Solutions for the exam on January 5th 2024

Vilhelm Agdur[1]

[1] vilhelm.agdur@math.uu.se

5 January 2024

Questions one through eight were all of the form "reproduce this result from the lecture notes", so we give no solution for those. For questions nine and ten we give a sketch of a solution.

Question 9

The right "intermediate concept" to consider here is that of a *topological ordering* of a directed acyclic graph. A topological ordering is an ordering of the vertices of the graph such that whenever $u \to v$ is an edge of the graph, u precedes v in the ordering.

It should be clear that our DAG will be Hamiltonian if and only if a topological ordering of its vertices is a path in the graph. So we can divide the problem into two parts – first we find a topological ordering, and then we check if this ordering is in fact a path. The latter step can clearly be done in time $O(|V|)$.

How does one find such an ordering? There are multiple algorithms for it,[2] so let us give just one, which is called *Kahn's algorithm*.

[2] Wikipedia suggests three different options: https://en.wikipedia.org/wiki/Topological_sorting

First, loop through all vertices and find the set of vertices which have no incoming edges – at least one such vertex must exist in any DAG.[3] Initialize a set of these vertices as S, and let L be an empty list. Then, as long as S is not empty, we proceed as follows:

[3] Keeping our end-goal of determining hamiltonicity in mind, we can also enumerate the vertices with no outgoing edges. If the graph is Hamiltonian, it must of course have exactly one of each – so if we find too many of either kind, we can terminate the algorithm already at this step.

1. Remove a vertex v from S, and add it to L.
2. For each vertex w such that $v \to w$ is an edge, remove that edge from the graph. If w had no other incoming edges, add w to S.

If the graph we started with had no cycles, this algorithm will eventually terminate with no edges of the graph remaining, and L will be a topological ordering of the graph.

It is clear that this algorithm also has time complexity linear in the number of vertices and edges, and so we have found a linear time algorithm for determining if a DAG is Hamiltonian.

Question 10

a) This is true. The easiest example is just the complete graph on $k+1$ vertices.

In fact, the stronger statement that for every k, there is a k-regular Hamiltonian graph on n vertices for every sufficiently large n is also true. One construction of an example of such a graph might go as follows:

Pick a large n, and create a graph G by starting with a cycle on n vertices. Then, pick $k-2$ matchings on G^c, and add those edges to G to make it Hamiltonian.[4] That this is graph is Hamiltonian is obvious, since the cycle the construction started with is a Hamiltonian cycle.

[4] Why is this possible? If you picked n large enough, Dirac's theorem tells you there is a Hamiltonian cycle in G^c, so picking every second edge of this gets you your first matching. Then, removing the edges of this matching from G^c doesn't reduce the degree of any vertex by too much (since you picked n very large, $n-3-k > n/2$), so there is still a Hamiltonian cycle by Dirac's theorem, and repeat until you have all k matchings.

b) This is false. Recall that we proved in the lecture notes that

$$|E| \leq 3|V| - 6$$

as a corollary to Euler's formula, by using a double counting argument counting edges incident to faces. Then, in our proof of the five-colour theorem, we used this to prove that the minimum degree of a planar graph can be at most five – that is, every planar graph has a vertex whose degree is at most five. So there are no k-regular planar graphs for $k > 5$.

c) This is true. Consider the graph with two vertices and k edges between those two vertices – clearly the spanning trees of this graph are gotten precisely by picking one of the edges.

d) This is false – but it fails only for $k = 2$! For $k = 1$, it is obviously true – any graph which is itself a tree has only one spanning tree. For $k > 2$, consider the cycle graph on k vertices: A spanning tree of this graph is gotten by removing one of the edges of the graph to render it acyclic.

To see that it fails for $k = 2$, let G be any graph with more than one spanning tree. That it has more than one spanning tree implies it is not a tree, so it contains a cycle, say C. Now pick a spanning forest F of $G \setminus C$.[5] For any edge e of C, we get a spanning tree of G by taking $F \cup (C \setminus e)$ – and any cycle of course has more than two edges, so this procedure gives us more than two spanning trees, and so G does not have exactly two spanning trees.

[5] $G \setminus C$ may of course be disconnected, so we need to say spanning *forest* here.

e) This is false. Suppose G is any planar graph. As we saw in a previous part of this question, G has a vertex v such that $d_v \leq 5$. So if we remove the set of neighbours $N(v)$ of v from G, this will render it disconnected. So we have removed a set of five vertices and thereby disconnected the graph, and thus it cannot be six-connected.

f) Whether this is true or false depends on if you interpret "tree" to mean "finite tree" or "any tree". It is false for finite trees, since all finite trees have leaves.[6] However, there definitely are k-regular

[6] We gave a proof of this fact in one of the earliest lectures.

trees for every k if you allow infinite trees – to get a two-regular tree, just turn $\mathbb{Z}$ into a graph by adding an edge between i and $i+1$ for every i.