Logical reasoning and programming, lab session II

(October 8, 2018)

II.1 Derive the empty clause from $\{\{\overline{a}, b\}, \{\overline{b}, c\}, \{a, \overline{c}\}, \{a, b, c\}, \{\overline{a}, \overline{b}, \overline{c}\}\}$ using resolution.

II.2 Define constraints *at least one* and *at most one* in CNF and discuss various variants of them.

II.3 Formulate graph coloring (a vertex coloring) as a SAT problem. Namely, given a graph G, does G admit a proper vertex coloring with k colors?

Discuss various possibilities how to formulate the problem. Moreover, are really all the constraints necessary?

II.4 If you want to play with SAT solving a bit, then a standard exercise is to formalize Sudoku as a SAT problem and hence produce a Sudoku solver. Write a program that generates a problem specification in the DIMACS format in such a way that it is possible to specify an input (a partially completed grid) by appending[1] clauses saying which variables are true. You can use MiniSat and some input is available from here.

You can try various cardinality constraints, e.g., the one based on binary encoding that requires $\mathcal{O}(n \log n)$ clauses and $\mathcal{O}(\log n)$ new variables.

By the way, is it possible to obtain also a generator of Sudoku puzzles this way?

[1]Note that this changes the number of clauses, a parameter specified in the DIMACS format.