CISC 440 - Artificial Intelligence & Robotics
Spring 2020

Assignment 05: Due Sunday, 4/5 by 11:59pm

For this assignment, you will begin a hybrid agent using the Java constraint solver "choco-solver" that plays the game "StudentWorld". Begin by forking and cloning the provided assignment as described in the "Assignment Setup and Submission" document in Canvas. You will need to finish the implementation of the StudentWorldConstraintSolver class.

StudentWorld

StudentWorld PEAS task environment description:

Performance measure
- Escaping through the exit door in the fewest number of moves without being consumed by a student.

Environment
- 5x5 grid of cells with 1 player starting in the upper left cell (0,0), 5 randomly selected cells containing students, and 1 randomly selected door. Neither a student nor the door can be placed in (0,0), (0,1), or (1,0). The student and the door cannot be placed in the same cell.

Actuators
- The player can move up, right, down, or left.

Sensors
- All cells adjacent to a student will smell. All cells adjacent to the door will glow. The size of the grid is known. The CSPPlayer knows which cells have been visited.

StudentWorldConstraintSolver (studentworld)

smellyCells, glowyCells, studentCells, doorCells : BoolVar[][]

These instance variables refer to 5x5 arrays of variables indicating whether the corresponding cell contains a smell, glow, student, and/or door (respectively). They are already declared and initialized in the provided initializeVariables method. You will need to access and/or update them in the createConstraints and updateCurrentCellVariables methods.

createConstraints() : void

This method will use the choco-solver library to create the required constraints for the constraint solver. My solution calls multiple functions to create different categories of constraints. Many of my constraints take the form:

```
model.ifThen(someCells[row][col], model.arithm(otherCells[row][col], "=", 1));
```

This statement may be read as "if someCells[row][col] is true, then assign a value of 'true' to otherCells[row][col]". In choco-solver, BoolVars are basically IntVars with two possible values: 0 or 1.

Other constraints in my solution make use of the model's "sum" method, as in:

```
model.sum(someCellArray, "=", 1).post();
model.ifThen(someCells[row][col], model.sum(otherCellArray, "=", 1));
```

These statements may be read as "1 and only 1 of the variables in 'someCellArray' can be true" and "if 'someCells[row][col]' is true, then 1 and only 1 of the variables in 'otherCellArray' can be true".

updateCurrentCellVariables() : void

This method will use the choco-solver library to assign values to variables determined by the player's current location. This method is called before each player move to update the constraint solver with the most recent environment information. It should post choco-solver constraints relating to the state of the cell the player is currently in. Many of my assignments take the form:

```
model.arithm(someCells[row][col], "=", 0).post();
```

This statement may be read as "assign a value of 'false' to someCells[row][col]".

Odds and Ends

- Each time the "propagate" method is called on StudentWorldConstraintSolver's "solver" instance variable, the solver will entail new variable assignments based on the existing assignments and constraints. So when a new cell is visited, variable assignments are made (by executing code you write in updateCurrentCellVariables), the propagate method is called, and the solver uses the constraints to assign new values to variables (if possible).

- The "printAllVariables" method is provided to help with your debugging.

Your StudentWorldConstraintSolver.java file will be used along with the other provided files when grading. Your grade be determined largely by the number of tests passed in StudentWorldConstraintSolverTest.java. While these tests are a good indicator of correctness, they do not guarantee a fully correct implementation. In other words, passing all tests may not translate to a 100% on the assignment.

Submit your assignment by pushing your code to your gitlab repository fork by the deadline.