Math 3322: Graph Theory[1] Mikhail Lavrov

Lecture 8: Degree sequences

September 11, 2023 *Kennesaw State University*

1 The graphic sequence algorithm

1.1 What the algorithm does

Let's summarize the algorithm from the previous lecture which tests if a sequence is graphic.

It begins by a cycle of steps that make the sequence shorter and shorter. Given a sequence of degrees, we:

1. **Sort** the sequence, so that it's ordered $d_1, d_2, \ldots, d_n$ with $d_1 \geq d_2 \geq \cdots \geq d_n$.
2. **Delete** the first term d_1, and subtract 1 from the next d_1 terms, leaving the sequence $d_2 - 1, d_3 - 1, \ldots, d_{d_1+1} - 1, d_{d_1+2}, \ldots, d_n$.

From here, there are several options:

- This sequence has $n-1$ terms. If they are not all in the range 0 to $n-2$, we immediately **give up**: the sequence we got is not graphic.
- If we get to a sequence which we know is graphic, we can **draw a graph** with that degree sequence. (It's enough for the algorithm to work if you do this when the sequence is $0, 0, \ldots, 0$, in which case we draw the empty graph.)
- Most of the time, we go back and **repeat** this operation.

Once we've simplified the sequence enough, we can draw a graph of that degree sequence. We will draw the graph so it has the degrees we want in order, from left to right.[2] Then we undo all the **sort** and **delete** steps we did above, in reverse order:

1. When we **undelete** a term d, we add a new vertex to the left of all the vertices, and add d edges from that vertex to the next d vertices in order.
2. When we **unsort** the graph, we rearrange our drawing so that the vertices have the degrees we want in the order we wanted them.

The goal: given a graph that realizes the length-$(n-1)$ degree sequence, the **undelete** and **unsort** operations should give us a graph that realizes the length-n degree sequence.

Then, we repeat these two steps until we get back to the original degree sequence.

[1] This document comes from the Math 3322 course webpage: http://facultyweb.kennesaw.edu/mlavrov/courses/3322-fall-2023.php

[2] The graph doesn't care how we draw it, but this is the order we'll use when talking about the "next" d vertices after the first vertex.

1.2 Guarantees about the algorithm

Given a sequence, the algorithm either gives up, or finds a graph with that degree sequence. The only worry is: will the algorithm ever give up, even though we started with a graphic degree sequence?

The algorithm only gives up when the current, shorter degree sequence is obviously not graphic. So another way to restate our worry is: do we ever go from a length-n graphic sequence to a length-$(n-1)$ sequence which is no longer graphic?

Our **delete** operation was based on imagining a realization of our sequence in which the vertex with degree d_1 is adjacent to the vertices with degrees $d_2, d_3, \ldots, d_{d_1+1}$. To prove that this is always valid, we want the Havel–Hakimi theorem:

Theorem 1.1 (Havel–Hakimi)**.** *Let G be a graph with vertex set $v_1, v_2, \ldots, v_n$ such that $\deg(v_1) \geq \deg(v_2) \geq \cdots \geq \deg(v_n)$. Let $d = \deg(v_1)$ (which is also the maximum degree $\Delta(G)$).*

Then there is a graph H with the same vertex set $v_1, v_2, \ldots, v_n$ and the same degrees of each vertex (with $\deg_G(v_i) = \deg_H(v_i)$ for all i) in which v_1 is adjacent to $v_2, v_3, \ldots, v_{d+1}$.

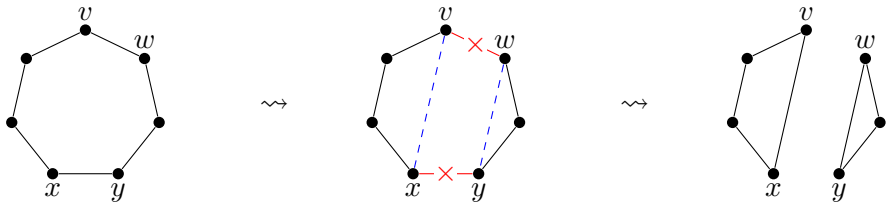
In other words: if the sequence $d_1, d_2, \ldots, d_n$ is graphic to begin with (if it has a graph G realizing it) then it also has a graph H realizing it. The conclusion of Theorem 1.1 is set up exactly so that when we do the **delete** step, the resulting sequence is the degree sequence of $H - v_1$. So we always go from a graphic sequence to another graphic sequence.

This makes sure that our algorithm always works: if the sequence is graphic, it will realize it, and if not, then it will give up eventually.

2 Proof of the Havel–Hakimi theorem

As in the statement of Theorem 1.1, let G be a graph with vertex set $v_1, v_2, \ldots, v_n$ such that $\deg(v_1) \geq \deg(v_2) \geq \cdots \geq \deg(v_n)$. Let $d = \deg(v_1)$.

