Test Report - Question 5

Tuba Kaya Chomette, Sander Leer, Martijn Stegeman
6 October 2013

We first defined the pre and post conditions for three functions:

- `randomSudoku` (generates a random Sudoku, which is consistent with normal Sudoku and NRC constraints)
- `genProblem` (takes the randomly generated Sudoku as parameter and generates the minimal problem for it with a unique solution)
- `solveNs` (takes the generated problem and finds the unique solution for it).

Below are the pre and post conditions of these functions. We implemented these conditions in file "`Week5Sol_Q5.hs`". We tried to use the assert and post wrappers to easily convert the "`rsolveNs [emptyN]`" function to an assertive version, but as IO is involved, it is not possible to automatically extract the resulting value for the wrapper. Because of this, we embedded the if-else syntax from the wrapper into the `randomSudoku'` and the `genProblem'` functions.

You can run function "`main`" to see that new functions with assertions and if conditions generating correct output.

Pre and Post conditions:

1. Generating a solved NRC Sudoku

   Post conditions :

   - Rows are injective
   - Columns are injective
   - Subgrids are injective
   - NRC subgrids are injective
   - No blanks

2. Generating a problem from the solved NRC Sudoku

   Pre conditions : Solved NRC Sudoku

   Post conditions:

   - Values given in the problem must be found on the same (row,column) in the solved Sudoku
   - The solution to the problem must be unique
   - If we try to remove any of the given numbers and solve the new problem, we must not get a unique solution. So the given values in the problem must be minimal to have a unique solution.

3. Solving the generated problem

   Preconditions: generated NRC problem with a unique solution

Post conditions:

- Rows are injective
- Columns are injective
- Subgrids are injective
- NRC subgrids are injective
- No blanks
- Values given in the problem are not changed
- The solution to the problem is unique