Sudoku 4x4 using FOL in Prover9

Deac Melinda-Anca

January 14, 2023

1 Description of the game

Sudoku is a logic-based puzzle game in which players fill in a grid of 9x9 squares with numbers, ensuring that each row, column, and 3x3 sub-grid contains all of the digits from 1 to 9 without repetition. The objective is to fill in the grid so that every row, column and region contains the digits 1-9 and the contents of the diagonally divided regions match. It typically requires a combination of deduction and trial-and-error to solve. In my interpretation of the game, for less writing, I implemented a game of Sudoku of 4x4 squares.

2 Example of a valid 4x4 Sudoku game

1	2	3	4
4	3	2	1
2	1	4	3
3	4	1	2

3 Example of an invalid 4x4 Sudoku game

1	2	3	4
4	2	1	3
2	1	4	3
3	4	1	2

4 Code

4.1 A few explanations of the code:

This is what the puzzle looks like in my code:

S_{11}	S_{12}	S_{13}	S_{14}
S_{21}	S_{22}	S_{23}	S_{24}
S_{31}	S_{32}	S_{33}	S_{34}
S_{41}	S_{42}	S_{43}	S_{44}

"digit(x)" refers to the digit that is on square x. All the other explanations needed are in the code.

4.2 The actual code

```
%S_row_column

%Possible values: numbers from 1 to 4.
%Rule:
%    different digit on each square of one row
%    different digit on each square of one column
%    different digit on each square in a 2x2 square inside the grid.

%At least one digit for each square
```

```
digit(x) -> one(x) | two(x) | three(x) | four(x).
digit(S_1_1).
digit(S_1_2).
digit(S_1_3).
digit(S_1_4).
digit(S_2_1).
digit(S_2_2).
digit(S_2_3).
digit(S_2_4).
digit(S_3_1).
digit(S_3_2).
digit(S_3_3).
digit(S_3_4).
digit(S_4_1).
digit(S_4_2).
digit(S_4_3).
digit(S_4_4).


%At most one digit for each square
one(x) -> -two(x) & -three(x) & -four(x).
two(x) -> -one(x) & -three(x) & -four(x).
three(x) -> -one(x) & -two(x) & -four(x).
four(x) -> -one(x) & -two(x) & -three(x).

%Different digit on each square of a row
one(S_1_1) -> -one(S_1_2) & -one(S_1_3) & -one(S_1_4).
one(S_1_2) -> -one(S_1_1) & -one(S_1_3) & -one(S_1_4).
one(S_1_3) -> -one(S_1_1) & -one(S_1_2) & -one(S_1_4).
one(S_1_4) -> -one(S_1_1) & -one(S_1_2) & -one(S_1_3).
one(S_2_1) -> -one(S_2_2) & -one(S_2_3) & -one(S_2_4).
one(S_2_2) -> -one(S_2_1) & -one(S_2_3) & -one(S_2_4).
one(S_2_3) -> -one(S_2_1) & -one(S_2_2) & -one(S_2_4).
one(S_2_4) -> -one(S_2_1) & -one(S_2_2) & -one(S_2_3).
