Ma6 PSET 4

Victoria Liu

November 10, 2020

Problem 1

Proof. I think the statement "any graph has a bipartite subgraph containing at least half its edges" won't hold for every non-simple graph. For example, consider a single vertex with a single self-loop. Since bipartite graphs cannot have self-loops, we will never be able to find a bipartite subgraph with more than 0 edges. In my answer, I will only prove the statement for graphs without self-loops; parallel edges are fine.

Let's prove the statement using induction on the number of vertices. Since we are looking at bipartite graphs, let's start with a base case of 2 vertices; graphs with zero vertices or one vertex are "trivially" bipartite already. When we have two vertices, we can partition each vertex into its own vertex set, thus preserving at least half its edges (actually all of its edges) this way.

Now, we assume the statement is true for all graphs with k vertices, and we want to show that the statement is still true for graphs with $k+1$ vertices. Let's say we have any graph G with $k+1$ vertices. Remove any vertex a and the set of edges with an endpoint in a. Call this set of removed edges E_a and the resulting graph after these removals H. Graph H has k vertices, so by our induction hypothesis, it contains a bipartite subgraph with at least half the edges in H. In other words, H has a partition of two vertex sets U and V such that the number of edges connecting elements of U to elements of V is at least half of the total number of edges in H; let's call the total number of edges in H $|E_H|$. Now, we go back to graph G. We simply place a into the vertex set (U or V) where at least half of E_a remains in the bipartite subgraph. For example, if exactly half of E_a remains when a is placed in U, then we could place a in either U or V. In another example, if exactly a third of E_a remains when a is placed in U, then exactly two-thirds of E_a remains when a is placed in V, and we should place a in V. We can think that the choice of placing a in either U or V will "partition" E_a into two disjoint edge sets, since every edge in E_a will either connect a to a vertex in U or to a vertex in V; if a is placed in U, we take away the edges in E_a connecting a to vertices in U, and vice versa. Of these two disjoint edge sets, at least one set will contain at least half of $|E_a|$.

Now let's count the number of edges we have in this bipartite subgraph of G–we have all the edges from H's bipartite subraph, and this count is at least $\frac{1}{2}|E_H|$. In addition to these edges, we also have at least half the edges in E_a, or $\frac{1}{2}|E_a|$. Thus, we have at least $\frac{1}{2}\left(|E_H| + |E_a|\right)$. Note that we defined E_H and E_a by $E_H + E_a = E_G$, so we are done. We have a bipartite subgraph of G with at least half its edges.

□

Problem 2

2.a

Proof. We can show this using induction on the number of edges $|E|$ in the connected graph. For our base case: when we have 0 edges, we have exactly 1 vertex (if we had 0 vertices, there wouldn't be a graph) and 1 face, the outer infinite face. The one-vertex graph is clearly planar, since we don't have any edges to cross. We also have $1 - 0 + 1 = 2$, satisfying the formula. Now, assume that this formula holds for any planar graph with k edges, so that $v - e + f = 2$ holds.

Now, let's examine a graph G with $k + 1$ edges. We assume that G is planar and want to show that $v_G - (k + 1) + f_G = 2$ holds for G. Let's remove a single edge from G, resulting in a graph H with k edges. Since we are only looking at connected graphs, let us only consider removing an edge that will result in a H where at most one vertex is disconnected (i.e. $v_G - 1 \leq v_H \leq v_G$). This will be useful for our induction hypothesis, and we have the prerogative to choose which edge to remove because we are not concerned with the planarity of disconnected graphs. We will always be able to find an edge in G whose removal will disconnect at most one vertex, as we will show now. There are two scenarios:

1. If G has no cycles (in our definition of cycle, we include parallel edges / self-loops, since those also create "faces"), then we have a tree structure, and there will be at least one "leaf." The leaf vertex a only has one degree, and we remove this edge, creating H where only a is disconnected.
2. If G has a cycle, remove any edge in the cycle. Having a cycle means that any two vertices in G are connected by two different paths; removing a single edge will guarantee that all vertices in G are still connected, if only by one path now.

