2D Challenge – 50.004: Introduction to Algorithms – Sept-Dec 2017 term

Preamble

In the 50.004 course, you are learning how to devise, describe, and analyze algorithms for given problems. You have learnt about SAT solvers in 50.001. For the 50.004 component of the 2D challenge you need to implement a 2-SAT solver. A 2-SAT problem restricts formulas to have clauses with at most 2 literals. For more information on 2-SAT, please refer to http://en.wikipedia.org/wiki/2-satisfiability

Your tasks

1. Design and implement in Java an algorithm that solves the 2-SAT problem in polynomial time. (*If you are a student from the ESD pillar taking the 50.004 course, you may use Java or Python for programming.*) A suggestion is to use an algorithm based on analyzing the strongly connected components of the implication graph, as described in Wikipedia in http://en.wikipedia.org/wiki/2-satisfiability.
   You can also use this method to find some satisfying solution.
   a. Your program must accept an input file in the CNF format as specified in https://app.box.com/s/opgbiv7310szl97nzv5devu3gi7qzbe3 (See Section 2.1).
   b. Your program must print the result of the satisfiability analysis on the console: print "FORMULA SATISFIABLE" or "FORMULA UNSATISFIABLE".
   c. If the formula is satisfiable, your program must also print a solution on the console. For example, if you are evaluating the formula *(x_1 or x_2) and (x_2 or x_3) and (x_3 or x_4) and (not x_1 or not x_3) and (not x_2 or not x_4)*, and you found a set of values for the four variables that leads to the whole formula being true to be *x_1=false, x_2=true, x_3=true, x_4=false*, then you produce the output

      ```
      0 1 1 0
      ```

   d. While evaluating your program, we will supply one or more input files that your program must accept and read. If this input file is not in the CNF form, the program needs to print the output, "INVALID INPUT".

2. Bonus: Implement a randomizing algorithm to check for satisfiability as outlined in http://www.fas.harvard.edu/~libcs124/CS/lec15.pdf. We like to run both algorithms on the same input and compare their results in terms of efficiency.

Your deliverables

1. A short report on your 2-SAT algorithm implementation and performance analysis.
2. Submission of the code with instructions for running it.
3. Bonus: In case you implemented the randomized algorithm, a short report (couple of pages) comparing the performance of the two algorithms on similar inputs. Is the randomizing algorithm a practical substitute for the deterministic one?

ALL DEADLINES ARE AS MENTIONED AT THE 2D CHALLENGE WEBSITE
http://50-002.wikispaces.com/2D+Design+Challenge+2017