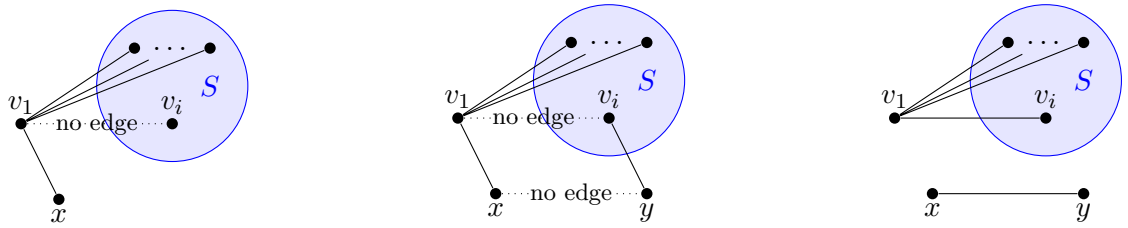
We will proceed by an operation called an **edge swap**, which changes a graph without changing its degree sequence. An edge swap is the following operation: whenever we find vertices v, w and x, y such that vw and xy are edges, but vx and wy are not, we delete the edges vw and xy then add new edges vx and wy. Here is an example of the "before" and "after" of this operation:



We begin by proving a lemma that essentially says "we can always use edge swaps to make progress toward what we want". For convenience, let $S = \{v_2, v_3, \ldots, v_{d+1}\}$: our goal is to make these vertices be neighbors of v_1.

Lemma 2.1. *If not all vertices of S are adjacent to v_1, we can perform an edge swap to* increase *the number of vertices in S which are adjacent to v_1.*

Proof. Suppose that $v_i \in S$ is not adjacent to v_1. Since $|S| = d$, and $\deg(v_1) = d$, in order for v_1 to have d neighbors, it must also be adjacent to some vertex $x \notin S$, as shown in Figure 1a.



(a) Vertices v_i and x (b) Goal: find y (c) The key edge swap

Figure 1: The three steps in the proof of Lemma 2.1.

To do an edge swap that replaces edge v_1x by edge v_1v_i, we need to find a fourth vertex: a vertex y such that v_i is adjacent to y, but x is not. (We should see Figure 1b in the graph.)

Here is where the order of degrees in G comes in. Since we sorted the degrees in descending order, the vertices in S have higher degrees than the vertices not in S (except for v_1). In particular, $\deg(v_i) \geq \deg(x)$. Moreover, x already has one neighbor that v_i does not have: the vertex v_1.

Therefore it is impossible for x to be adjacent to every neighbor of v_i, and also to v_1: then, we'd have $\deg(v_i) < \deg(x)$. So we can find a neighbor y of x such that v_j is not adjacent to y.

Now perform the edge swap which replaces v_1x and v_iy by v_1v_i and xy (as in Figure 1c). After this switch, v_1 is adjacent to one more vertex of S. □

I put Lemma 2.1 first because from here, the proof of Theorem 1.1 is essentially just an application of the lemma, together with the extremal principle. It is a good exercise to try to do this yourself before reading ahead.

Proof of the Havel–Hakimi theorem. Of all graphs with the same vertex set and vertex degrees as G, let H be chosen to maximize the number of neighbors of v_1 in S.

Then actually, all vertices of S must be adjacent to v_1. If not, we could use Lemma 2.1 to increase the number of neighbors of v_1 in S. But H was chosen to have the largest possible number of such neighbors, so this can't happen.

So we have the graph we wanted. □

3 More on degree sequences

We only used edge swaps in the proof of the Havel–Hakimi theorem, but they have other applications. To begin with, we can show the following theorem:

Theorem 3.1. *If two graphs G and H have the same vertex set V, and $\deg_G(v) = \deg_H(v)$ for all $v \in V$, then we can turn G into H by doing edge swaps.*

Proof. We prove this by induction on the size of V (the common vertex set of G and H).

When $|V| = 1$, there is nothing to show: there is only one possible graph on 1 vertex.

Assume this is possible for all pairs of $(n-1)$-vertex graphs, and let G and H both have n vertices. Let v be a vertex with the highest degree (in both G and H) and let S be the $\deg(v)$ vertices with the highest degrees after v. (This is the same S as in our proof of the Havel–Hakimi theorem and the edge swap lemma.)

By what we've already shown today, we can do edge swaps on G to get a graph G' in which v's neighbors are the vertices in S. We can do the same to H, getting a graph H' (and so there is also a sequence of edge swaps that turn H' into H, by reversing those edge swaps).

By induction, there is a sequence of edge swaps that turns $G' - v$ into $H' - v$ (both $(n-1)$-vertex graphs on the same vertex set). We can perform thse edge swaps on G' instead, and they will work equally well, turning G' into H'. Finally, we know that we can turn H' into H.