Let's analyze these two scenarios more closely. In the first case, H has a disconnected vertex a. If we also remove a, then the resulting graph J will have $v_G - 1$ vertices, k edges, and f_G faces. The face count doesn't change since we aren't "combining" any cycles. Note that J will be planar; J is connected, and because none of the edges in G intersected at non-vertex points, none of the edges in J will intersect at non-vertex points either. Luckily, our induction hypothesis guarantees that $v - e + f = 2$ for connected graphs with k edges, so we see that:

$$(v_G - 1) - k + f_G = 2 \tag{1}$$

Let's manipulate the equation...

$$v_G - (k + 1) + f_G = 2 \tag{2}$$

We realize 2 is exactly the equation for our graph G, which has $k + 1$ edges, v_G vertices, and f_G faces. So we have finished the proof for scenario one.

In scenario two, H has v_G vertices and remains connected. However, because we disturbed a cycle, we are "combining" two adjacent faces together, so H has $f_G - 1$. Since we removed one edge, H has k edges. Note that H is still planar because it is connected and because removing an edge will not make the other edges intersect each other at non-vertex points. Our induction hypothesis then guarantees:

$$v_G - k + (f_G - 1) = 2 \tag{3}$$

Rearranging this equation, we get:

$$v_G - (k+1) + (f_G) = 2 \tag{4}$$

We realize 4 is exactly the equation for our graph G, which has $k+1$ edges, v_G vertices, and f_G faces. So we have finished the proof for scenario two as well, and we are done.

□

2.b

Proof. We will do a proof by contradiction, assuming that the complete $K_{3,3}$ graph is actually planar. We can easily see that a complete $K_{3,3}$ graph has 6 vertices and 9 edges. Since we are assuming it to be planar, it would have 5 faces to satisfy $v - e + f = 2$. In lecture 9, we proved that bipartite graphs can only have cycles of even length. Since $K_{3,3}$ does not have parallel edges, the shortest cycle length is 4; in other words, the smallest "face" contains at least 4 edges. In addition, we note that an edge can be part of at most two different faces. Thus, in order to have 5 faces, we would need at least $\frac{4 \cdot 5}{2}$ edges. This is greater than the 9 edges we have, so it is impossible to have 5 faces. Here is a contradiction, so $K_{3,3}$ must not be planar.

□

Problem 3

Proof. We first note that the length of the shortest Hamilton cycle (denoted as $|S|$) will be greater than the size of a minimum spanning tree (denoted as $|T|$) since all the weights are positive. This is because the minimum spanning tree represents the shortest (or least "weighty") way of connecting all the cities, and only has $n-1$ edges, while the Hamilton cycle has n edges. Suppose for contradiction that our MST T has a greater size $|T|$ than our shortest Hamilton cycle S. We would be able to find a smaller MST T by taking away an edge from our Hamilton cycle S, and this MST T would be smaller than our Hamilton cycle S. This contradicts the minimality of our original MST, T, so $|T| < |S|$.

Before moving forward, we also realize that the triangle inequality can be expanded to more than three points. For example, for four points (w, x, y, and z), the direct distance between two points (i.e. w and x) is smaller than (or equal to) the wrap-around distance (i.e. w to y to z to x). This can be shown using induction on the number of points, with the base case being three points. Let's assume that the "triangle inequality" holds for k points, so that the RHS of the following inequality contain k edges:

$$w(u,y) \leq w(u,v) + ...w(w,x) + w(x,y) \tag{5}$$

That is our induction hypothesis, and we want to show whether:

