Assignment #3: Sudoku (Constraint Satisfaction Problems)

CSCI 364 Spring 2017 Oberlin College
Due: Monday March 13 at 11:00 AM

Background

In class, we used Sudoku puzzles as an illustrative example of Constraint Satisfaction Problems (CSPs) in AI, where the goal is to find a complete, consistent assignment of values to a set of variables X (taken from their domains D) satisfying a set of constraints C that limit the valid combinations of variable values. In this assignment, you will have an opportunity to develop a program using CSP solution techniques to solve Sudoku puzzles.

As a reminder, a Sudoku puzzle is a 9x9 grid (81 variables) where each cell in the grid can take on the integer values 1-9 (the domains of the variables). A solution to a Sudoku puzzle is an assignment of values for each cell in the grid such that no two cells in the same row/column/3x3 box have the same value.

For example, for an initial configuration of a Sudoku puzzle, you might be given:

```
..3|.2.|6..
9..|3.5|..1
..1|8.6|4..
-----------
..8|1.2|9..
7..|...|..8
..6|7.8|2..
-----------
..2|6.9|5..
8..|2.3|..9
..5|.1.|3..
```

which has the solution:

```
483|921|657
967|345|821
251|876|493
-----------
548|132|976
729|564|138
136|798|245
-----------
372|689|514
814|253|769
695|417|382
```

Assignment

Your assignment is to write a program in **Python or Java** that can take a set of Sudoku puzzles as input from a file and output solutions to each. You should make use of both Constraint Propagation (e.g., AC3) and Backtracking Search algorithms as part of your solution.

You can get started on the assignment by following this link:
https://classroom.github.com/assignment-invitations/4b1236d81cfcef09e2ec9603742a38b8

Unlike the previous two homework assignments, there is no code base to start with. Instead, you are free to develop your program and represent your data structures however you wish. Your GitHub repository will only initially contain two files of Sudoku puzzles:

1. euler.txt, a set of Sudoku puzzles from Project Euler https://projecteuler.net/problem=96

2. magictour.txt, a more difficult set of Sudoku puzzles from http://magictour.free.fr/top95

Each file contains a single Sudoku puzzle on each line, in the following format:

- Each line is a string of 81 characters, where characters in positions 0-8 correspond to the first row of the puzzle, characters in positions 9-17 correspond to the second row of the puzzle, etc.
- Known values are represented by the digits 1-9
- Initially unknown values are represented by a decimal point .

Your program should be able to read in these puzzles, solve them, then output the solutions in the same format (a string of 81 digits, followed by a newline character) in the same order they were read in from file.

Within a README file, you should include:

1. The output of your program on all puzzles from both input files (please add a header saying which file each set of solutions corresponds with),
2. A short paragraph describing your experience during the assignment (what did you enjoy, what was difficult, etc.)
3. An estimation of how much time you spent on the assignment, and
4. An affirmation that you adhered to the honor code

Please remember to commit your solution to your repository on GitHub. You do not need to wait until you are done with the assignment; it is good practice to do so not only after each coding session, but maybe after hitting important milestones or solving bugs during a coding session. ***Make sure to document your code***, explaining how you implemented the CSP as a data structure, as well as the algorithms used to solve the CSP.

<u>Honor Code</u>

Each student is to complete this assignment individually. However, students are encouraged to collaborate with one another to discuss the abstract design and processes of their implementations. For example, please feel free to discuss the pseudocode for the Constraint Propagation and Backtracking Search algorithms to help each other work through issues understanding exactly how the algorithms work. At the same, since this is an individual assignment, no code can be shared between students, nor can students look at each other's code.

And please be careful – there are many Sudoku solvers with freely available code online. This is a popular assignment and fun exercise for many programmers. It is easy to accidentally stumble upon someone else's solution, so please do not look at or copy anyone else's code (which I don't expect to be a problem!).