Lab 3： 基于推理的人工智能：数独求解器

231880101 孙俊晖

1 引言

1.1 背景

逻辑编程是一种编程范式，它通过设置答案必须满足的规则来解决问题，事实 + 规则 = 结果。而数独作为一种经典的数字游戏，玩家需要根据9×9盘面上的已知数字，在数独规则的约束下，推理出所有剩余空格的数字。9×9盘面上的已知数字即为事实，满足每一行、每一列、每一个九宫格内的数字均含1-9且不重复即为数独的规则，得出所有剩余空格的数字即为结果。由此可见，逻辑编程非常适合用来求解数独问题。

1.2 目标

利用Python中的逻辑求解器Z3实现一个数独求解器，对于输入的任意合法数独问题，若数独可解则输出一个解，否则输出"fail to solve"

2 方法

2.1 技术栈

开发环境：Visual Studio Code 1.93.0

编程语言：Python 3.7.9 64-bit

2.2 实现数独求解器

2.2.1 导入所需的Python库

导入z3库

```
from z3 import *
```

2.2.2 建立数独问题的约束

首先，为数独的每一个单元格定义一个整数变量(第i行第j列的单元格的变量名为x_i_j)：

```
X = [ [ Int("x_%s_%s" % (i+1, j+1)) for j in range(9) ] for i in range(9) ]
```

接着，根据数独的规则建立约束，共4个约束：

（1）每个单元格中的数字在1-9之间

```
num_valid = [ And(1 <= X[i][j], X[i][j] <= 9) for i in range(9) for j in
range(9) ]
```

（2）每一行含1-9且不重复

```
rows_valid = [ Distinct(X[i]) for i in range(9) ]
```

（3）每一列含1-9且不重复

```
cols_valid = [ Distinct([ X[i][j] for i in range(9) ]) for j in range(9) ]
```

（4）每一个九宫格内含1-9且不重复

```
squares_valid = [ Distinct([ X[3*dx + i][3*dy + j]for i in range(3) for j in
range(3) ]) for dx in range(3) for dy in range(3) ]
```

对于一个合法的数独来说，它需要同时满足上述四个约束，那么将上述四个约束相加即为对合法数独的约束：

```
sudoku_valid = num_valid + rows_valid + cols_valid + squares_valid
```

2.2.3 输入待求解的数独问题

输入待求解的数独问题的格式为：一共输入9行，每行9个数字（输入的数独中空位置填入0），同一行中相邻的两个数用空格隔开

用一个二维列表sudoku_input来存储输入的数独问题的状态：

```
sudoku_input = [[0 for _ in range(9)] for _ in range(9)]
for i in range(9):
    line = input().strip()
    numbers = line.split()
    for j in range(9):
        num = int(numbers[j])
        sudoku_input[i][j] = num
```

根据输入的数独问题的状态，为每一个非空单元格添加一个约束sudoku_solve_rule，要求非空单元格的变量值等于输入的值，即在数独求解器求解数独问题时必须保持原有的非空单元格的值不变：

```
sudoku_solve_rule = [ If(sudoku_input[i][j] == 0,True,X[i][j] ==
sudoku_input[i][j])
for i in range(9) for j in range(9) ]
```

2.2.4 利用Z3求解器求解该数独问题

首先，创建一个Z3求解器的实例s,并为s添加上面定义的约束sudoku_valid和sudoku_solve_rule

```
s = Solver()
s.add(sudoku_valid + sudoku_input_rule)
```

接着，检查求解器是否能找到一个满足所有约束的解，如果能的话，就输出找到的解，输出格式为输出9行，每行9个数，同一行中相邻两个数之间用空格隔开；如果找不到一个满足所有约束的解的话，输出"fail to solve"

```
if s.check() == sat:
    m = s.model()
    r = [ [ m.evaluate(X[i][j]) for j in range(9) ] for i in range(9) ]
    for line in r:
      print(' '.join(map(str, line)))
else:
    print ("failed to solve")
```

3 总结

纵观人工智能的发展历程，人工智能的发展与逻辑编程的发展是一个相辅相成的过程，而早期的人工智能以规则和逻辑推理作为主要研究方向，由此可见，了解和学习逻辑编程是很有必要的。在本次实验中，通过逻辑编程的方法，我成功地利用Python中的逻辑求解器Z3实现了一个数独求解器。通过本次实验，我不仅掌握了利用逻辑求解器求解数独问题的方法，还进一步理解了逻辑编程的范式和原理。同时，我也进一步体会到了Python这一编程语言的强大功能和灵活性。

4 参考文献

1.https://ericpony.github.io/z3py-tutorial/guide-examples.htm