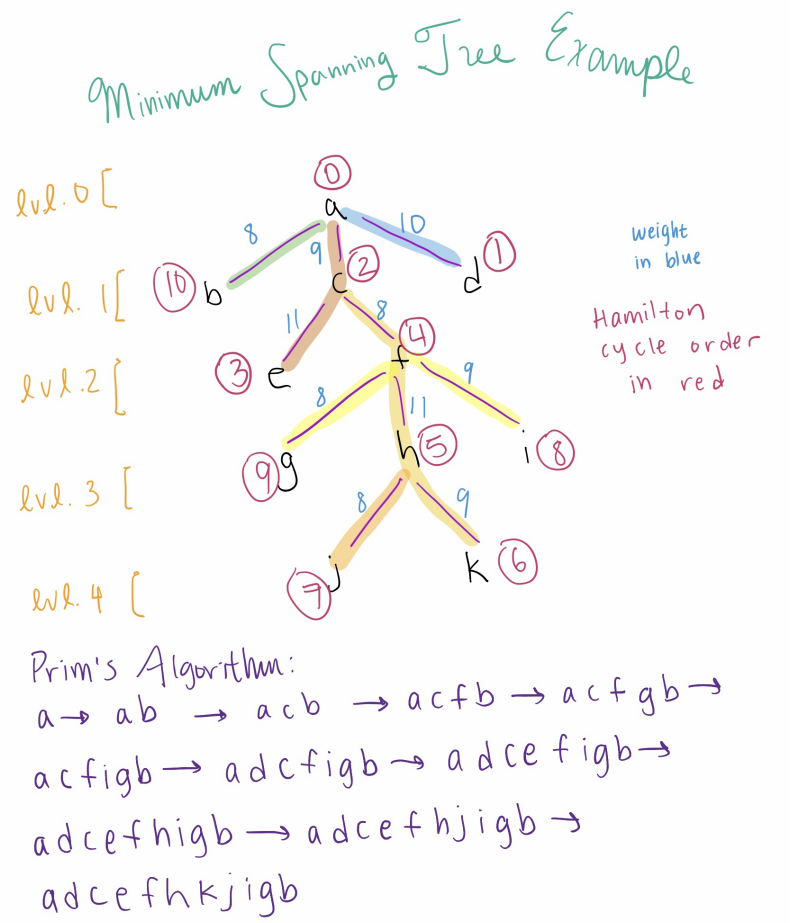
$$w(u,y) \overset{?}{\leq} w(u,v) + ... + w(w,x) + w(x,z) + w(z,y) \tag{6}$$

also holds. Note that now the RHS has $k+1$ points, although the extra point z could've been added at any point in the path, but we place it after x without loss of generality. We note that by the triangle inequality, $w(u,y) \leq w(u,z) + w(z,y)$, so we can rewrite 6 as:

$$w(u,y) \leq w(u,z) + w(z,y) \overset{?}{\leq} w(u,v) + ... + w(w,x) + w(x,z) + w(z,y) \tag{7}$$

It's easy to show the second inequality (the one with the question mark over it) because by the induction hypothesis, $w(u,z) \leq w(u,v) + ... + w(w,x) + w(x,z)$. Thus, we have extrapolated the triangle inequality to more than three points. All of this will be very helpful later on. Ok, let's move onto the meat of the proof.

We can think of the Prim algorithm described in the homework footnote in a more intuitive way by directly looking at the full MST generated by Prim and running BFS on our MST from the starting vertex. Running BFS should be very straightforward since we already know the parents of each vertex from running Prim. We put the starting vertex at the top, in level 0, the children of the starting vertex in level 1, and so on. Refer to the MST example diagram for a clearer idea. This diagram is just an example, but it gets the idea across well; note that it is of the MST, and not the originally fully-connected graph (we could think of the fully-connected graph as a graph as having edge weights as shown in the MST and then having all un-shown edges be of weight 12). The way Prim works is that when we visit a certain vertex A in level k, we will not visit any of the other (as of yet unvisited) vertices in level k until we have visited every vertex in the branch that A forms. If we get to a branching point B in level m within the branch that A forms (so $m > k$), we will not visit any other (unvisited) points in level m until all the vertices in branch B are visited, and so on. A more thorough (and perhaps unnecessary) explanation is given in the following paragraph, and we will tie together these ideas in the conclusion to show that our cycle is a 2-approximation.



In constructing our Hamilton cycle, we start at level 0 and visit the closest vertex (aka the adjacent vertex with the least weighty connection) in the layer below. In our diagram, we would be going from vertex a to vertex b. Then, we find a vertex that requires the least weight to be connected to our current path of a and b. That would be vertex c. Since vertex c is least connected to vertex a, we add c after a, and our current path is $a \to c \to b$. We continue this until we put all the vertices in our Hamilton cycle. Note that for a given branch point v, we are "seeing" adjacent vertices with "lighter weight" before "seeing" adjacent vertices with "heavier weight", but we are always putting the chosen vertex immediately after v. This means that the complete Hamilton cycle will go from v to the heaviest adjacent vertex u, traverse all vertices under branch u (or if u is a leaf, then just traverse u), before moving to a lighter adjacent vertex (adjacent to v). In addition, we traverse entire branches (or sub-branches) before moving onto other branches (or sub-branches) starting at the same level. Because we have a tree, a vertex can only be "seen" through its parents, and it will always be added after its parent and, more importantly, before any (as of yet "unseen") vertices in the same level as the branching point. In other words, on our Hamilton cycle, all vertices in between the branching vertex v and another vertex in the same level as v are children / grandchildren etc. of vertex v. Thus, this explains the intuition in the previous paragraph, where realized we are traversing the vertices based on which branch they are in. The colors of the highlighted paths (in the diagram) give a good idea of how the branches are traversed. After starting at vertex a, we go to vertex d (the heaviest adjacent vertex to our starting vertex a), then to c (the second heaviest adjacent vertex to our starting vertex a), traverse the entire branch of c,

