Coursework for Heuristics 2023-2024

This coursework is motivated by a real-world problem, arising in an industrial bakery in Iran.

The Iranian bakery produces, among others, a specialty biscuit that cooks for many hours at slow temperature. The biscuits have different shapes. Therefore, the cooking duration of biscuits might vary. Biscuits are prepared and baked in trays, and trays with the same cooking duration are called hereafter a job. Each job t is associated with a cooking duration c_t and a space occupied in the oven s_t . The trays in a job have to be cooked in the same oven. Each oven o has a capacity of S_o, so multiple jobs can be cooked simultaneously as long as the capacity of the oven is satisfied (the jobs cooking simultaneously in the same oven are called a batch) Note that an oven can cook more than a batch of biscuits in a day, but a new batch can start cooking only when all jobs in the previous batch are finished cooking. Your goal is to schedule the baking of all jobs, so that time to complete the baking of the last job is minimized. The literature refers to this objective as the *makespan minimization*.

To solve this problem, you are expected to:

1) Develop a constructive heuristic (different to the one explained in class) [35 marks].
2) Develop two local search neighbourhoods (in addition to the one which will be provided in class) [35 marks].
3) Develop a metaheuristic among random restart, tabu search, iterated local search and simulated annealing (different to the one explained in class) [30 marks].

For each of the components of the assignment, please:

1) Describe the algorithm chosen and your implementation.
2) Provide a pseudocode for the constructive heuristic.
3) Present (when possible) the computational complexity of the constructive algorithm.
4) Discuss pros and cons of your design choices.
5) Execute your algorithms on the set of instances provided.
6) Present your results in a table and, using appropriate data visualisation techniques; discuss and analyse your results.
7) Highlight limitations of the problem definition and possible extensions to this project.

Instances in text format will be provided (on Moodle) containing:

a) The number of ovens
b) The number of jobs
c) The ovens' capacities
d) The space occupied by each job in an oven
e) The time necessary to complete a job in the oven.

You are given a VBA macro that can read the instances and fill suitable data structures, as well as a function that computes if a given solution is feasible and its objective function value. You are expected to use these data structures within the coursework, however, you can include additional data structures as you deem necessary.

The problem definition will be discussed in class, when you will be allowed to ask questions about the initial code provided.

Your coursework will be marked based on the algorithmic performance (both in terms of solution quality and speed of computation), originality of the algorithms developed, quality and correctness of the report, correctness, and precision of the analysis of complexity, quality of the pseudocodes, evaluation of the pros/cons of the algorithms, as well as the analysis of the results achieved.

Please submit on Moodle by 14:00 of the 21th of March 2024:

- A Microsoft Word file describing your work and your results. The maximum word count is 3000 words, but tables, figures and pseudocodes are not included in the word count. Note that there is no 10% allowance. Name your file "GroupXX2021.docx", with XX corresponding to your group number.

- The Excel spreadsheet with your codes and results. The code should successfully run and the lecturer should be able to replicate your final results. Name your file "GroupXX2021.xlsm"

- The coursework submission sheet, as instructed by the MSc office for all group coursework submissions. Name this file "CSSGroupXX2021.pdf"

You will be communicated your group number shortly on the Moodle.

Example:

- On the left: a problem instance (you can refer to the instance “Debug.txt”),
- On the right: the representation of the instance (in which each oven and job are represented based on the space occupied- x-axis, and time necessary to complete the job- y-axis), and a feasible solution (below).

NumJobs

7

OvenCapacity

3

5

4

SpaceOccupiedPerJob

2

1

3

4

5

2

1

TimeNecessaryforEachJob

2

4

5

3

4

6

5

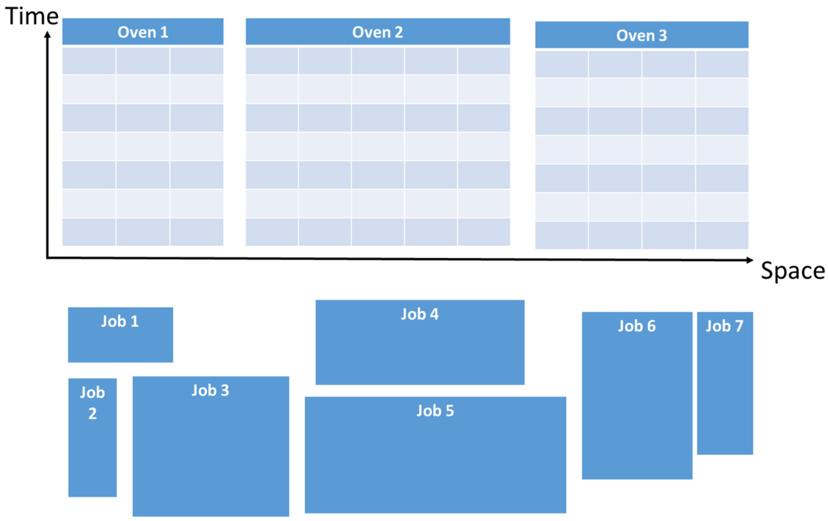

Figure 1 Representation of the instance, on a time/space network

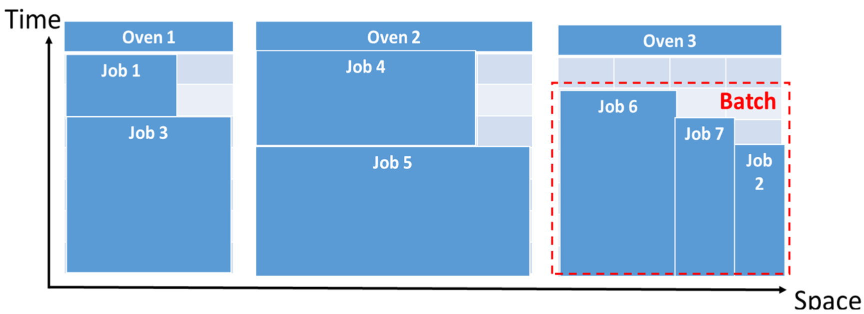

Figure 2 Depiction of a feasible solution with makespan7

Example of solution representation, based on the data structures used in the code (all other entries would be -1)

sol.assignments(1, 1, 1) = 3 ’ in oven 1, for batch number 1, the first job is 3

sol.assignments(1, 2, 1) = 1 ‘ in oven 1, for batch number 2, the first job is 1

sol.assignments(2, 1, 1) = 5 ‘ in oven 2, for batch number 1, the first job is 5

sol.assignments(2, 2, 1) = 4 ‘ in oven 2, for batch number 2, the first job is 4

sol.assignments(3, 1, 1) = 6 ‘ in oven 3, for batch number 1, the first job is 6

sol.assignments(3, 1, 2) = 7 ‘ in oven 3, for batch number 1, the second job is 7

sol.assignments(3, 1, 3) = 2 ‘ in oven 3, for batch number 1, the third job is 2