ALGORITHMICS - Bachelor of Software Engineering, 2024-2025

LAB GUIDE. SESSION 4

GOALS:

- **Greedy algorithms: Map Colouring**

1. Introduction

Map colouring is one of the application cases of the greedy algorithm. Given a map with n regions, the objective is to assign each region a different colour from all adjacent regions.

In this case, the map will be represented by a graph built on a grid, in which each node represents a region of the map and the connections between nodes represent boundaries between regions. A node can have up to 8 connections to other nodes, symbolising cardinal and ordinal directions.

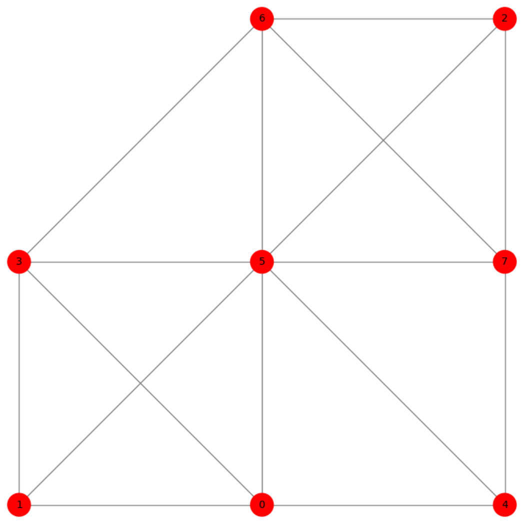

Figure 1 -Visualization of graph g8.json with function draw_coloured_map

2. Map Colouring

You are provided with the module **helper.py**, which has two functions:

- *Generate_graph_map* constructs a graph having the described characteristics with the indicated number of nodes. It returns a JSON with two values: '*nodes*' contains the positions of each node in the grid (exclusively to build the visualisation) and '*graph*' contains, for each node, a list of its neighbouring nodes. Also, this JSON is stored in a file '*grafo.json*'.
- *Draw_coloured_map* receives as parameters a graph in JSON format generated by the previous function and a JSON '*colours'* that must contain, for each node, the colour that corresponds to it.

For proper operation, run the command **pip install -r requirements.txt** in the project directory, which will install the necessary libraries.

This module additionally includes an example of use in which a 128-node graph and a visualization with homogeneous colours are generated.

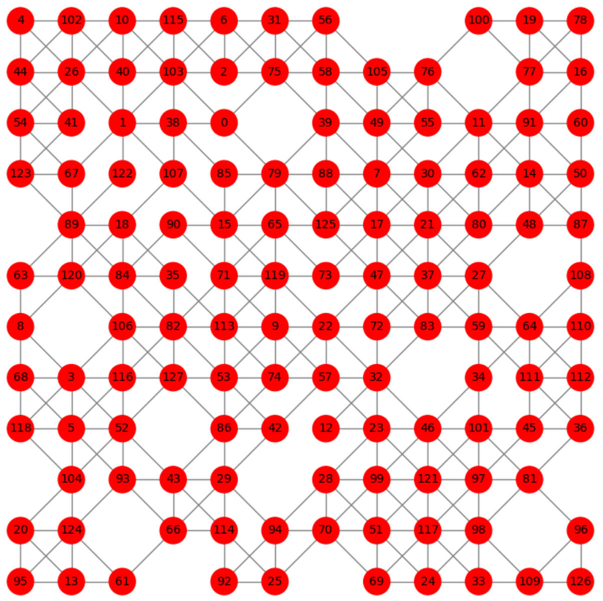

Figure 2 – Visualization of graph g128.json with homogeneous colours

The aim of this practice is to solve, through the implementation of a greedy algorithm, the colouring of these graphs according to the described restrictions. For example, for the previous one, a possible solution -with 6 colours used- would be:

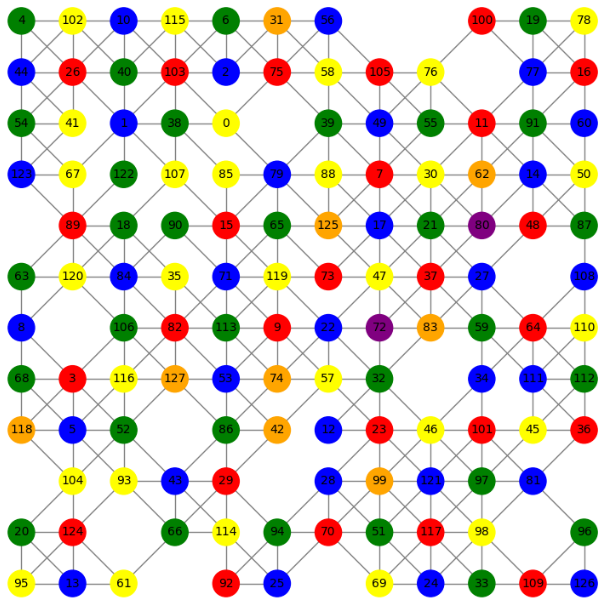

Figure 3 – Visualization of graph g128.json with the colours of solution s128.json

In the *sols* directory, possible solutions are provided as a reference, calculated with a certain heuristic, for graphs of varying size. In the case that there are several optimal solutions with the same number of colours used, any of them is valid, so there could be other valid solutions to those proposed, but obviously they should use the same number of colours, or less, if a better heuristic is defined.

A proposal of the list of colours to use is the following: *["red", "blue", "green", "yellow", "orange", "purple", "cyan", "magenta", "lime"].*

YOU ARE REQUESTED TO:

Depending on the instructions of your professor:

Implement a class **GraphColouring.java**, so that with the provided class **Greedy.java** we can calculate a solution and visualize it with the provided Python module. You must add the provided JAR to the build path for the latter to work.

Explain the time complexity of the implemented algorithm.

Implement a class **GreedyTimes.java**, using the graphs contained in **sols** and calculating the time taken by the algorithm made in the previous section to solve the problem, so that the following table can be filled in.

Does the previously calculated complexity follow the times in the table?

Implement the module **graph_colouring.py**, so that we can calculate a solution and visualize it with the provided Python module.

Explain the time complexity of the implemented algorithm.

Implement a module **greedy_times.py**, using the graphs contained in **sols** and calculating the time it takes for the algorithm done in the previous section to solve the problem, so that the following table can be filled in.

Does the previously calculated complexity follow the times in the table?

TABLE FOR GRAPH COLOURING TIMES

n	***t Colouring (ms)***
8	
16	
32	
…	
4096	
8192	

16384	
32768	
65536	

3. Work to be done

- An `algstudent.s4` **package** in your course project. The content of the package should be the files used and created during this session.
- A `session4.pdf` **document** using the course template (the document should be included in the same package as the code files). You should create one activity each time you find a "YOU ARE REQUESTED TO" instruction.

Deadline: The delivery of this lab will be carried out, in time and form, according to the instructions given by the lab teacher.