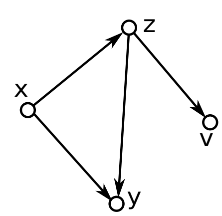
CS420/520: Graph Theory with Applications to CS, Winter 2017

Midterm

Thr, March/2/17

(1) This exam has 11 problems, each worth 10 points. (2) This is a closed book exam, and notes are NOT allowed. (3) The exam is **90 minutes** long.

Problem 1. Give the adjacency *matrix* of the following unweighted directed graph.



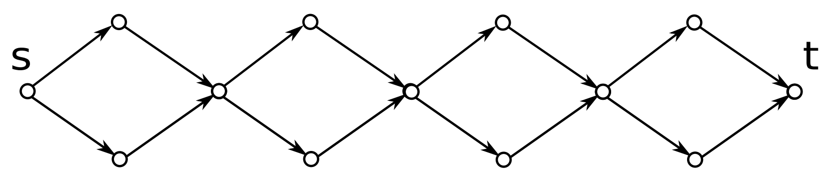
Problem 2. Let $G = (V, E)$ be a connected undirected graph with $n = 5k + 1$ vertices. We have run BFS on G from a vertex $s \in V$. Suppose, (i) there is one vertex in level zero, namely s, and (ii) there are exactly five vertices in each other level. What is the maximum and minimum number of possible edges for G?

Problem 3. Not having enough memory, one decides to search a graph without marking the vertices. Here is her code:

```
procedure DoryDFS(v)
    for all v → x ∈ E do
        DoryDFS(x)
```

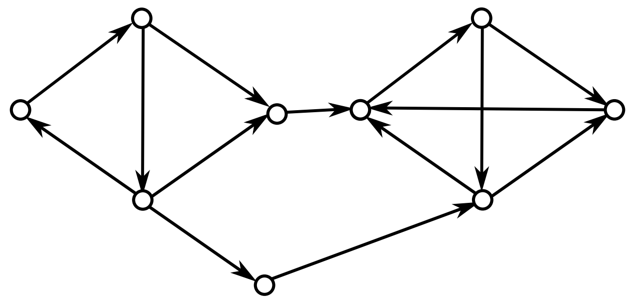
(a) Does DoryDFS always finish? If yes, argue why, if no, give a counter example.

(b) Consider the following graph, s, and t. Suppose DoryDFS(s) is called. How many times does the algorithm visit t before it stops? (You can say "unbounded" if you think that the algorithm visits t infinitely many times)



(c) Suppose $G = (V, E)$ is a graph, and $s \in V$. Does DoryDFS(s) end in polynomial time (in terms of V and E) for a DAG input? If yes, argue why, if no, provide a counter example.

Problem 4. Consider the following graph.



(a) Mark all strongly connected components.

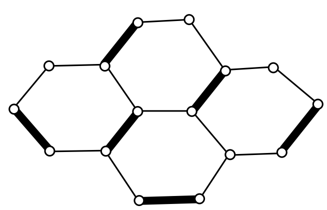
(b) Give the meta (directed acyclic) graph.

(c) Give a topological sort of the meta graph.

Problem 5. Let $G = (V, E)$ be a weighted DAG, and let $s, t \in V$. Design an $O(E + V)$ time algorithm to compute the shortest s-to-t path. For simplicity assume that there exists at least one s-to-t path. Also, your algorithm should only report the length of the shortest path. (Hint: this is very similar to HW2, you should be able to obtain it by modifying DFS)

Problem 6. Let $G = (V, E)$ be a directed weighted graph, with exactly one negatively weighted edge, and let $s, t \in V$. Give an $O(E \log V)$ time algorithm to compute the shortest s-to-t path. (Comment: Dijkstra runs in $O(E \log V)$ time, but, it cannot handle negative edges. Shimbel runs in $O(EV)$ time, and it can handle negative edges.)

Problem 7. Answer the following questions about the matching in the figure.



(a) Show an alternating path that is *not* augmenting.

(b) Show two disjoint augmenting paths.

Problem 8.

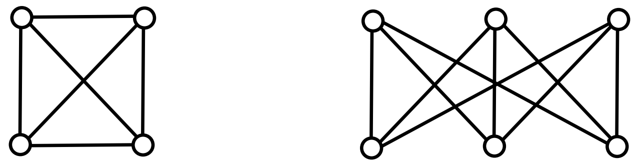
(a) Give an example of a connected graph with 20 vertices whose maximum matching size is one.

(b) Give an example of a connected graph G, and a maximal matching M in G such that the size of M is exactly half the maximum matching size in G.

(c) Give a bipartite graph with vertex set $\{1, \ldots, 6\}$ and minimum number of edges whose Tutte polynomial is G is $x_{1,1}x_{2,2}x_{3,3} - x_{1,1}x_{2,3}x_{3,2}$.

(d) Give a bipartite graph with vertex set $\{1, \ldots, 6\}$ and maximum number of edges whose Tutte polynomial is G is $x_{1,1}x_{2,2}x_{3,3} - x_{1,1}x_{2,3}x_{3,2}$.

Problem 9. K_n is a complete graph with n vertices, and $K_{m,n}$ is a bipartite graph with m vertices on one set, and n vertices on the other set, such that every pair of vertices in two different sets are adjacent. The following figure illustrates K_4 and $K_{3,3}$.



Answer the following questions about complete (bipartite) graphs.

(a) What is the number of different Hamiltonian cycles of $K_{3,3}$?

(b) What is the number of different Hamiltonian cycles of $K_{n,n}$?

(c) What is the number of different Hamiltonian cycles of K_n?

(d) What is the number of different perfect matchings in $K_{n,n}$?

(e) What is the number of different perfect matchings in K_n for an odd n?

(f) What is the number of different perfect matchings in K_n for an even n?

(Comment: a perfect matching is a matching, in which all vertices are matched, and a Hamiltonian cycle is a cycle that visits each vertex exactly once.)