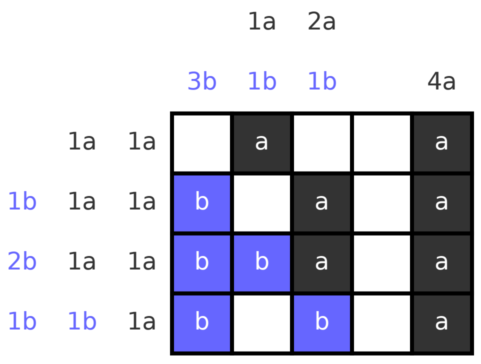
Problem 4: Solve Picture Puzzles

AI1SysProj 2021/2022

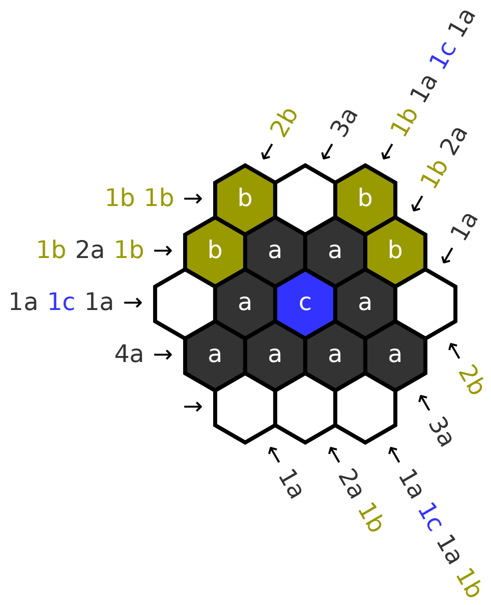
Topic:	SAT Solvers
Due on:	March 1, 2022
Version from:	January 20, 2022

1 Task Summary

Solve picture puzzles using a SAT solver. The puzzles are based on a multi-color variant of **nonograms** [WN], which are NP-complete logic puzzles originating in Japan. Besides the traditional rectangular grids, we will also cover hexagonal grids.



(a) Rectangular grid.



(b) Hexagonal grid.

Figure 1: Two example puzzles (solved).

2 Puzzle Rules

The goal of the puzzle is to color the cells of a grid using clues. A **clue** is a sequence $n_1c_1, \ldots, n_kc_k$, where n_i are numbers and c_i are color references. Each element n_ic_i indicates

```
rect 4 5
#ffffff #333333 #6666ff
1a 1a
1b 1a 1a
2b 1a 1a
1b 1b 1a
3b
1a 1b
2a 1b

4a
```

```
hex 3
#ffffff #333333 #999900 #3333ff
1b 1b
1b 2a 1b
1a 1c 1a
4a

1a
2a 1b
1a 1c 1a 1b
3a
2b
1a
1b 2a
1b 1a 1c 1a
3a
2b
```

Listing 1: Encoding of the puzzles from Figure 1a (left) and Figure 1b (right).

a continuous **block** of n_i cells of color c_i. The blocks occur in the same order as listed in the clue. Consecutive blocks of the same color must be separated by at least one empty cell.

As an example, let us take a look at the third row of the example puzzle in Figure 1a. We use letters (here `a` and `b`) to refer to colors. The clue `2b 1a 1a` for the third row indicates that it starts with a 2-cell block of color `b` (in this case blue), which is followed by two 1-cell blocks of color `a` (black). Since the last two blocks have the same color, they must be separated by at least one empty cell.

3 Puzzle Format

Puzzles are represented as text files. First, the grid is described:

- Rectangular grids start with the line `rect <height> <width>`, where `<height>` is the number of rows and `<width>` is the number of columns.
- Hexagonal grids start with the line `hex <size>`, where `<size>` indicates the side length

of the hexagon.

The second line lists the colors, starting with the background color, followed by the colors that are later referred to as `a`, `b`, `c`, etc.

Each of the remaining lines corresponds to a clue. For a rectangular grid, the row clues are listed first, followed by the column clues. In a hexagonal grid, we have clues in three directions (see Figure 1b). The file lists them counter-clockwise, starting with the top-left corner (`1b 1b` in the example). Listing 1 shows the encodings of the example puzzles.

4 Using SAT Solvers

To solve the puzzles, you should translate them into a SAT problem that can be solved by off-the-shelf SAT solvers.

