Metaheuristics, 2025/2026
Problem assignment 2

Fernando Lobo

Assignment details

- Grading: this assignment is worth 25% of the practical grade.
- Submission deadline: 24/Oct/2025, until 23:59.
- Submission procedure: submit a ZIP file via the tutoria at `https://tutoria.ualg.pt/2025/` . The ZIP file must contain the source code that you wrote, and a file named `Report.pdf` reporting the results that you obtained. (Email submissions won't be accepted)
- Discussion: individual discussion of the assignment will take place on the lab lectures the following week: 30/Oct. These discussions are mandatory.

Description

The purpose of this assignment is to have you implement and test hillclimbing algorithms. You will be working with instances of the MAXSAT problem.

You should run your algorithms on the following DIMACS instances, available at the tutoria:

- `uf20-01.cnf` : 20 variables, 91 clauses
- `uf100-01.cnf` : 100 variables, 430 clauses
- `uf250-01.cnf` : 250 variables, 1065 clauses

1. Implement *Next Ascent Hillclimbing* using a 1-bit *Hamming Distance* neighbourhood, and test it on the 3 problem instances mentioned above. Execute 30 independent runs your algorithm for each problem instance, and report the results obtained.
   For each instance report the best solution quality obtained, the number of objective function evaluations needed to reach that solution, and the CPU time taken to do so.

2. Repeat the previous question using *Multistart Next Ascent Hillclimbing* (MSNAHC) instead of *Next Ascent Hillclimbing*. Again, perform 30 independent runs of your algorithm for each problem instance. Each execution of MSNAHC should stop if either a global optimum is found or if by the time a given restart begins a maximum of 10 million objective function evaluations have already elapsed, whichever occurs first.
   For each instance report the best solution quality obtained, the number of objective function evaluations needed to reach that solution, and the CPU time taken to do so.

3. Implement *Variable Neighbourhood Ascent* using upto a 3-bit *Hamming Distance* neighbourhood, and test it on the 3 problem instances mentioned above. Again, perform 30 independent runs of your algorithm for each problem instance, and report the results obtained. For each instance report the best solution quality obtained, the number of objective function evaluations needed to reach that solution, and the CPU time taken to do so.

4. Repeat the previous question using a Multistart version. That is, implement *Multistart Variable Neighbourhood Ascent* (MSVNA). Again, perform 30 independent runs of your algorithm for each problem instance. Each execution of MSVNA should stop if either a global optimum is found or if by the time a given restart begins a maximum of 10 million objective function evaluations have already elapsed, whichever occurs first.
   For each instance report the best solution quality obtained, the number of objective function evaluations needed to reach that solution, and the CPU time taken to do so.

5. Write a couple of paragraphs discussing the results that you obtained in this assignment, and present whatever concluding remarks you may have.