CS 70 Discrete Mathematics and Probability Theory HW 3
Spring 2017 Rao

1 Sundry

Before you start your homework, write down your team. Who else did you work with on this homework? List names and email addresses. (In case of homework party, you can also just describe the group.) How did you work on this homework? Working in groups of 3-5 will earn credit for your "Sundry" grade.

Please copy the following statement and sign next to it:

I certify that all solutions are entirely in my words and that I have not looked at another student's solutions. I have credited all external sources in this write up.

2 Leaves in a Tree

A *leaf* in a tree is a vertex with degree 1.

(a) Prove that every tree on $n \geq 2$ vertices has at least two leaves.

(b) What is the maximum number of leaves in a tree with $n \geq 3$ vertices?

3 Build-Up Error?

What is wrong with the following "proof"?

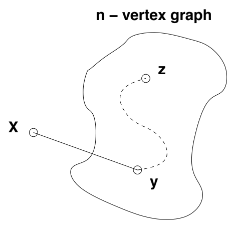
False Claim: If every vertex in an undirected graph has degree at least 1, then the graph is connected.

Proof: We use induction on the number of vertices $n \geq 1$.

Base case: There is only one graph with a single vertex and it has degree 0. Therefore, the base case is vacuously true, since the if-part is false.

Inductive hypothesis: Assume the claim is true for some $n \geq 1$.

Inductive step: We prove the claim is also true for $n+1$. Consider an undirected graph on n vertices in which every vertex has degree at least 1. By the inductive hypothesis, this graph is connected. Now add one more vertex x to obtain a graph on $(n+1)$ vertices, as shown below.

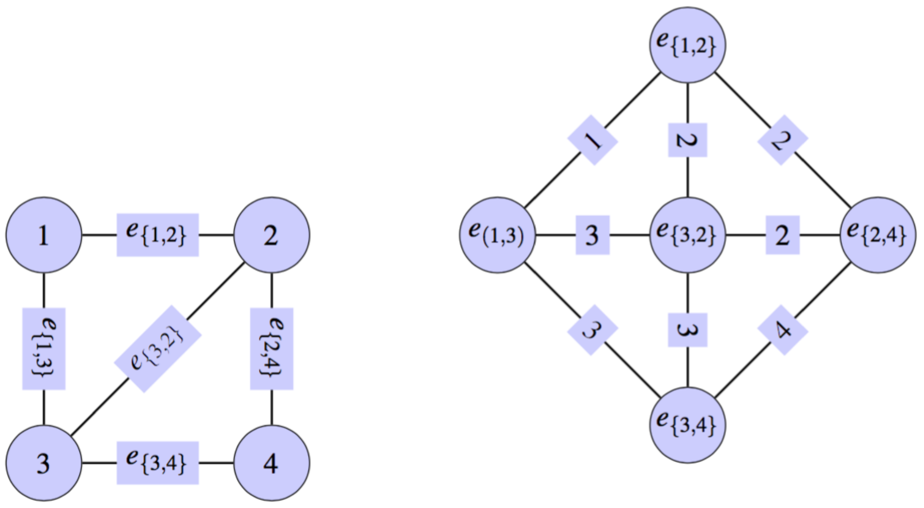


All that remains is to check that there is a path from x to every other vertex z. Since x has degree at least 1, there is an edge from x to some other vertex; call it y. Thus, we can obtain a path from x to z by adjoining the edge $\{x,y\}$ to the path from y to z. This proves the claim for $n+1$.

4 Graph Coloring

Prove that a graph with maximum degree at most k is $(k+1)$-colorable.

5 Edge Complement



The **edge complement** graph of a graph $G = (V,E)$ is a graph $G' = (V',E')$, such that $V' = E$, and $(i,j) \in E'$ if and only if i and j had a common vertex in G. In the above picture, the graph on the right is the edge complement of the graph on the left: for every edge $e_{\{i,j\}}$ in the graph on the left there is a vertex in the graph on the right. If two edges $e_{\{i,j\}}$ and $e_{\{j,k\}}$ share a vertex j on the left, then the corresponding vertices on the right have an edge j connecting them.

(a) Prove or disprove: if a graph G has an Eulerian tour, then its **edge complement** graph has an Eulerian tour.

(b) Prove or disprove: if a graph's **edge complement** graph G' has an Eulerian tour, then graph G has an Eulerian tour.

6 Proofs in Graphs

Please prove or disprove the following claims.