4.1 Encoding SAT Problems and their Solutions

We will represent SAT problems using (a subset of) the DIMACS format, which is commonly used for SAT competitions. The first line is always of the form `p cnf <nvars> <nclauses>`, where `<nvars>` is the number of variables used and `<nclauses>` is the number of clauses. Each subsequent line encodes one clause as a list of integers. Positive integers denote variables and negative integers their negations (i.e. the literal X_i is denoted by i and the literal $\neg X_i$ by $-i$). The end of a clause is marked by a 0. Figure 2 shows the encoding of an example formula.

Problems in that format can be solved by many SAT solvers, including MiniSat [MS], which is relatively easy to use and install, and Kissat [KS], which has won recent SAT competitions.

Variable assignments that satisfy the problem should also be encoded as a list of integers, where positive integers indicate variables that are true and negative integers indicate variables set to false (see Figure 2 for an example). Both MiniSat and Kissat produce solutions in that format.

4.2 Linking Variables to Cell Colors

Ultimately, we want to infer the cell colors from the SAT solution. For that, we can link variables possible cell colors, which can be done in many different ways. In this assignment we decided to prescribe one way to make the evaluation easier. First, all cells are enumerated,

```
p cnf 3 2
1 2 0
-1 3 0
```

```
-1 2 -3
```

Figure 2: Encoding of $(X_1 \lor X_2) \land (\neg X_1 \lor X_3)$ (left) and the solution $X_1 = F, X_2 = T, X_3 = F$ (right).

row by row from left to right, starting from 0. The i-th cell is then linked to the variables $X_{i \cdot n+1}, X_{i \cdot n+2}, \ldots, X_{i \cdot n+n}$, where n is the number of non-background colors. The cell will then get color c iff $X_{i \cdot n+c}$ is true. If $X_{i \cdot n+c}$ is false for all $c \in \{1, \ldots, n\}$, the cell remains empty, i.e. it gets the background color.

For example, the SAT problem for the example in Figure 1b has $n = 3$. The solution should start with $X_1 = F, X_2 = T, X_3 = F$ (the first cell has color `b` ($c = 2$)), followed by $X_4 = F, X_5 = F, X_6 = F$ for the second cell, which has to remain empty.

Apart from the variables linked to cell colors you can (should) of course use additional helper variables.

5 Submission

At the deadline, we will download a snapshot of your repository. It should contain:

1. All your code.
2. Solutions to all the example puzzles in the assignment repository[1]. A solution to a puzzle `<name>.clues` should consist of:
   (a) the generated SAT file `<name>.sat` (or `<name>.sat.gz` if you prefer to gzip the file),
   (b) the solution to the SAT problem `<name>.sol`,
   (c) a visualization of the result `<name>.svg`) in the SVG format (the details of the visualization are up to you – the resulting picture should be recognizable, but there is no need to e.g. include the clues).
3. A README file explaining:
   (a) how to run your code to solve a new puzzle,
   (b) which of the example puzzles have a unique solution (people usually prefer puzzles

[1] `https://gitlab.rrze.fau.de/wrv/AISysProj/ws2122/sat/assignment`

with a unique solution), and

(c) what you did to determine if a puzzle has a unique solution.

6 Random Tips

1. For debugging purposes, you might want to make your own, small puzzles.
2. As a starting point, you could focus on classical nonograms (rectangular grid and just one non-background color). When you have a good understanding of how to solve them, moving on to the other puzzles should be fairly straight-forward.
3. The main challenge is to find an efficient representation of the clues in CNF (in particular the asymptotic growth of the number of clauses matters). Using the variables linked to cell colors as a starting point might lead you to simple but very inefficient representations that won't scale to larger problems (it might still be a good place to get started though). To be able to handle larger problems, it might be better if you come up with a good representation from scratch and then link it to the color variables described in Section 4.2.
4. Have a lot of fun :-)

References

[KS] *Kissat SAT Solver.* URL: `http://fmv.jku.at/kissat/` (visited on 01/18/2022).

[MS] *The MiniSat Page.* URL: `http://minisat.se/` (visited on 01/18/2022).

[WN] *Nonogram.* URL: `https://en.wikipedia.org/wiki/Nonogram` (visited on 01/18/2022).