then finally to vertex b (the lightest adjacent vertex to our starting vertex a). This diagram is here for better explanation (I tried explaining it without a diagram, but it was so hard!), but the logic of traveling down an entire branch before moving onto another branch starting at the same level holds in all cases due to the nature of the "insert". Another point is that the Prim algorithm for Hamilton cycles works on the fully connected graph G, but since we already have an MST, we can easily figure out the sequence of vertices in the Hamilton cycle because the "lightest weight connecting the current tree to a non-tree vertex" has already been cherry-picked in the MST.

Now, let's tie everything together. Let's call the size of the Hamilton cycle generated by Prim $|H|$, the size of the shortest Hamilton cycle $|S|$, and the size of the MST $|T|$. We want to show that $|H| \leq 2|S|$. Let's start! We have our Hamilton vertex order, and $|H|$ is determined by traveling along these vertices in the original, fully-connected graph G. For fun, let's imagine if we took away all the edges in G except for the edges in the MST. If we were to still travel the list of vertices in order, we would have to "go back" to the first common parent of successive vertices. For example, going back to our diagram's MST, we would travel from $a \to d \to a \to c \to e \to c \to f$ etc. This would cause every edge to be traveled twice–once going down to visit the child vertex and once coming back up to the parent vertex. This cycle (no longer a Hamilton cycle!) has a size of $2|T|$. Due to the triangle inequality (and its more general version with more than three points), we would have a shorter path if we could just travel according to the Hamilton cycle without needing to go back to common parent nodes; going back to the common parent node would require at least two edges between consecutive vertices in our ordered list of vertices, but the triangle inequality tells us that it would be better to go directly to the next vertex, since that would only require one edge. Luckily, the actual Hamilton cycle we generated is to be traveled on the fully-connected graph and does just that. In other words, $|H| \leq 2|T|$. At the beginning of the problem, we noted that $|T| < |S|$, so $2|T| < 2|S|$, and $|H| < 2|S|$. This is exactly the definition of a 2- approximation, and we are done.

□

Problem 4

We will do a direct proof to show that Hall's Marriage Theorem conditions are satisfied. Hall's Marriage theorem says that for a bipartite graph $G = (V_1 \cup V_2$ where $|V_1| = |V_2|$, there exists a perfect matching in $G \iff$ for every $A \subset V_1$, $|A| \leq |N(A)|$, where the $N(A)$ notation represents the set of neighbor vertices. We will first show that for our graph G, $|V_1| = |V_2|$ before proving the neighboring set requirement.

Since G is r-regular, it must have exactly $\frac{2 \cdot n \cdot r}{2}$, or nr edges in total, connecting vertices in V_1 to vertices in V_2. Since all vertices in V_1 and V_2 have degree r, that means there must be n vertices in V_1 and n other vertices in V_2. In other words, $|V_1| = |V_2|$. Now, we want to show that for every $A \subset U$, we have $|A| \leq |N(A)|$. Let's call the set of edges with an endpoint in $|A|$ as E_A. Since all vertices in A have degree r, $|E_A| = r|A|$. Let's call the set of edges with an endpoint in $N(A)$ as $E_{N(A)}$. Again, $|E_{N(A)}| = r|N(A)|$. Now, we note that $E_A \subseteq E_{N(A)}$ because all of the edges with an endpoint in A also have an endpoint in $N(A)$, so these edges would be counted in $E_{N(A)}$. This means that:

$$|E_A| \leq |E_{N(A)}| r|A| \leq r|N(A)||A| \leq |N(A)| \tag{8}$$

Now Hall's Marriage condition is satisfied, so there must be a perfect matching in G. Since G has $2n$ vertices, the size of the perfect matching would be size n.

Problem 5

Please refer to the attached diagram. We find a maximum flow of size 19.