ELL781: Minor Test IIb

September 28, 2019

Maximum Marks: 6

1. Consider an undirected graph with 5 vertices and 9 edges, similar to the kind of graph we constructed to represent conflicting turns at a traffic intersection (there are no self-loops). Answer the following questions, with proof/justification:

   (a) How many vertices does the largest clique in the graph contain? **[1.5]**

   (b) Based on just the answer to part (a), what can you say about the minimum number of colours needed to colour this graph? Give a range of possible values for this. **[0.5]**

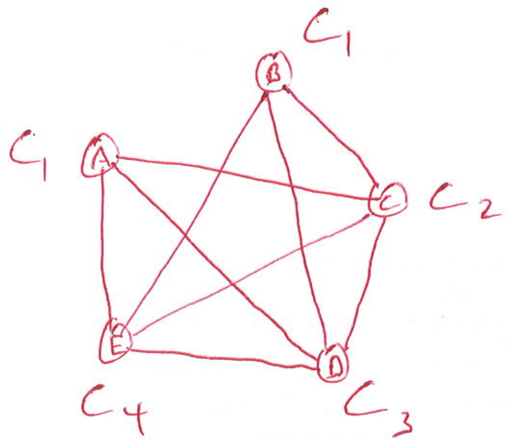
   (c) Can you apply some further reasoning (beyond just using the answer to part (a)) to give a definite value for the minimum number of colours needed? Explain/illustrate your reasoning clearly. If you can come up with a definite value, draw an example graph with the above properties and show a colouring using that many colours. **[2]**

   (d) Will the greedy graph colouring algorithm discussed in class always give the optimal solution in this situation? Clearly explain your reasoning as to why or why not. **[2]**

(a) ${}^5C_2 = 10$, so only one possible edge is missing. Let it be between vertices A and B. Then any subset with 4 vertices, which doesn't include both A and B, will form a clique. So the graph has 2 4-cliques (one excluding A, one excluding B) and the answer is $\underline{4}$.

(b) 4 is a lower bound on the min. no. of colours needed. So it must be 4 or 5.

(c) The nodes A and B (as per part (a)) can certainly be given the same colour, because there is no edge between them, and every other node will have a different colour. So A and B having the same colour will not cause any conflicts.



$C_1, \ldots, C_4$ denote the colours

Hence the min. no. of colours is definitely <u>4</u>.

(d) The only way the greedy algorithm can fail to be optimal is if it fails to assign the same colour to A and B (as per above notation). Suppose, without loss of generality, that A is coloured first, in a particular iteration of the algorithm. Now, none of the nodes other than B can get this same colour, as all are connected to A. So the iteration will come to B; and since only A has this colour, and B is not connected to A, B will also get the same colour. Hence the greedy algorithm will always be optimal for this kind of graph.