CSDS 455: Homework 9

Shaochen (Henry) ZHONG, `sxz517`

Due and submitted on 09/23/2020
Fall 2020, Dr. Connamacher

Problem 1

Proof. Suppose we have k edge-disjoint $a \to b$ paths in G, this suggest there are at least k entirely distinct ways of getting from vertex a to b. So intuitively, we need to cut out all of these paths to ensure a to be disconnected from b. And it is known that to cut off a path, at least 1 edge needs to be removed, thus for a graph with k edge-disjoint $a \to b$ paths, we need to remove k edges to make a to be disconnected from b.

Note when selecting an edge to remove from an edge-disjoint path, such edge must be an common edge of all paths from $a \to b$ which share edge(s) with this particular edge disjoint path. □

Problem 2

I consulted https://math.stackexchange.com/questions/3113602/ *for this problem.*

Proof.

> *Proof.* **Menger's Theorem: For a connected, finite undirected graph G. The minimum vertex cut for $u, v \in V(G)$ is equal to the maximum number of vertex-disjoint paths from u to v.**
>
> Say we have k vertex-disjoint paths from u to v. We will need to remove a vertex to break a path (similar to *Problem 1*, we will need to remove the common vertex of all paths from u to v where these path has a shared vertex with the path we removing vertex from), so we must remove k vertices to disconnect u to v.
>
> **By promoting this proof to all vertex pairs, this means any k-connected graph will have k (internally) vertex-disjoint paths from any vertex pair in G.**
>
> □

With the lemma proven, we may arbitrarily select k desired vertices and find a cycle C of G which has j common vertices with the k desired vertex set (we call this set D_k), say $v_1, v_2, ..., v_j$. If $j = k$, then the statment is automatically proven.

If $j < k$ but $|C| \geq k$, by the lemma and knowing that G is a k-connected graph, we know that there must be k vertex-disjoint paths from C to v_k, where $v_k \in D_k$ and $\notin C$. We also know that these k vertex-disjoint paths from C to v_k can end on k different vertices on C.

Thus, we may find two adjacent vertices on C (denotes them v_i and v_{i+1}) and replace the edge between them with a path[1] of $v_i \to v_k \to v_{i+1}$. Since there are k paths between C to any vertex in D_k, we may do replace-edge-with-a-path manuver entil all k vertices in D_k is included in the cycle.

If $j < k$ but $|C|$ =k-1, there must be $k-1$ vertex-disjoint paths from C to v_k, each ending on a different vertex on C. We know that $|C|$ must be 2 as otherwise C will not be a cycle, so we may always find two adjacent vertices v_i and v_{i+1} on C. Replace the edge between them with the path $v_i \to v_k \to v_{i+1}$, now we have included the only leftover desired vertex v_k into the cycle.

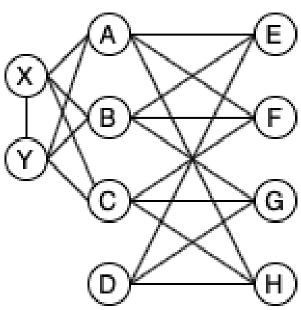
With $\geq k$ and $k-1$ both being true, and known that $k=2$ is trivially true[2], we have proven the statement by induction.

$\square$

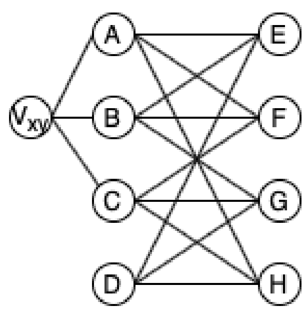
Problem 3

The to-be-prove statement is "Let G be a 3-connected graph. Prove that G/xy is 3-connected if and only if $G\{x, y\}$ is 2-connected." This suggests for a 3-connected G, if we have a 3-connected G/xy, we should be able to have a 2-connected $G\{x, y\}$.

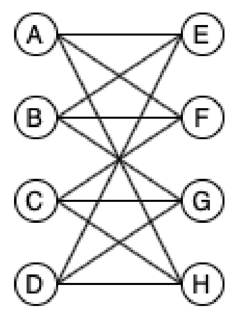
It seems I can find the below G as a counterexample for this statement. It seems as long as the connectivity of G is depended on vertices x, y, $G\{x, y\}$ can have a uncontrollable connectivity. I don't know if it is due to my interpretation of the question instruction.



G (3-connected)



$G/_{xy}$ (3-connected)



G - {x, y} (3-connected)

[1] This path must exist as there will be at least k distinct vertices connecting v_i or v_{i+1} to v_k, so we can always connect v_i to one of the vertex, reach v_k via a path, and get back to v_{i+1} from another vertex via another path. Notices we use two intermidiate vertices here, so the graph must be at least 2-connected

[2] Due to any two vertices in a 2-connected graph will have 2 vertex-disjoint paths between them. So by identifying the two desired the vertices and connecting the two vertex-disjoint paths between them, we will automatically have a cycle containing them.