ELEC-E7852 Computational Interaction and Design 2024
Antti Oulasvirta
Aalto University

Assignment 1a: Combinatorial optimization (Mandatory)

Preparation: Decide your *topic* and *case*

Decide your topic. We first choose a *topic* based on the second letter of your last name:

1. A-K: Assignment of colors to UI elements
2. L-T: Assignment of properties to UI elements
3. U-Z: Assignment of UI elements to positions in a UI layout
4. Å-Ö: Wild card! Pick what you want!

For example, Oulasvirta would work on topic number 3. The three cases pose similar challenges -- they are all assignment problems. However they call for different objective functions.

Specify your case. Then, ideate a *case* for your topic. The case should be a small version of an arguably *realistic* problem in computational design. For example, Oulasvirta chooses to work on a case where: 1) a menu layout and a list of command names is given and 2) it must be decided which command is *assigned* to which slot in the menu. He defines his objective function to be two-fold: 1) first, maximize the average speed with which commands can be selected, assuming that commands that are closer to the top are faster to select; and 2) to minimize learning time, assuming that it is easier to learn the menu when items that are associated (e.g., Save, Open) are close to each other.

Tips: We **strongly** recommend to go through the materials in MyCourses prior to trying this assignments; the notebook, slides and the optional reading have lots of ideas. After familiarizing with the materials, ideate a few case ideas. Finally, pick the winner by assessing what you might be able to implement.

Task: Create an optimizer for your case

Your task is to write a combinatorial optimizer in Python that 1) takes as input a *task instance* of your case, 2) searches for an optimal solution for 2 minutes on your computer, and 3) outputs the best design found.

Reporting format:

1. Name, student ID, and assignment ID (A1a)

2. Introduction: verbal characterization and a visual illustration of the case
3. Formulation of the assignment problem
4. Approach to the solver: what and why
5. Screenshot of key pieces of code
6. Results: quantitative (improvement in objective score over time) and qualitative (example outputs for 5 different task instances)
7. Conclusion: analysis of what works and what does not

Grading:

- A meaningful case chosen and described +1
- The objective function is user-centric +1
- The formulation of the assignment problem matches the case +1
- The algorithm demonstrably achieves meaningful results in toy problems +1
- Clear and illustrative documentation of: case, approach, results, conclusion +1