TU Wien
Institute of Logic and Computation
Algorithms and Complexity Group

Heuristic Optimization Techniques, WS 2023

Programming Exercise: Assignment 2

v1, 16-8-2023

Work on the problem specified in the separate problem description and general information document within your programming exercise group. You are encouraged to split tasks evenly but every group member has to understand and to be able to explain all the concepts involved, your overall implementation, and the submitted report. Hand in your complete report, submit the solutions for the instances, and upload your source code via TUWEL by **Friday, 12th January 2024, 23:55**. For further questions please send an e-mail to `heuopt@ac.tuwien.ac.at`

Competition: The three student teams that will have uploaded the best solutions to a selection of *three competition instances* in TUWEL by the aforementioned deadline will receive a small prize! Submissions are evaluated according to the mean objective value over the competition instances. Upload date/time serves as tie breaker. Your rank in the competition has no influence on the final mark you receive for the course.

The second programming assignment is to **develop more advanced metaheuristics** for the *Weighted s-Plex Editing Problem* (WsPEP) in your preferred programming language, and to perform a **more advanced parameter tuning and statistical comparison**. You can build upon everything you developed for the first programming exercise.

The tasks for this exercise are:

1. Design and implement **two** of the following approaches to solve the WsPEP.
   - Evolutionary/Genetic Algorithm including a suitable heuristic to estimate the cost of an s-plex.
   - Ant Colony Optimization including a suitable heuristic to estimate the cost of an s-plex.
   - (Adaptive) Large Neighborhood Search
   - Some other hybrid algorithm: You can (and are encouraged to) build upon the algorithms you implemented so far, but the algorithm has to involve something substantially new.

On TUWEL you find tuning and test instance sets containing 20 separate instances each (for which there is no solution uploading) which you should use for the following two subtasks.

2. Perform a more advanced automated parameter tuning of the parameters of each of your two algorithms of Task 1 with the aim to find as good as possible solutions within limited time. Use the 20 **tuning** instances to do so, leaving the test set untouched. Get inspired by the possibilities stated in the slides regarding the tuning of metaheuristics. Recall that you have access to the AC group's cluster in case you want to perform computationally more demanding tuning, see the general information PDF. Report the found parameters and the procedure to obtain them.
3. After having finished Task 2, test whether there is a significant difference between the two tuned algorithms concerning the quality of the best found solutions on the **test** instances. You may use the statistical testing Jupyter notebook from the supplementary material.
4. Run experiments on the mentioned instances provided in TUWEL using one of the two tuned solution approaches that turned out to be better on the test instances. If the test results were inconclusive, choose one which you expect to be more promising and argue why.
5. Write a concise report containing the description of your solution approaches, the applied parameter tuning method with corresponding results, and the final experimental results. Include also some plots where the achieved solution quality over time, iterations, or generations on selected instances is shown.

We hope you enjoy these tasks and wish you success!

At the end: Reflect on your work. Which experiences do you take away from the programming assignments?