one(S_3_1) -> -one(S_3_2) & -one(S_3_3) & -one(S_3_4).
one(S_3_2) -> -one(S_3_1) & -one(S_3_3) & -one(S_3_4).
one(S_3_3) -> -one(S_3_1) & -one(S_3_2) & -one(S_3_4).
one(S_3_4) -> -one(S_3_1) & -one(S_3_2) & -one(S_3_3).
one(S_4_1) -> -one(S_4_2) & -one(S_4_3) & -one(S_4_4).
one(S_4_2) -> -one(S_4_1) & -one(S_4_3) & -one(S_4_4).
one(S_4_3) -> -one(S_4_1) & -one(S_4_2) & -one(S_4_4).
one(S_4_4) -> -one(S_4_1) & -one(S_4_2) & -one(S_4_3).

two(S_1_1) -> -two(S_1_2) & -two(S_1_3) & -two(S_1_4).
two(S_1_2) -> -two(S_1_1) & -two(S_1_3) & -two(S_1_4).
two(S_1_3) -> -two(S_1_1) & -two(S_1_2) & -two(S_1_4).
two(S_1_4) -> -two(S_1_1) & -two(S_1_2) & -two(S_1_3).
two(S_2_1) -> -two(S_2_2) & -two(S_2_3) & -two(S_2_4).
two(S_2_2) -> -two(S_2_1) & -two(S_2_3) & -two(S_2_4).
two(S_2_3) -> -two(S_2_1) & -two(S_2_2) & -two(S_2_4).
two(S_2_4) -> -two(S_2_1) & -two(S_2_2) & -two(S_2_3).
two(S_3_1) -> -two(S_3_2) & -two(S_3_3) & -two(S_3_4).
two(S_3_2) -> -two(S_3_1) & -two(S_3_3) & -two(S_3_4).
two(S_3_3) -> -two(S_3_1) & -two(S_3_2) & -two(S_3_4).
two(S_3_4) -> -two(S_3_1) & -two(S_3_2) & -two(S_3_3).
two(S_4_1) -> -two(S_4_2) & -two(S_4_3) & -two(S_4_4).
```

```
two(S_4_2) -> -two(S_4_1) & -two(S_4_3) & -two(S_4_4).
two(S_4_3) -> -two(S_4_1) & -two(S_4_2) & -two(S_4_4).
two(S_4_4) -> -two(S_4_1) & -two(S_4_2) & -two(S_4_3).

three(S_1_1) -> -three(S_1_2) & -three(S_1_3) & -three(S_1_4).
three(S_1_2) -> -three(S_1_1) & -three(S_1_3) & -three(S_1_4).
three(S_1_3) -> -three(S_1_1) & -three(S_1_2) & -three(S_1_4).
three(S_1_4) -> -three(S_1_1) & -three(S_1_2) & -three(S_1_3).
three(S_2_1) -> -three(S_2_2) & -three(S_2_3) & -three(S_2_4).
three(S_2_2) -> -three(S_2_1) & -three(S_2_3) & -three(S_2_4).
three(S_2_3) -> -three(S_2_1) & -three(S_2_2) & -three(S_2_4).
three(S_2_4) -> -three(S_2_1) & -three(S_2_2) & -three(S_2_3).
three(S_3_1) -> -three(S_3_2) & -three(S_3_3) & -three(S_3_4).
three(S_3_2) -> -three(S_3_1) & -three(S_3_3) & -three(S_3_4).
three(S_3_3) -> -three(S_3_1) & -three(S_3_2) & -three(S_3_4).
three(S_3_4) -> -three(S_3_1) & -three(S_3_2) & -three(S_3_3).
three(S_4_1) -> -three(S_4_2) & -three(S_4_3) & -three(S_4_4).
three(S_4_2) -> -three(S_4_1) & -three(S_4_3) & -three(S_4_4).
three(S_4_3) -> -three(S_4_1) & -three(S_4_2) & -three(S_4_4).
three(S_4_4) -> -three(S_4_1) & -three(S_4_2) & -three(S_4_3).

four(S_1_1) -> -four(S_1_2) & -four(S_1_3) & -four(S_1_4).