(a) Suppose we have n websites ($n \geq 2$) such that for every pair of websites A and B, either A has a link to B or B has a link to A. Prove or disprove that there exists a website that is reachable from every other website by clicking at most 2 links. (*Hint: Induction*)

(b) In the lecture, we have shown that a connected undirected graph has an Eulerian tour if and only if every vertex has even degree.

Prove or disprove that if a connected graph G on n vertices has exactly $2d$ vertices of odd degree, then there are d walks ($d > 0$) that *together* cover all the edges of G (i.e., each edge of G occurs in exactly one of the d walks; and each of the walks should not contain any particular edge more than once).

7 Triangulated Planar Graph

In this problem you will prove that every triangulated planar graph (every face has 3 sides; that is, every face has three edges bordering it, including the unbounded face) contains either (1) a vertex of degree 1, 2, 3, 4, (2) two degree 5 vertices which are connected together, or (3) a degree 5 and a degree 6 vertices which are connected together. Justify your answers.

(a) Place a charge on each vertex v of value $6 - \text{degree}(v)$. What is the sum of the charges on all the vertices? (*Hint*: Use Euler's formula and the fact that the planar graph is triangulated.)

(b) What is the charge of a degree 5 vertex and of a degree 6 vertex?

(c) Move $1/5$ charge from each degree 5 vertex to each of its negatively charged neighbors. Conclude the proof in the case where there is a degree 5 vertex with positive remaining charge.

(d) If no degree 5 vertices have positive charge after discharging, does there exist a vertex with positive charge after discharging? If there is such a vertex, what are possible degrees of that vertex?

(e) Suppose there exists a degree 7 vertex with positive charge after the discharging process of degree 5 vertices. How many neighbors of degree 5 might it have?

(f) Continuing the last question. Since the graph is triangulated, are two of these degree 5 vertices adjacent?

(g) Finish the proof from the facts you obtained from the previous questions.

8 Hypercube Routing

Recall that an n-dimensional hypercube contains 2^n vertices, each labeled with a distinct n bit string, and two vertices are adjacent if and only if their bit strings differ in exactly one position.

(a) The hypercube is a popular architecture for parallel computation. Let each vertex of the hypercube represent a processor and each edge represent a communication link. Suppose we want to send a packet from vertex x to vertex y. Consider the following "bit-fixing" algorithm:

> In each step, the current processor compares its address to the destination address of the packet. Let's say that the two addresses match up to the first k positions (reading the bits from left to right). The processor then forwards the packet and the destination address on to its neighboring processor whose address matches the destination address in at least the first $k+1$ positions. This process continues until the packet arrives at its destination.

Consider the following example where $n = 4$: Suppose that the source vertex is (1001) and the destination vertex is (0100). Give the sequence of processors that the packet is forwarded to using the bit-fixing algorithm.

(b) The *Hamming distance* $H(x,y)$ between two n-bit strings x and y is the number of bit positions where they differ. Show that for an arbitrary source vertex and arbitrary destination vertex, the number of edges that the packet must traverse under this algorithm is the Hamming distance between the n-bit strings labeling source and destination vertices.

(c) Consider the following example where $n = 3$: Suppose that x is (110) and y is (011). What is the length of the shortest path between x and y? What is the set of all vertices and the set of all edges that lie on shortest paths between x and y? Do you see a pattern? You do not need to prove your answer here – you'll provide a general proof in part (d).

(d) Answer the last question for an arbitrary pair of vertices x and y in the hypercube. Can you describe the set of vertices and the set of edges that lie on shortest paths between x and y? Prove that your answers are correct. (*Hint:* Consider the bits where x and y differ.)

(e) There is another famous graph, called the butterfly network, which is another popular architecture for parallel computation. You will see this network in CS 170 in the context of circuits for implementing the FFT (fast fourier transform). Here is a diagram of the butterfly network for $n = 3$. In general, the butterfly network has $(n+1)\cdot 2^n$ vertices organized into $n+1$ columns of 2^n vertices each. The vertices in each column are labeled with the bit strings in $\{0,1\}^n$, and all vertices in the same row have the same label. The source is on the leftmost column and the destination is on the right.

It turns out the n-butterfly network is equivalent to the n-dimensional hypercube unrolled into n bit-fixing steps. On the graph below, trace all the paths from source $x = (110)$ to destination $y = (011)$, so that these paths are the shortest bit-fixing paths that you obtained from part (c). For this, you need to label the vertices in the graph explicitly. This example should convince you that the butterfly network is indeed equivalent to the hypercube routing.