &= \bigwedge_{j=0}^{n-1} \bigwedge_{k=0}^{b_j-2} \bigwedge_{l=0}^{m-c_{j,b_j-1}} (\neg t_{j,k,l} + t_{j,k+1,c_{j,k}+l+1} + t_{j,k+1,c_{j,k}+l+2} + \dots + t_{j,k+1,m-c_{j,b_j-1}}), \end{aligned}$$
and $f_{3,3}$ represents the constraint of non-overlap of any block and its previous block in each row,
$$\begin{aligned} f_{3,3} &= \bigwedge_{i=0}^{m-1} \bigwedge_{k=1}^{a_i-1} \bigwedge_{l=0}^{n-r_{i,a_i-1}} [l_{i,k+1,l} \Rightarrow (l_{i,k,0} + l_{i,k,1} + l_{i,k,l-r_{i,k}-1})] \\ &= \bigwedge_{i=0}^{m-1} \bigwedge_{k=1}^{a_i-1} \bigwedge_{l=0}^{n-r_{i,a_i-1}} (\neg l_{i,k+1,l} + l_{i,k,0} + l_{i,k,1} + l_{i,k,l-r_{i,k}-1}) \end{aligned},$$

and $f_{3,4}$ represents the constraint of non-overlap of any block and its previous block in each column,

$$f_{3,4} = \bigwedge_{j=0}^{n-1} \bigwedge_{k=1}^{b_j-1} \bigwedge_{l=0}^{m-c_{j,b_j-1}} [t_{j,k+1,l} \Rightarrow (t_{j,k,0} + t_{j,k,1} + t_{j,k,l-c_{j,k}-1})]$$
$$= \bigwedge_{j=0}^{n-1} \bigwedge_{k=1}^{b_j-1} \bigwedge_{l=0}^{m-c_{j,b_j-1}} (\neg t_{j,k+1,l} + t_{j,k,0} + t_{j,k,1} + t_{j,k,l-c_{j,k}-1}))$$

Denote $f_3 = f_{3,1} \wedge f_{3,2} \wedge f_{3,3} \wedge f_{3,4}$.

4. After discussing about the constraint of each block, we consider the relation between a colored cell and each block. It is clear that the (i,j) cell is colored if and only if there exists some block containing (i,j) cell. That is, for each i,j,

$$x_{i,j} = 1 \iff \bigvee_{k=0}^{a_i-1} [(l_{i,k} \le j) \wedge (j < l_{i,k} + r_{i,k})]$$
$$x_{i,j} = 1 \iff \bigvee_{k=1}^{b_j} [(t_{j,k} \le i) \wedge (i < t_{j,k} + c_{j,k})],$$

where $l_{i,k}$ and $t_{j,k}$ represent the left-most and top-most position of the corresponding block for short. Then, we can generate the CNF of the relation as follows:

$$f_{4,1} = \bigwedge_{i=0}^{m-1} \bigwedge_{j=0}^{n-1} \left(\neg x_{i,j} \vee \bigvee_{k=0}^{a_i-1} (l_{i,k,j} + l_{i,k,j-1} + ... + l_{i,k,j-r_{i,k}+1}) \right)$$
$$f_{4,2} = \bigwedge_{i=0}^{m-1} \bigwedge_{j=0}^{n-1} \left(\neg x_{i,j} \vee \bigvee_{k=0}^{b_j-1} (t_{j,k,i} + t_{j,k,i-1} + ... + t_{j,k,i-c_{j,k}+1}) \right)$$
$$f_{4,3} = \bigwedge_{i=0}^{m-1} \bigwedge_{j=0}^{n-1} \bigwedge_{k=0}^{a_i-1} (l_{i,k,0} + l_{i,k,1} + ... + l_{i,k,j-r_{i,k}} + l_{i,k,j+1} + ... + l_{i,k,n-r_{i,a_i-1}} + x_{i,j})$$
$$f_{4,4} = \bigwedge_{i=0}^{m-1} \bigwedge_{j=0}^{n-1} \bigwedge_{k=0}^{b_j-1} (t_{j,k,0} + t_{j,k,1} + ... + t_{j,k,i-c_{j,k}} + t_{j,k,i+1} + ... + t_{j,k,m-c_{j,b_j-1}} + x_{i,j})$$

Denote $f_4 = f_{4,1} \wedge f_{4,2} \wedge f_{4,3} \wedge f_{4,4}$.

Let $f = f_1 \wedge f_2 \wedge f_3 \wedge f_4$. As we use SAT solvers to solve whether f is satisfiable, we can extract $x_{i,j}$ for each i,j as the answer of the nonogram if f is satisfiable. If f is unsatisfiable, it implies the nonogram has no solution.

3 Experimental Result

In Table 1, we demonstrate on the relationship between running time and the size of a solvable nonogram. To prevent the influence of clues, we set all the clues to 1.

size	5x5	10x10	15x15	20x20	25x25	50x50	100x100	150x150	200x200
time(s)	0.0038	0.011	0.021	0.072	0.040	0.78	8.06	54.92	222.34

Table 1

In Table 2, we experiment on the relationship between running time and the size of an unsolvable nonogram. Similarly, to prevent the influence of clues, we set all the clues to 1 except the clue for the first row, which is set to 2. For cases marked with "-", it indicates that the runtime exceeds 30 minutes and is interrupted. In the unsatisfiable case, the running time is much longer than in the satisfiable case.

size	5x5	10x10	15x15	20x20	25x25	50x50	100x100	150x150	200x200
time(s)	0.0027	13.26	-	-	-	-	-	-	-

Table 2

4 Conclusion

Table 1 indicates that finding a solution for a nonogram is a quick process. Considering the majority of nonograms in the App Store game have a size not exceeding 50x50, our model demonstrates efficiency in solving them. However, the challenge of generating a unique solution for a nonogram persists, as checking its satisfiability with this model is time-consuming.