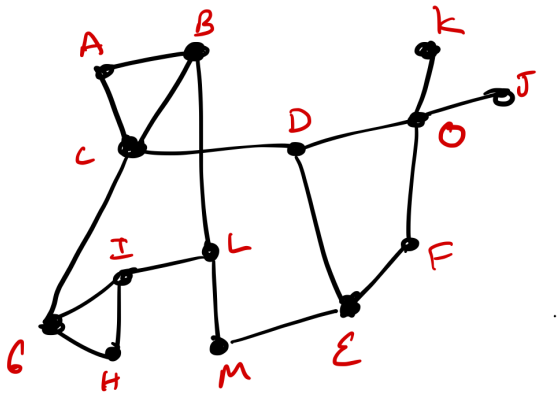
CSCI 1311: (non-comprehensive) Final Review Worksheet
[SOLUTIONS]

29 Apr. 2020

---

1. Consider the following graph G below



(a) Describe the set V and E for the graph?

$$V = \{A, B, C, D, E, F, G, H, I, J, K, L, M, 0\}$$
$$E = \{(A, B), (A, C), (B, C), (C, D), (D, O), (O, K), (O, J), (O, F), (F, E), (D, E), (M, E), (M, L), (L, I), (I, H), (I, G), (G, H), (G, C)\}$$

Note, using either () or {} for edges is fine as the graph is undirected.

(b) Create a matrix representation for the graph?

$$\begin{bmatrix}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0
\end{bmatrix}$$

(c) What are the graphs articulation point(s)?

O

(d) Draw the minimum spanning tree using DFS starting from vertex D? Break ties by alphabetic ordering.

(e) What was the ordering of the DFS traversal that produce the spanning tree above?

$D, C, A, B, L, I, G, H, M, E, F, O, J, K$

(f) Draw the minimum spanning tree using BFS starting from vertex D? Break ties by alphabetic ordering.

(g) What was the ordering of the BFS traversal that produce the spanning tree above?

$D, C, E, O, A, G, M, F, K, J, B, I, H, L$

(h) Using the BFS, find the radius of each vertex?

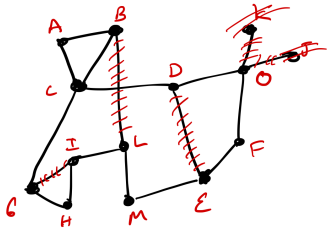
A	4
B	4
C	3
D	3
E	4
F	5
G	4
H	5
I	5
J	5
K	5
L	5
M	5
O	4

(i) What is/are the center(s) of the graph?

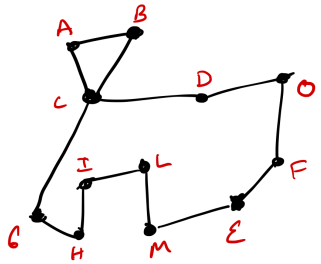
C, D, G are the centers because they have min radius.

(j) Find a subgraph G' of G that has an Euler Circuit that contains the maximum number of vertices? Draw it and describe why it has a Euler Circuit.

We must find a sub graph where all vertices have even degree. We start by removing edges until we have even degrees. However, since O is a articulation point, we cannot include K and J in the Euler trail as removing their edges from O disconnects the graph.

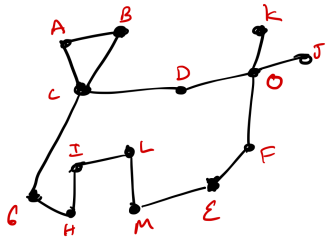


Leaving us the following sub graph.



(k) Find a subgraph G'' of G that has an Euler Trail that contains all the vertices of G? Draw it and describe why it has a Euler Trail.

To find a trail, we need all degrees to be even, except two. Working from the subgraph above, we can add back any edge that creates two odd degree vertices, but since we want all the vertices of G, we need to add back the edges that connect J and k, as well as J and k.



2. Prove, using induction on the number of vertices $n >= 2$ in the tree, that if you remove any edge from a tree T, you get a forest of two Trees, T_1 and T_2.

Proof by induction on the number of vertices on a tree n.

- $P(2)$: With two vertices,to be a tree, there must be one edge connecting them. If we remove that edge, we get two vertices, disconnected, each a trivial tree. Thus we get a forest of two trees.

- $P(n) \implies P(n+1)$: We must show that when the property is true with n vertices (IH), it is also true with $n+1$ vertices (TS).

  Aside: Note that from theorem we proved in class, that if we have a tree with n vertices, it has $n-1$ edges. If we add one vertex to that tree, then we have a tree because it is connected graph and has $n+1$ vertices and n edges.

