MAT332 - Fall 2016: Final Exam Topics

Important theorems that you should be able to state, but not prove:

- A graph has an even number of odd-degree vertices
- A connected graph has an Eulerian circuit iff every vertex has even degree
- The following are equivalent: G is connected and has no cycles; G is connected and has $n-1$ edges; G has $n-1$ edges and no cycles; between any two vertices of G exists exactly one path.
- König's theorem
- Hall's Theorem
- Every regular bipartite graph has a perfect matching
- Tutte's Theorem
- Menger's Theorem
- Kuratowski's Theorem.
- The chromatic number of a graph is $\leq \Delta(G)+1$.

Important theorems that you should know the statement AND proof of:

- The Min-Cut Max-Flow theorem (the "proof" consists of showing that when the F-F algorithm terminates, the source-sink cut $[R, V(G)-R]$ has capacity $\text{val}(f)$)
- Edge Menger's theorem (the "proof" consists of deriving it from Menger's theorem using the edge graph)
- Every interval graph has $\omega(G)=\chi(G)$.
- $\chi(G;k)$ is a polynomial
- $R(3,3)=6$.
- Ramsey's Theorem
- Euler's formula ($|V(G)|-|E(G)|+|F(G)|=2$) for connected planar graphs.
- $K_{3,3}$ and K_5 are non-planar (a proof of this requires showing that $|E(G)|\leq 3|V(G)|-6$, and for triangle-free graphs, $|E(G)|\leq 2|V(G)|-4$)
- $\lfloor n/3 \rfloor$ people are required to "guard" a polygon with n vertices. (If I ask you to prove this, I will expect you to know the general strategy of finding a way to cut the graph into two pieces so that you can apply induction to show that a triangulation exists and is 3-colourable, and how to place the "guards". I will not ask you to go into details, as this is too long).

Important algorithms whose purpose you should know, but I won't ask you to apply:

- Dijkstra's algorithm for finding shortest distance between two vertices in a weighted graph.
- Gale Shapley algorithm for preference matching

Important algorithms that you should be able to apply:

- Kruskal's algorithm for finding a minimum-weight spanning tree.

- Ford-Fulkerson algorithm for finding a minimal flow (there are many different implementations of this algorithm in the world, it's okay if you don't memorize the specific "breadth first" implementation we covered in class. Also, you don't need to know anything about networks with non-integer flows)
- Greedy algorithm for vertex coloring
- Recursive algorithm for counting spanning trees using edge contraction/deletion.
- Recursive algorithm for computing the chromatic polynomial using edge contraction/deletion

Important names (you should know the following concepts by their name):

- Konig's Theorem, Hall's Theorem, Menger's Theorem, Ramsey's Theorem, Kuratowski's Theorem, Euler's Formula, Kruskal's Algorithm, Dijkstra's algorithm, Gale Shapley algorithm, Ford-Fulkerson algorithm.

On the final, you will be asked to prove one (or more) of these theorems or apply one (or more) of these algorithms to a certain graph. You will not be asked to prove the "correctness" of the algorithms listed above, except that you are expected to know why the Ford-Fulkerson algorithm produces maximum flow (it's an essential part of the min-cut max-flow theorem)

Notice that to keep things managable, some items from the first half of class are not listed above (e.g. the theorem that graphs with sufficiently high vertex degrees have hamiltonian cycles, or that 3-regular graphs without a cut edge have a perfect matching). If you are wondering if you need to know these theorems for the final exam, the answer is *no*. But if you are wondering if you need to know these theorems to lead a fulfilling life, the answer is *yes*.

There will also be a section where you are asked to consider some real-world problem and translate it into a graph theory problem. For example, maybe I'll ask you to imagine a situation where you need to schedule a bunch of students' final exam times without creating conflicts, and you'll be asked to translate the problem into a graph theory problem.

There will also be three new proofs and a few short answer questions.

Some questions – ones marked (!) are proofs that I almost put on the exam, but didn't (because I had other questions I liked better).

- (!) Let G be a connected graph with exactly two vertices of odd degree. Prove that G has an Eulerian walk (not *cycle*!)
- T/F: A tree has at most one matching of maximal size.
- T/F: A tree has at most one perfect matching.
- (!) Let G be a tree with at least 2 vertices, and let S be the set of all vertices with degree ≥ 3. Prove that the number of leaves in a tree G equals $2 + \sum_{v \in S}(d(v) - 2)$
- Give an example of a matching that is *maximal*, but not *of maximum size*.
- (!) Prove that any connected k-regular bipartite graph, where $k \geq 2$, has no cut edge.
- Recall that $\kappa(G) \leq \kappa'(G) \leq \delta(G)$ for a simple graph G (here, κ is the connectivity, κ' is the edge connectivity, and $\delta(G)$ is the minimum degree of a vertex). Find examples where $\kappa(G) < \kappa'(G)$ and $\kappa'(G) < \delta(G)$
- Let G be a simple graph. Prove that G is 2-edge-connected if and only if every edge of G is contained in some cycle.
- (!) Let G be a plane graph. Prove that G is bipartite if and only if every face has even length.