Modelling with Graphs Summative

February 27, 2020

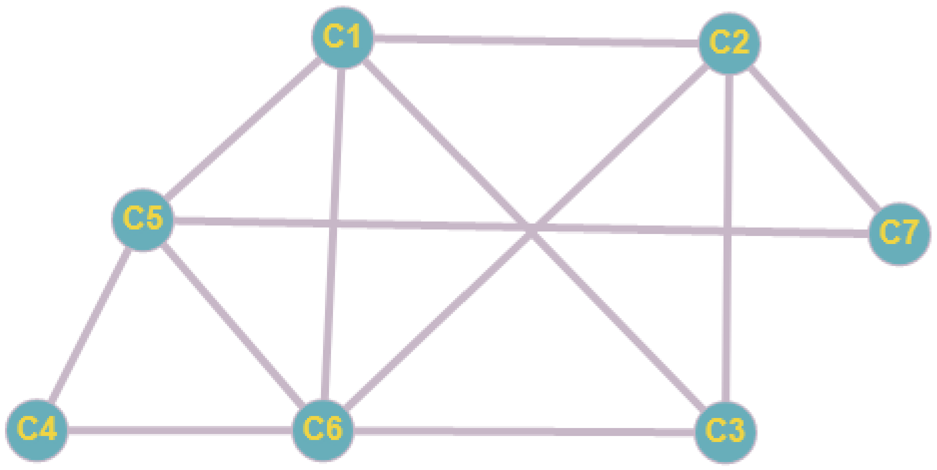
A.3

A.3.1

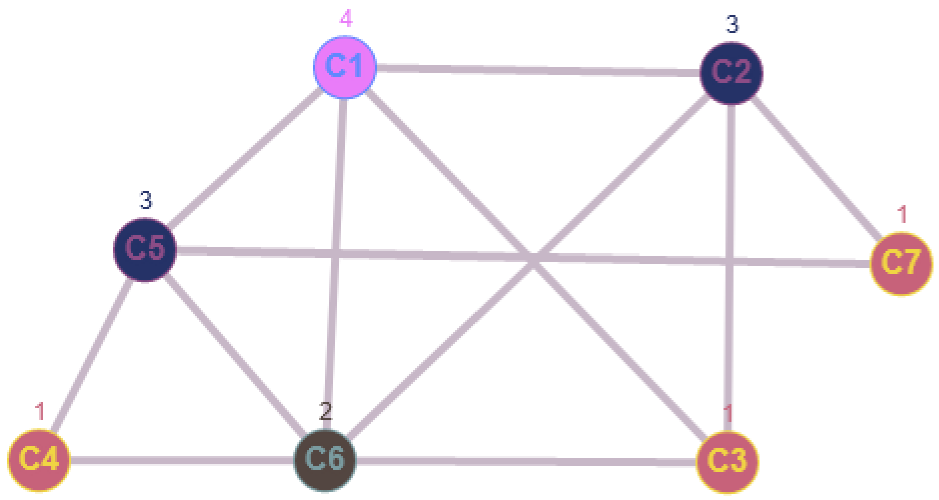
To transform problem A.3 into a graph colouring problem , the problem is represented as a graph where:
Nodes = individual classes ($C_1, ..., C_7$)
Edges = classes with at least one shared student (eg. "Amy" in C_1 and C_5)
Colour = A timeslot
A representation of the graph as an adjacency matrix is:

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

An example graph would be:



The solution to the problem is the answer to the question, can the graph be coloured in at most 4 colours?
This will solve the problem because if the graph can be coloured so that no neighbouring nodes have the same timeslot then the student can physially attend every class, and by doing it in at most 4 colours means there are sufficient time slots.
An example colouring of the above graph would be:



A.3.2

As there are currently no visited nodes, the algorithm will first visit node C_1.

Visited - C_1
Adjacent - C_2, C_3, C_5, C_6

Visit C_2

Visited - C_1, C_2
Adjacent - C_3, C_5, C_6, C_7

Visit C_3

Visited - C_1, C_2, C_3
Adjacent - C_5, C_6, C_7

Visit C_6

Visited - C_1, C_2, C_3, C_6
Adjacent - C_5, C_7

Visit C_4

Visited - C_1, C_2, C_3, C_6, C_4
Adjacent - C_5, C_7

Visit C_5

Visited - $C_1, C_2, C_3, C_6, C_4, C_5$
Adjacent - C_7

Visit C_7

Visited - $C_1, C_2, C_3, C_6, C_4, C_5, C_7$

Order in which the algorithm visits the vertices to find a proper colouring is:

$$C_1, C_2, C_3, C_6, C_4, C_5, C_7$$

A.3.3

The colours the algorithm assigns to each vertex are below:

Vertex	Colour
C_1	1
C_2	2
C_3	3
C_4	1
C_5	2
C_6	4
C_7	1

A.3.4

The chromatic number of G, χG, is 4 because it the minimum number of colours needed to properly colour G is 4.

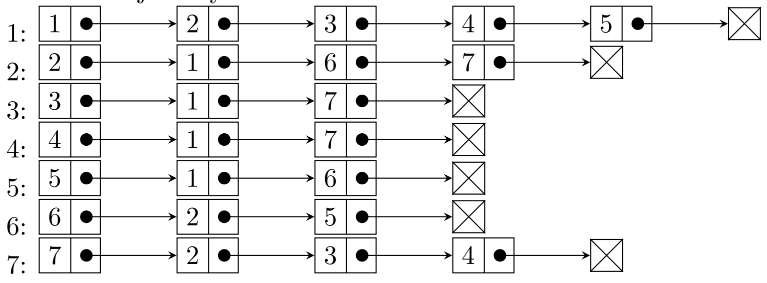
B.2

B.2.1

The adjacency matrix for the graph is below:

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

And the adjacency linked list is below:


1: 1 → 2 → 3 → 4 → 5
2: 2 → 1 → 6 → 7
3: 3 → 1 → 7
4: 4 → 1 → 7
5: 5 → 1 → 6
6: 6 → 2 → 5
7: 7 → 2 → 3 → 4