Graph Theory Problem Ideas

March 20, 2017

Note: Please let me know if you have a problem that you would like me to add to the list!

1 Classification

Given a degree sequence \( d_1, \ldots, d_n \), let \( N_{d_1, \ldots, d_n} \) denote the number of isomorphism classes of graphs with that degree sequence. Find a formula for \( N_{d_1, \ldots, d_n} \). Find an algorithm that computes a graph in each isomorphism class.

Example. In the case of the degree sequence 2, 2, 2, 2, 2, 2, 2, 2, 2, the four different isomorphism classes are drawn at the top of page 65. We could write this as \( N_{2^9} = 4 \). Similarly, if you solve Exercise 3.16, you should realize that \( N_{6^3} = 4 \) (hint: use Theorem 3.1). What is the symmetry at work here? Can you generalize this to get an identity for \( N_{d_1, \ldots, d_n} \)?

Remarks. • Start with simple degree sequences! For example, compute \( N_{1^n} \) and \( N_{2^n} \).

• Solving this problem for \( N_{r^n} \) (\( r \)-regular graphs with \( n \) vertices) would already be extremely interesting.

• Ideally, you would find a closed formula for \( N_{d_1, \ldots, d_n} \) in terms of the variables \( d_1, \ldots, d_n \). If a closed formula proves to be elusive, it may be possible to show that the numbers \( N_{d_1, \ldots, d_n} \) for various choices of degree sequences can be assembled into a nice structure, such as a power series that has a nice factorization.

• One approach to computing \( N_{d_1, \ldots, d_n} \) is to find relationships between these numbers as you vary the degree sequences. For instance, you could investigate how the number changes when you subtract 1 from \( d_1 \).

• Sometimes small changes to a counting problem can lead to a much nicer formula. For instance, counting all pseudographs with a given degree sequence or counting only connected graphs with a given degree sequence may be easier.

2 Checking isomorphism

Devise an efficient algorithm to determine whether two graphs are isomorphic.
Remarks.  
- The brute-force algorithm we saw in class takes $n!$ time, which is considered to be extremely bad. You should aim for a polynomial-time algorithm (e.g. $n^3$).
- Once again, restricting to a special class of graphs (such as $r$-regular graphs or connected graphs) may make this problem more tractable.

3 Graphical degree sequences

Devise a list of conditions for degree sequences such that every degree sequence satisfying those conditions is graphical.

Remark. We already have an efficient (linear time) algorithm for determining whether a sequence is graphical. You could try to use this algorithm to help you devise the list of conditions.

4 Properties of a graph and its adjacency matrix

How are properties of a graph (such as the degree sequence, connectedness, bipartiteness, etc.) reflected in the adjacency matrix? Conversely, how are properties of the adjacency matrix (such as invertibility, eigenvalues, eigenvectors, etc.) reflected in the associated graph?

5 Graceful trees

Does every tree have a graceful labeling? (For the definition of graceful labeling, see Section 8.3.)

6 Partitioning edge-connectivity

Let $G = (V,E)$ be an $(a + b + 2)$-edge-connected graph. Does there exist a partition of $E$ into sets $A, B$ so that the subgraph $(V, A)$ is $a$-edge-connected and the subgraph $(V, B)$ is $b$-edge-connected?

(This problem was submitted to the Open Problem Garden by Matt Devos.)

Example. $K_5$ is $(1 + 1 + 2)$-edge-connected. Letting $A$ be the edges in any 5-cycle in $K_5$, the remaining edges of $K_5$ also form a 5-cycle. Cycles are 1-edge-connected (which just means connected), so the partition in the question does exist for $K_5$.

7 Decomposing an Eulerian graph into cycles

Is it true that every (simple) Eulerian graph of order $n$ can be decomposed into $\leq \frac{n-1}{2}$ cycles?

(In the Open Problem Garden, this problem is attributed to György Hajós.)

Example. A complete graph of odd order ($n = 2k + 1$) can be decomposed into exactly $k = \frac{n-1}{2}$ cycles.
8 Hamiltonian cycles

Devise a polynomial time algorithm for determining whether a graph has a Hamiltonian cycle.

Remark. This is an extremely hard problem in general. As usual, it could be helpful to restrict to an interesting class of graphs instead of general graphs.

9 Ramsey numbers

Develop an efficient algorithm to compute Ramsey numbers (such as \( r(K_3, K_{10}) \), \( r(K_4, K_6) \), or \( r(K_5, K_5) \)). Alternatively, invent a formula (such as Theorem 11.8 in the textbook) for a certain class of Ramsey numbers and prove your formula is correct.

Remark. Warning: this problem is harder than it looks! You might think that \( r(K_5, K_5) \) cannot be that hard to compute using a computer because \( K_5 \) is a fairly small graph, but you should realize that this Ramsey number is at least 43 and that there are \( 2^{\binom{43}{2}} \approx 7 \times 10^{271} \) possible red-blue colorings of \( K_{43} \). One thing you could try to use to your advantage is that \( K_{43} \) has a lot of symmetry; in fact its automorphism group consists of all permutations of 43 vertices, which is a group of order \( 43! \approx 6 \times 10^{52} \). Unfortunately, this is much much much smaller than the number of colorings, so even a brute force approach that makes use of this symmetry has no chance of considering all possible colorings.

10 Domination numbers

Develop an efficient algorithm for computing the domination number of a graph. Alternatively, try to prove one of the many conjectural inequalities such as the following: does every 3-connected cubic graph have domination number at most \( \lceil n/3 \rceil \), where \( n \) is the order of the graph?