  By the IH, we can decompose a tree T with n vertices into two trees, each with T_a with n_a vertices and $n_a - 1$ edges, and T_b with n_b vertices and $n_b - 1$ edges. Note that $n = n_a + n_b$

  Now considering adding any vertex v and edge e that connects to any vertex in T to produce a new tree T' with $n+1$ vertices and n edges. From the aside above, we know that T' is a tree. During decomposition, there are two cases to consider. Either e is the edge removed, or it is some other edge in T. If it is some other edge in T, then we know it is attached to either T_a or T_b by the IH.

  - Case 1: e is removed. We have a forest of two trees, T and the trivial tree containing just v.
  - Case 2: e is not removed. Then by IH, e would connect v to either T_a or T_b.
    * Case 2a: v is connected to T_a. Then T_a has $n_a + 1$ vertices and n_a edges, thus is a tree. And by IH T_b is also a tree.
    * Case 2b: v is connect to T_b. Then T_b has $n_b + 1$ vertices and n_b edges, thus is a tree. And by IH, T_a is also a tree.

  Thus in all cases, removing an edge from a tree with $n+1$ vertices leaves a forest of two trees.

QED.

3. Prove, using induction on the height of a tree, that a full/complete binary tree T with height $h \geq 0$, that there are an odd number of internal nodes.

By induction on the height h of tree T, we can show that for all $h \geq 1$ there are an odd number of internal nodes.

- $P(1)$: This is a tree with either three nodes. One node is the root and two are leafs. There there are an odd number of internal nodes.
- $P(h) \implies P(h+1)$: Can we show that if a tree T with height h has odd internal nodes, than a tree T' with height $h+1$ has odd internal nodes.

  Consider that if we remove all the leafs from T' we have the tree T of height h from our IH. From the IH we learn that number of internal nodes m is odd, or $m = 2k + 1$.

  Aside: Note, that in a full tree, the number of nodes on each level ℓ is 2^ℓ.

  In the last row of T, there are 2^h number of leaf nodes, each of which become internal nodes when we add back in the last row to get T'. So the number of internal nodes m' in T' is

$$m' = m + 2^h$$
$$m' = 2k + 1 + 2^h$$
$$m' = 2(k + 2^{h-1}) + 1$$

  Thus m' is odd.

QED.

4. Consider the Boolean algebra of digital logic. In that language, we add the $\odot$ operator with the following truth table

x	y	$x \odot y$
0	0	1
0	1	0
1	0	1
1	1	0

Define the Relation $x \, R \, y$ if, and only if, $x \odot y = 1$

(a) Prove, or provide a counter example, that the $\odot$ operator is reflexive.

For $\odot$ to be reflexive, it must be the case that $x \, R \, x$, or that $x \odot x = 1$, however $1 \odot 1 = 0$ but $0 \odot 0 = 1$, so it can't be reflexive.

(b) Prove, or provide a counter example, that the $\odot$ operator is symmetric.

For $\odot$ to be transitive, it must be the case that if $x \, R \, y$ then $y \, R \, x$. But this is not the case because $1 \odot 0 = 0$ but not $0 \odot 1 = 1$.

5. Using the same truth table as above for $\odot$

(a) What is a DNF formula for $\odot$?

$$x \odot y = x'y' + xy'$$

(b) What is a CNF formula for $\odot$?

$$x \odot y = (x + y')(x' + y')$$

(c) Use equivalence statement to show that CNF and DNF for $\odot$ are equivalent.

$$\begin{aligned}
x \odot y &= (x + y')(x' + y') && \\
&= (x'y)'(xy)' && \text{DeMorgan} \\
&= ((x'y) + (xy))' && \text{DeMorgan} \\
&= ((x' + xy)(y + xy))' && \text{distributive} \\
&= ((x' + x)(x' + y))((y + xy))' && \text{distributive} \\
&= ((x' + y)(y + xy))' && \text{complement/reduction} \\
&= ((x' + y)((y + x)(y + y))' && \text{distributive} \\
&= ((x' + y)(y + x)y)' && \text{complement/reduction} \\
&= ((x' + y)y)' && \text{subsumption} \\
&= (y)' && \text{subsumption} \\
&= y' && \\
&= y'(1) && \text{identity} \\
&= y'(x' + x) && \text{negation} \\
&= x'y' + xy' && \text{distributive/associative}
\end{aligned}$$

6. Consider a machine that takes in votes from a set of judges. Judges either vote for (1) or against (0). The machine produces an affirmation (1), when there is a lone dissent, either $2/3$ in favor, one against, or $2/3$ in against, and one in favor.

   (a) Create a truth table for the judges.

j_1	j_2	j_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

   (b) Use a K-map to find a simplified statement for the judges functions.

		00 $j_1'j_2'$	01 $j_1'j_2$	11 j_1j_2	10 j_1j_2'
0	j_3'	0	1	< 1	$1 >$
1	j_3	1)	1	0	(1

$$f = j_1'j_2 + j_1j_3' + j2'j3$$

Note: The goal is to cover with the least number of covers, which leads to the least number of terms. So while there may be some equivalent forms, this provides the smallest number of terms.