four(S_1_2) -> -four(S_1_1) & -four(S_1_3) & -four(S_1_4).
four(S_1_3) -> -four(S_1_1) & -four(S_1_2) & -four(S_1_4).
four(S_1_4) -> -four(S_1_1) & -four(S_1_2) & -four(S_1_3).
four(S_2_1) -> -four(S_2_2) & -four(S_2_3) & -four(S_2_4).
four(S_2_2) -> -four(S_2_1) & -four(S_2_3) & -four(S_2_4).
four(S_2_3) -> -four(S_2_1) & -four(S_2_2) & -four(S_2_4).
four(S_2_4) -> -four(S_2_1) & -four(S_2_2) & -four(S_2_3).
four(S_3_1) -> -four(S_3_2) & -four(S_3_3) & -four(S_3_4).
four(S_3_2) -> -four(S_3_1) & -four(S_3_3) & -four(S_3_4).
four(S_3_3) -> -four(S_3_1) & -four(S_3_2) & -four(S_3_4).
four(S_3_4) -> -four(S_3_1) & -four(S_3_2) & -four(S_3_3).
four(S_4_1) -> -four(S_4_2) & -four(S_4_3) & -four(S_4_4).
four(S_4_2) -> -four(S_4_1) & -four(S_4_3) & -four(S_4_4).
four(S_4_3) -> -four(S_4_1) & -four(S_4_2) & -four(S_4_4).
four(S_4_4) -> -four(S_4_1) & -four(S_4_2) & -four(S_4_3).

%Different digit on each square of a column
one(S_1_1) -> -one(S_2_1) & -one(S_3_1) & -one(S_4_1).
one(S_2_1) -> -one(S_1_1) & -one(S_3_1) & -one(S_4_1).
one(S_3_1) -> -one(S_1_1) & -one(S_2_1) & -one(S_4_1).
one(S_4_1) -> -one(S_1_1) & -one(S_2_1) & -one(S_3_1).
one(S_1_2) -> -one(S_2_2) & -one(S_3_2) & -one(S_4_2).
one(S_2_2) -> -one(S_1_2) & -one(S_3_2) & -one(S_4_2).
one(S_3_2) -> -one(S_1_2) & -one(S_2_2) & -one(S_4_2).
one(S_4_2) -> -one(S_1_2) & -one(S_2_2) & -one(S_3_2).
one(S_1_3) -> -one(S_2_3) & -one(S_3_3) & -one(S_4_3).
one(S_2_3) -> -one(S_1_3) & -one(S_3_3) & -one(S_4_3).
one(S_3_3) -> -one(S_1_3) & -one(S_2_3) & -one(S_4_3).
one(S_4_3) -> -one(S_1_3) & -one(S_2_3) & -one(S_3_3).
one(S_1_4) -> -one(S_2_4) & -one(S_3_4) & -one(S_4_4).
one(S_2_4) -> -one(S_1_4) & -one(S_3_4) & -one(S_4_4).
one(S_3_4) -> -one(S_1_4) & -one(S_2_4) & -one(S_4_4).
one(S_4_4) -> -one(S_1_4) & -one(S_2_4) & -one(S_3_4).
```

```
two(S_1_1) -> -two(S_2_1) & -two(S_3_1) & -two(S_4_1).
two(S_2_1) -> -two(S_1_1) & -two(S_3_1) & -two(S_4_1).
two(S_3_1) -> -two(S_1_1) & -two(S_2_1) & -two(S_4_1).
two(S_4_1) -> -two(S_1_1) & -two(S_2_1) & -two(S_3_1).
two(S_1_2) -> -two(S_2_2) & -two(S_3_2) & -two(S_4_2).
two(S_2_2) -> -two(S_1_2) & -two(S_3_2) & -two(S_4_2).
two(S_3_2) -> -two(S_1_2) & -two(S_2_2) & -two(S_4_2).
two(S_4_2) -> -two(S_1_2) & -two(S_2_2) & -two(S_3_2).
two(S_1_3) -> -two(S_2_3) & -two(S_3_3) & -two(S_4_3).
