CMPT 477 Program Assignment 2 description

Bowei Pan

301435285

This graph coloring program uses the Z3 solver to solve the graph coloring problem. Below is a brief explanation of the program's code.

1. Input reading:
   a) The program first reads the input file (.txt format), which should contain a 9*9 number sequence, with each number separated by a space, and the number range is integers from 0 to 9 (except for 0, there can be no duplication). Then the program creates a Z3 array list to store the value of each position (except for 0). Since Z3 can only store one-dimensional arrays, the index is from 0 to 80. (The index of the first number in the second line is 9).
2. Adding Constrain
   a) **Each element should between 1 – 9**: For all the empty spaces in the array, their values must be between 1 and 9. If they are outside this range, the solution cannot be found.
   b) **Each row must contain each of the digits 1– 9 exactly once**: Based on the rules of Sudoku, the 9 elements in each row must be 1-9, and each

element must appear once (no repetition)

c) **Each column must contain each of the digits 1– 9 exactly once**: Based on the rules of Sudoku, the 9 elements in each column must be 1-9, and each element must appear once (no repetition)

d) **Each of the nine 3×3 blocks (see Figure 1) must contain each of the digits 1– 9 exactly once**: Based on the rules of Sudoku, the 9 elements in each 3*3 block must be 1-9, and each element must appear once (no repetition)

Encoding description:

The program reads the initial state of the Sudoku board from the input file. The input format is as follows:

Each line contains 9 integers ranging from 0 to 9 (except 0, no duplication), a total of nine lines. Represents each grid of the 9x9 Sudoku board, 0 represents the number to be inserted, and other numbers represent the number already filled in.

The grids are separated by spaces.

Design choices:

Design Choices:

Based on its capability to efficiently solve constraint satisfaction problems, this program utilizes Z3 to tackle the Sudoku puzzle. The problem is encoded using Z3 integer array, where each variable Si,j represents the value assigned to the cell in the i-th row and j-th column of the Sudoku grid. The program implements the following constraints:

- **Cell Value Range:** Ensures that each cell contains a value between 1 and 9, inclusive.
- **Unique Values per Row:** Guarantees that no two cells in the same row have the same value, which is essential for the validity of the Sudoku solution.
- **Unique Values per Column:** Ensures that no two cells in the same column share the same value, maintaining the integrity of the Sudoku rules.
- **Unique Values in 3x3 Subgrids:** Confirms that each 3x3 subgrid contains distinct values, further adhering to the Sudoku requirements.

Issue:

When I first started writing this program, I wanted to create a 2Darray in Z3, but it never worked. I later found out that Z3 only has one-dimensional arrays. If you want to represent a two-dimensional array, you need to expand the two-dimensional array. When I was writing constraints, I encountered an ArithExpr problem. I later learned that the return type of mkSelect in Z3 is Expr, not ArithExpr. When this function is used with other Z3 arithmetic functions (such as mkLe), there will be a type mismatch problem (because mkLe expects the type to be ArithExpr), so I used type coercion to solve this problem.