Progress Report

Team Windwolf

Roldan Gammad, 29423247

Sai Smaran Macha, 92269171

1 SCOPE

We have implemented token reading and forward checking for solving monster sudoku problems. All code is implemented in Java using the provided Java coding shell.

2 PROGRESS

We have completed the implementation of token reading and forward checking. We have also moved time out to BTSolver and altered our outputlog to properly read (fake) preprocessing start/end times, and altered the format of our output.

In the main file: we set all token arguments to false at the beginning of the program. We iterate through the given arguments, separating the necessary arguments (input & output file, timeout time) from the token arguments (FC, DH, etc…). We then iterate through all of the token arguments, checking if any of them are present, if they are, we set them to true. We then check which token argument is true and change the value and variable heuristic enums and consistency check enum accordingly (from BTSolver).

In the BTSolver file: we implemented forward checking functionality. We added timeout to the solve() function of the BTSolver, checking if timeout is greater than preprocessing time and current solving time. We also set the timeout status to true when a timeout exception is thrown here, so that we can change the STATUS output to timeout if a timeout error occurs.

In the outputLog file: we included DoubleFormat so that we can print out our seconds in decimal. We also subtracted all of the times we have recorded from System.currentTimeMillis() so that we receive relative time rather than system time in our output. We also altered the format of our output to be closer to the specified output in the SudokuTester ver0.2 README.txt.

APPENDIX

1. Forward Checking

private boolean forwardChecking()

{

for(Variable v: network.getVariables()){

if(v.isAssigned()){

for(Variable vOther: network.getNeighborsOfVariable(v)){

vOther.removeValueFromDomain(v.getAssignment());

if(vOther.getDomain().size() == 0){

return false;

}

}

}

}

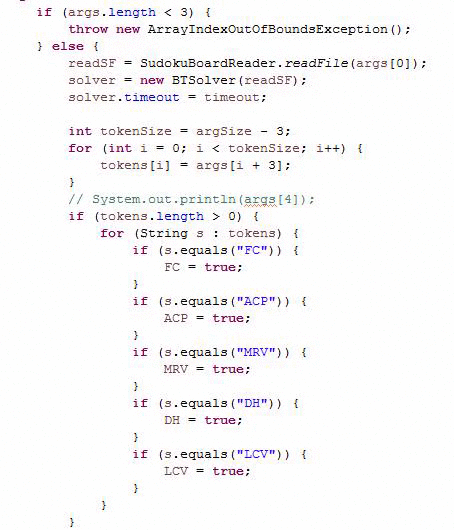
return true;

}

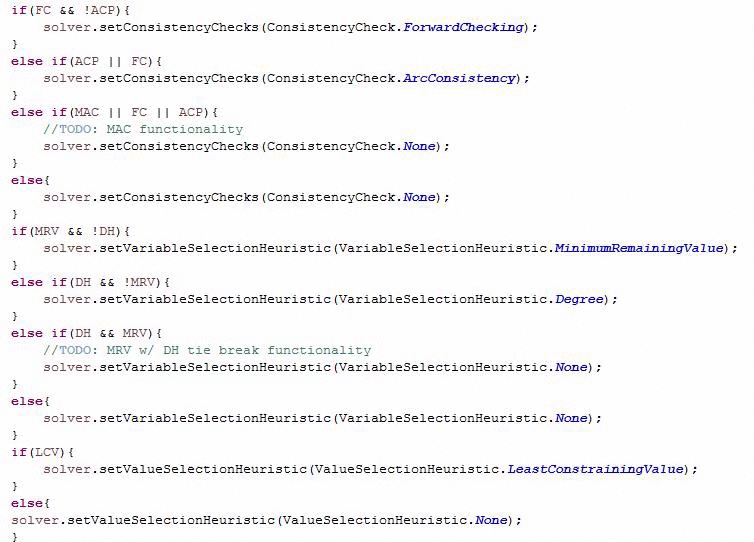
2. Check if args has less than the minimum required arguments (3: hopefully input filename, output file name, and timeout time)

if passes, iterate through arguments, move tokens (any element in argument found after the first 3 elements) to a new array (of size argument length - 3).

Iterate through new token array and check if any tokens exist, if they do, set bools to true.



3..Check which tokens have been set to true. set Consistency Check, Variable Heuristics, and Value Heuristics to corresponding tokens.



4.Included DecimalFormat. set decimal format to 3 decimal places

set times to relative time