two(S_2_3) -> -two(S_1_3) & -two(S_3_3) & -two(S_4_3).
two(S_3_3) -> -two(S_1_3) & -two(S_2_3) & -two(S_4_3).
two(S_4_3) -> -two(S_1_3) & -two(S_2_3) & -two(S_3_3).
two(S_1_4) -> -two(S_2_4) & -two(S_3_4) & -two(S_4_4).
two(S_2_4) -> -two(S_1_4) & -two(S_3_4) & -two(S_4_4).
two(S_3_4) -> -two(S_1_4) & -two(S_2_4) & -two(S_4_4).
two(S_4_4) -> -two(S_1_4) & -two(S_2_4) & -two(S_3_4).

three(S_1_1) -> -three(S_2_1) & -three(S_3_1) & -three(S_4_1).
three(S_2_1) -> -three(S_1_1) & -three(S_3_1) & -three(S_4_1).
three(S_3_1) -> -three(S_1_1) & -three(S_2_1) & -three(S_4_1).
three(S_4_1) -> -three(S_1_1) & -three(S_2_1) & -three(S_3_1).
three(S_1_2) -> -three(S_2_2) & -three(S_3_2) & -three(S_4_2).
three(S_2_2) -> -three(S_1_2) & -three(S_3_2) & -three(S_4_2).
three(S_3_2) -> -three(S_1_2) & -three(S_2_2) & -three(S_4_2).
three(S_4_2) -> -three(S_1_2) & -three(S_2_2) & -three(S_3_2).
three(S_1_3) -> -three(S_2_3) & -three(S_3_3) & -three(S_4_3).
three(S_2_3) -> -three(S_1_3) & -three(S_3_3) & -three(S_4_3).
three(S_3_3) -> -three(S_1_3) & -three(S_2_3) & -three(S_4_3).
three(S_4_3) -> -three(S_1_3) & -three(S_2_3) & -three(S_3_3).
three(S_1_4) -> -three(S_2_4) & -three(S_3_4) & -three(S_4_4).
three(S_2_4) -> -three(S_1_4) & -three(S_3_4) & -three(S_4_4).
three(S_3_4) -> -three(S_1_4) & -three(S_2_4) & -three(S_4_4).
three(S_4_4) -> -three(S_1_4) & -three(S_2_4) & -three(S_3_4).

four(S_1_1) -> -four(S_2_1) & -four(S_3_1) & -four(S_4_1).
four(S_2_1) -> -four(S_1_1) & -four(S_3_1) & -four(S_4_1).
four(S_3_1) -> -four(S_1_1) & -four(S_2_1) & -four(S_4_1).
four(S_4_1) -> -four(S_1_1) & -four(S_2_1) & -four(S_3_1).
four(S_1_2) -> -four(S_2_2) & -four(S_3_2) & -four(S_4_2).
four(S_2_2) -> -four(S_1_2) & -four(S_3_2) & -four(S_4_2).
four(S_3_2) -> -four(S_1_2) & -four(S_2_2) & -four(S_4_2).
four(S_4_2) -> -four(S_1_2) & -four(S_2_2) & -four(S_3_2).
four(S_1_3) -> -four(S_2_3) & -four(S_3_3) & -four(S_4_3).
four(S_2_3) -> -four(S_1_3) & -four(S_3_3) & -four(S_4_3).
four(S_3_3) -> -four(S_1_3) & -four(S_2_3) & -four(S_4_3).
four(S_4_3) -> -four(S_1_3) & -four(S_2_3) & -four(S_3_3).
four(S_1_4) -> -four(S_2_4) & -four(S_3_4) & -four(S_4_4).
four(S_2_4) -> -four(S_1_4) & -four(S_3_4) & -four(S_4_4).
four(S_3_4) -> -four(S_1_4) & -four(S_2_4) & -four(S_4_4).
four(S_4_4) -> -four(S_1_4) & -four(S_2_4) & -four(S_3_4).

%Different digit on each square of a 2x2 grid
