Algorithmic Methods for Mathematical Models (AMMM)

Lab Session 4 – Greedy + Local Search Heuristics

In this fourth session we will focus on using heuristics to solve the problem described in the last lab session regarding assigning tasks to computers in a data center.

1. Problem statement

Recall that the *P3* problem was formally stated as follows:

Given:

- The set T of tasks. Each task t consists of a set of threads $H(t)$. For each thread h the amount of requested resources r_h is specified.
- The set C of computers. Each computer c consists of a set of cores $K(c)$. All cores of each computer c have the same capacity r_c.

Find the assignment of tasks to computers and threads to cores subject to the following constraints:

- Each thread is assigned to a single core.
- Each task is assigned to a single computer, i.e. all the threads of a task are assigned to cores of the same computer.
- The capacity of each core cannot be exceeded.

with the *objective* to minimize the highest loaded computer.

2. Tasks

In pairs, do the following tasks and prepare a lab report using the Python code that is provided.

a) Prepare a pseudocode for the Greedy algorithm. Specify the greedy function.

b) Prepare a pseudocode for the Local search algorithm. What is the difference between best improvement (BI) and first improvement (FI)? Which neighborhoods are implemented?

c) Generate instances of increasing size. Store these instances as they will be used in the coming lab sessions.

d) Solve the instances previously generated using:

- Greedy function only,
- Greedy + Local search with N_1 and BI
- Greedy + Local search with N_1 and FI
- Greedy + Local search with N_2 and BI

- Greedy + Local search with N_2 and FI

Plot the quality of the solutions and time to solve. Select the best combination.

e) Solve the instances previously generated using the ILP from lab session 3. Configure CPLEX to stop after 30min or GAP≤1%.

f) Plot the best combination for the Greedy and the ILP in terms of quality of the solutions and time to solve.