CSSE 373 - Formal Methods in Spec. and Design

Lab 7

Purpose

Get more experience in applying formal methods tools on interesting problems. In particular, learn how to formulate real-world problems to the forms that a SAT solver can understand and in the process, learn more about Yices.

Download the attached Eclipse project (zip) and import it in your Eclipse IDE.

Also, download a Yices bundle from: http://yices.csl.sri.com. Copy the bundle to a directory where you store software that has no installer. You will need to **extract the bundle and add the full path to its bin directory to the PATH environment variables**. Here is a link that explains how to set the PATH variable for Windows (http://www.computerhope.com/issues/ch000549.htm).

Problem 1: Solving Linear Programming Problem (LPP) [10 points]

Go to **Slide #11** of the **7-1 Program Verification and SAT Solving** slides deck. Model the LPP in the **module/example1-lpp.ys** file. [**5 points**]

The example only generates one solution, and the solution is by no means optimal. Explore the **src/lpp** package to understand how the **LppCegarProcedure** works. Create **turnins/Lab7-1.pdf** and append a paragraph (or bullet points) describing how the procedure works. [**3 points**]

Run the **LppCegarRunner** and append the snapshot of the optimal solution generated by the program to **Lab7-1.pdf**. [**2 points**]

Problem 2-4: Programming Verification (PV) [20 points]

Go to the **src/pv** package. Explore the **Examples2** class. Create a Yices file (**modules/example2-min.ys**) to verify the correctness of the given method. Yices should either return a counterexample to denote that the program has a bug or return unsat to denote that the program is correct. **Append the snapshot of what yices return to Lab 7-1.pdf**. [**3 points**]

Also complete the **TODOs** in **Example2.java** as specified in the source file. [**2 points**]

Repeat the process for **Example3** [**3+2 = 5 points**]

Repeat the process for **Example4** [**7+3 = 10 points**]

Problem 5: Sudoku Solver [20 points]

Go to the **src/sudoku** package. Explore the three classes provided. Also explore **data/sudoku.txt**, it is claimed to be the world's hardest Sudoku problem (created by a Finnish mathematician, Arto Inkala, in 2012). Your job is to auto generate Yices code to solve this Sudoku problem. When you are done, you will have created a universal Sudoku solver that works for any Sudoku puzzle given that the input format is satisfied.

You may quickly run **SudokuSolverApp** without changing anything just to see what happens. It will print out verbose output as you will notice that the **SudokuSolverApp.DEBUG** flag is enabled at the moment. Refresh the project and you should see auto-generated **modules/Sudoku.ys** and **data/solution-sudoku.txt** files. There is not much happening in these two files and your job is to fix that by generating correct output. The only class that needs to change to generate the correct output is **ConcreteSudokuSolver**.

Submission

You can discuss these problems with your classmates but everyone must turn-in their own work. Bundle the Eclipse project [zip **– not rar**] and turn it in on Moodle. Check that your bundle has **turnins/Lab7-1.pdf** in it.