%Square S_1_1, S_1_2, S_2_1, S_2_2
one(S_1_1) -> -one(S_2_2) & -one(S_2_1) & -one(S_1_2).
one(S_1_2) -> -one(S_2_1) & -one(S_1_1) & -one(S_2_2).
one(S_2_1) -> -one(S_1_2) & -one(S_1_1) & -one(S_2_2).
```

```
one(S_2_2) -> -one(S_1_1) & -one(S_1_2) & -one(S_2_1).
two(S_1_1) -> -two(S_2_2) & -two(S_2_1) & -two(S_1_2).
two(S_1_2) -> -two(S_2_1) & -two(S_1_1) & -two(S_2_2).
two(S_2_1) -> -two(S_1_2) & -two(S_1_1) & -two(S_2_2).
two(S_2_2) -> -two(S_1_1) & -two(S_2_1) & -two(S_1_2).
three(S_1_1) -> -three(S_2_2) & -three(S_2_1) & -three(S_1_2).
three(S_1_2) -> -three(S_2_1) & -three(S_1_1) & -three(S_2_2).
three(S_2_1) -> -three(S_1_2) & -three(S_1_1) & -three(S_2_2).
three(S_2_2) -> -three(S_1_1) & -three(S_2_1) & -three(S_1_2).
four(S_1_1) -> -four(S_2_2) & -four(S_2_1) & -four(S_1_2).
four(S_1_2) -> -four(S_2_1) & -four(S_1_1) & -four(S_2_2).
four(S_2_1) -> -four(S_1_2) & -four(S_1_1) & -four(S_2_2).
four(S_2_2) -> -four(S_1_1) & -four(S_2_1) & -four(S_1_2).

%Square S_1_3, S_1_4, S_2_3, S_2_4
one(S_1_3) -> -one(S_1_4) & -one(S_2_3) & -one(S_2_4).
one(S_1_4) -> -one(S_2_3) & -one(S_1_3) & -one(S_2_4).
one(S_2_3) -> -one(S_1_4) & -one(S_1_3) & -one(S_2_4).
one(S_2_4) -> -one(S_1_3) & -one(S_1_4) & -one(S_2_3).
two(S_1_3) -> -two(S_1_4) & -two(S_2_3) & -two(S_2_4).
two(S_1_4) -> -two(S_2_3) & -two(S_1_3) & -two(S_2_4).
two(S_2_3) -> -two(S_1_4) & -two(S_1_3) & -two(S_2_4).
two(S_2_4) -> -two(S_1_3) & -two(S_1_4) & -two(S_2_3).
three(S_1_3) -> -three(S_1_4) & -three(S_2_3) & -three(S_2_4).
three(S_1_4) -> -three(S_2_3) & -three(S_1_3) & -three(S_2_4).
three(S_2_3) -> -three(S_1_4) & -three(S_1_3) & -three(S_2_4).
three(S_2_4) -> -three(S_1_3) & -three(S_1_4) & -three(S_2_3).
four(S_1_3) -> -four(S_1_4) & -four(S_2_3) & -four(S_2_4).
four(S_1_4) -> -four(S_2_3) & -four(S_1_3) & -four(S_2_4).
four(S_2_3) -> -four(S_1_4) & -four(S_1_3) & -four(S_2_4).
four(S_2_4) -> -four(S_1_3) & -four(S_1_4) & -four(S_2_3).

%Square S_3_1, S_3_2, S_4_1, S_4_2
one(S_3_1) -> -one(S_4_2) & -one(S_3_2) & -one(S_4_1).
one(S_3_2) -> -one(S_4_1) & -one(S_3_1) & -one(S_4_2).
one(S_4_1) -> -one(S_3_2) & -one(S_3_1) & -one(S_4_2).
one(S_4_2) -> -one(S_3_1) & -one(S_3_2) & -one(S_4_1).
two(S_3_1) -> -two(S_4_2) & -two(S_3_2) & -two(S_4_1).
two(S_3_2) -> -two(S_4_1) & -two(S_3_1) & -two(S_4_2).
two(S_4_1) -> -two(S_3_2) & -two(S_3_1) & -two(S_4_2).
