Informed Search Methods in AI

Fall 2020 Assignment 1

In this assignment you will implement a generic *CSP solver* and then use it to solve *SuDoKu* puzzles. A skeleton of a *Simple- CSP solver* is provided. It currently uses only *Generate-and-test backtracking* for solving but your project is to augment and improve it using some the techniques we have been studying. More specifically, you should implement the following backtracking-based search algorithms:

- *Chronological Backtracking (BT)*
- *Backjumping search (BJ)*
- *Conflict-directed Backjumping search (CBJ*)

as well as arc-consistency using the *AC-3* algorithm.

Run the different algorithms on the test-suite provided (it will be made available at a later date) to compare their run-time efficiency, both with and without forcing arc-consistency on the constraint network. Write a brief report (2-3 pages) that summarizes your experimental result. Include the experimental data in there (i.e. whether a problem instance was solved, how many nodes expanded, number of constraint checks, and running time). Also briefly discuss how you think the solver could be further improved.

PART 1

Your first task is to convert Sudoku puzzles into constraint networks in a format recognized by the *Simple-CSP* solver. You are provided with a skeleton program for doing this (*sudoku.py*), but you need to implement the missing routines (see code). Run the program on the puzzles provided (in *sodoko.txt)* as well as on any additional Sudoku puzzle instances you might come across (you can add them to *sodoko.txt* or have them in a separate file). See code for details.

PART II

In the second part you implement the missing backtracking-based search algorithms in the solver (*solvers.py*). Also add *arc consistency* pre-processing to the constraint network by implement the missing routine (also in solvers.*py*). You can start this part even though the first part is not finished because example constraint/domain files are provided (generated from the puzzle instances in *sudoku.txt*). See code for details. Hint: when initially testing your backtracking algorithms it might be a good idea to do so on trivial CSP problem (the *Simple-CSP* solver can take in specification of any problem that uses only binary not-equal constraints).

Hand in the two python programs you modified (*sudoku.py* and *solvers.py*) as well as your report (as a PDF file called *report.pdf*).