Metaheuristics, 2024/2025

Fernando Lobo

Assignment details

- Grading: this assignment is worth 15% of the practical grade.
- Submission deadline: 04/Oct/2024, until 23:59.
- Submission procedure: submit a ZIP file via the tutoria at `https://tutoria.ualg.pt/2024/` . The ZIP file must contain the source code that you wrote, and a file named `Answers.pdf` containing written answers for the questions below. (Email submissions won't be accepted)
- Discussion: individual discussion of the assignment will take place on the lab lectures the following week: 08/Oct and 10/Oct. These discussions are mandatory.

Description

The purpose of this assignment is to let you have hands-on experience with a classical combinatorial optimization problem: MAXSAT.

MAXSAT stands for Maximum Satisfiability Problem and is the problem of maximizing the number of clauses of a boolean formula in conjunctive normal form (CNF) that evaluate to True, given an assignment of boolean values to the variables of the formula.

1. Visit `https://www.cs.ubc.ca/~hoos/SATLIB/benchm.html` and get familiar with the DIMACS cnf format explained there. Then download the first set of Uniform Random-3-SAT instances: `uf20-91`. Consider the first such instance: `uf20-01.cnf` and make sure you understand the format.
2. The following is a SAT instance in CNF format, containing 5 variables and 6 clauses, taken from page 20 of Hoos and Stützle's book. Specify this instance in the DIMACS format, and name it as `hoos.cnf`

$$
\begin{aligned}
F = & (\neg x_1 \vee x_2) \\
& \wedge (\neg x_2 \vee x_1) \\
& \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \\
& \wedge (x_1 \vee x_2) \\
& \wedge (\neg x_4 \vee x_3) \\
& \wedge (\neg x_5 \vee x_3)
\end{aligned}
$$

3. Write a computer program to read a DIMACS cnf instance from a text file. Then, write an algorithm to systematically visit the entire search space of such an instance and to report all solutions that satisfy it, i.e. that evaluate the formula to True. Your program must be written in a general way. That is, it should be able to receive as input any instance in the DIMACS cnf format.

4. Run your algorithm on the `hoos.cnf` instance, and list all solutions that satisfy the problem instance.

5. Run your algorithm on the `uf20-01.cnf` instance. How many solutions satisfy the problem instance?

6. It is impractical to do the same for a larger instance, say one contained in `uf100-430`. How could you estimate the time needed by your computer program to complete the task on such an instance? Justify your answer.