two(S_4_2) -> -two(S_3_1) & -two(S_3_2) & -two(S_4_1).
three(S_3_1) -> -three(S_4_2) & -three(S_3_2) & -three(S_4_1).
three(S_3_2) -> -three(S_4_1) & -three(S_3_1) & -three(S_4_2).
three(S_4_1) -> -three(S_3_2) & -three(S_3_1) & -three(S_4_2).
three(S_4_2) -> -three(S_3_1) & -three(S_3_2) & -three(S_4_1).
four(S_3_1) -> -four(S_4_2) & -four(S_3_2) & -four(S_4_1).
four(S_3_2) -> -four(S_4_1) & -four(S_3_1) & -four(S_4_2).
four(S_4_1) -> -four(S_3_2) & -four(S_3_1) & -four(S_4_2).
four(S_4_2) -> -four(S_3_1) & -four(S_3_2) & -four(S_4_1).

%Square S_3_3, S_3_4, S_4_3, S_4_4
one(S_3_3) -> -one(S_4_4) & -one(S_3_4) & -one(S_4_3).
one(S_3_4) -> -one(S_4_3) & -one(S_3_3) & -one(S_4_4).
one(S_4_3) -> -one(S_3_4) & -one(S_3_3) & -one(S_4_4).
one(S_4_4) -> -one(S_3_3) & -one(S_3_4) & -one(S_4_3).
two(S_3_3) -> -two(S_4_4) & -two(S_3_4) & -two(S_4_3).
```

```
two(S_3_4) -> -two(S_4_3) & -two(S_3_3) & -two(S_4_4).
two(S_4_3) -> -two(S_3_4) & -two(S_3_3) & -two(S_4_4).
two(S_4_4) -> -two(S_3_3) & -two(S_3_4) & -two(S_4_3).
three(S_3_3) -> -three(S_4_4) & -three(S_3_4) & -three(S_4_3).
three(S_3_4) -> -three(S_4_3) & -three(S_3_3) & -three(S_4_4).
three(S_4_3) -> -three(S_3_4) & -three(S_3_3) & -three(S_4_4).
three(S_4_4) -> -three(S_3_3) & -three(S_3_4) & -three(S_4_3).
four(S_3_3) -> -four(S_4_4) & -four(S_3_4) & -four(S_4_3).
four(S_3_4) -> -four(S_4_3) & -four(S_3_3) & -four(S_4_4).
four(S_4_3) -> -four(S_3_4) & -four(S_3_3) & -four(S_4_4).
four(S_4_4) -> -four(S_3_3) & -four(S_3_4) & -four(S_4_3).

%Configuration
one(S_1_1).
three(S_1_3).
three(S_2_2).
two(S_2_3).
two(S_3_1).
four(S_3_3).
four(S_4_2).
two(S_4_4).
```

5 Mace4 result for the given configuration

```
interpretation( 4, [number = 1,seconds = 0], [
function(S_1_1, [0]),
function(S_1_2, [1]),
function(S_1_3, [2]),
function(S_1_4, [3]),
function(S_2_1, [3]),
function(S_2_2, [2]),
function(S_2_3, [1]),
function(S_2_4, [0]),
function(S_3_1, [1]),
function(S_3_2, [0]),
function(S_3_3, [3]),
function(S_3_4, [2]),
function(S_4_1, [2]),
function(S_4_2, [3]),
function(S_4_3, [0]),
function(S_4_4, [1]),
relation(digit(_), [1,1,1,1]),
relation(four(_), [0,0,0,1]),
relation(one(_), [1,0,0,0]),
relation(three(_), [0,0,1,0]),
relation(two(_), [0,1,0,0])]).
```

The given puzzle configuration:

1		3	
	3	2	
2		4	
	4		2

The solution provided by Mace4:

1	2	3	4
4	3	2	1
2	1	4	3
3	4	1	2