INFO9015: Logic for Computer Science

Sudoku solving using propositional logic

Olivier Joris - 182113

Academic year 2021 - 2022

1 Encoding to SAT

Let $c(i, j, k)$ asserting that the cell in row i and in column j contains the number k. Let N be the size of the sudoku.

1.1 Individual cells

The first set of clauses indicates that each cell of the sudoku must contains at least one number in [1, N] : $\bigwedge_{i=1}^{N} \bigwedge_{j=1}^{N} \bigvee_{k=1}^{N} c(i, j, k)$

The second set of clauses indicates that each cell can not contain more than one number in [1, N]: $\bigwedge_{i=1}^{N} \bigwedge_{j=1}^{N} \bigwedge_{k=1}^{N-1} \bigwedge_{l=k+1}^{N} \neg(c(i, j, k) \wedge c(i, j, l))$

1.2 Rows

The first set of clauses indicates that each number in [1, N] appears at least once in each row : $\bigwedge_{i=1}^{N} \bigwedge_{k=1}^{N} \bigvee_{j=1}^{N} c(i, j, k)$

The second set of clauses indicates that each number in [1, N] appears at most once in each row : $\bigwedge_{i=1}^{N} \bigwedge_{k=1}^{N} \bigwedge_{j=1}^{N-1} \bigwedge_{l=j+1}^{N} \neg(c(i, j, k) \wedge c(i, l, k))$

1.3 Columns

The first set of clauses indicates that each number in [1, N] appears at least once in each column : $\bigwedge_{j=1}^{N} \bigwedge_{k=1}^{N} \bigvee_{i=1}^{N} c(i, j, k)$

The second set of clauses indicates that each number in [1, N] appears at most once in each column : $\bigwedge_{j=1}^{N} \bigwedge_{k=1}^{N} \bigwedge_{i=1}^{N-1} \bigwedge_{l=i+1}^{N} \neg(c(i, j, k) \wedge c(l, j, k))$

1.4 Squares

The first set of clauses indicates that each number in [1, N] appears at least once in each square : $\bigwedge_{i_{offset}=1}^{\sqrt{N}} \bigwedge_{j_{offset}=1}^{\sqrt{N}} \bigwedge_{k=1}^{N} \bigvee_{i=1}^{\sqrt{N}} \bigvee_{j=1}^{\sqrt{N}} c(i_{offset} * \sqrt{N} + i,\ j_{offset} * \sqrt{N} + j,\ k)$

The second set of clauses indicates that each number in [1, N] appears at most once in each square : $\bigwedge_{i_{offset}=1}^{\sqrt{N}} \bigwedge_{j_{offset}=1}^{\sqrt{N}} \bigwedge_{k=1}^{N} \bigwedge_{i=1}^{N} \bigwedge_{j=i+1}^{N} \neg c(i_{offset} * \sqrt{N} + (i\ mod \sqrt{N}),\ j_{offset} * \sqrt{N} + (i\ mod \sqrt{N}),\ v) \vee \neg c(i_{offset} * \sqrt{N} + (j\ mod \sqrt{N}),\ j_{offset} * \sqrt{N} + (j\ mod \sqrt{N}),\ k)$

2 Program features

The program can solve a given sudoku in a text file, solve all the sudokus contained in a directory, and generates sudoku of size 4, 9, and 16. The different execution modes are described in this section.

2.1 Solve a sudoku

To solve a sudoku of size 4, 9, or 16, the program can be launched using this command :

```
$ python3 sudokub.py 'text_file'
```

where `text_file` corresponds to the text file containing the sudoku that has to be solved.

2.2 Solve all sudokus from a repository

To solve all the sudokus of size 4, 9, or 16 contained in a repository, the program can be launched using this command :

```
$ python3 sudokub.py 'repository'
```

where `repository` corresponds to the repository containing the sudokus that have to be solved. The solutions are `.sol` files corresponding to the inital file names created in the same repository

2.3 Generate a sudoku of a given size

To generate a sudoku of size 4, 9, or 16, the program can be launched using this command :

```
$ python3 sudokub.py -generate 'size'
```

where `size` corresponds to the size of the sudoku that has to be generated. The sudoku is then display on the standard output. Two executions of this command can give two different sudokus.