This shows that we can turn G into H, and so by induction, this works when G and H have any number of vertices. □

Why is this theorem useful? Well, we know how to answer two possible questions about potential degree sequences...

1. Is this sequence graphic? (Is it the degree sequence of a graph?)
2. How is this sequence graphic? (Find a graph with this degree sequence.)

...but there is a third, harder question we can ask:

3. What is a typical graph with this degree sequence?

That's deliberately vague. However, Theorem 3.1 can, surprisingly, help us answer this question.

What we can do is start with *some* realization of a degree sequence, then do lots of edge swaps at random, getting a random graph with this degree sequence.[3] We can approximately answer questions like "are any graphs with this degree sequence bipartite" or "what is the average diameter of a graph with this degree sequence" by sampling lots of random graphs and doing statistics with the results.

We can even approximately answer questions like "how many different graphs have this degree sequence?" To do this, just sample lots of graphs, then count how many repeats you get. Compare this to how many repeats you'd expect, if there were N possible graphs total. This should let you estimate the most likely value of N.

[3] Fine print: this does not sample a graph uniformly at random. Essentially, it samples a graph with probability proportional to how many edge swaps are possible to do in it. But that's a known bias we can account for.

4 Practice problems

1. Here are some more practice problems for the Havel–Hakimi algorithm. For each sequence, determine whether or not it is graphic, and if it is, find a graph with this degree sequence.

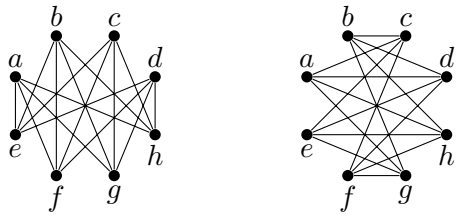
   (a) $5, 5, 5, 3, 3, 3, 3, 3$.

   (b) $7, 6, 6, 4, 4, 4, 2, 1$.

   (c) $3, 3, 3, 2, 2, 1, 1$.

2. If we can turn any graph G into any graph H with the same degree sequence, using edge swaps, then we can turn $K_{4,4}$ into $K_{4,4}$ using edge swaps.

   Specifically, let G be the graph on the left, and let H be the graph on the right. Corresponding vertices have the same degrees, and both of these graphs have the structure of $K_{4,4}$, but they are not the same graph: they have different edges. G has all edges between $\{a, b, c, d\}$ and $\{e, f, g, h\}$, while H has all edges between $\{a, b, e, f\}$ and $\{c, d, g, h\}$.



   Find a sequence of edge swaps to turn G into H. *(Hint: four edge swaps are enough.)*

3. Find a graph with 6 vertices and 9 edges in which is not possible to do any edge swaps.

   In particular, this means that there is only one possible realization of the degree sequence of the graph you found.

4. There are 7 possible graphs with the vertex set $\{v_1, v_2, v_3, v_4, v_5\}$ such that $\deg(v_1) = \deg(v_2) = 3$ and $\deg(v_3) = \deg(v_4) = \deg(v_5) = 2$. Find all of them, and demonstrate directly that we can turn any one of them into any other via edge swaps.

   (Hint: of the 7 graphs, there is a "central" example that is one edge swap away from all 6 others.)

5. Let G be a graph with the vertex set $\{v_1, v_2, \ldots, v_n\}$ such that

$$\deg(v_1) \geq \deg(v_2) \geq \cdots \geq \deg(v_n).$$

   Then $\deg(v_1)$ (the maximum degree of G) can be at most $n - 1$; in fact, it can be at most the number of nonzero degrees among $\deg(v_2), \ldots, \deg(v_n)$.

   (a) Explain why this is equivalent to the following inequality:

$$\deg(v_1) \leq \sum_{i=2}^{n} \min\{\deg(v_i), 1\}.$$

   Here, $\min\{a, b\}$ is simply the smaller of the two numbers a and b.

(b) Prove a 2-vertex version of this inequality:

$$\deg(v_1) + \deg(v_2) \leq 2 + \sum_{i=3}^{n} \min\{\deg(v_i), 2\}.$$

(Hint: consider the number of edges between v_1, v_2 and the rest of the graph. Why is the "2+" there?)

(c) In general, we have an inequality of the form

$$\sum_{i=1}^{k} \deg(v_i) \leq f(k) + \sum_{j=k+1}^{n} \min\{\deg(v_j), k\}$$

for each $k \in \{1, \ldots, n\}$. What is the best possible function $f(k)$ that makes this inequality true? (I ask for "best possible" because some incredibly big function like $f(k) = 2^{2^{2^k}}$ will also work, but will not be as useful as the smallest possible number you could put there.)

More is true: a result called the Erdős–Gallai theorem states that if a sequence satisfies the inequality in part (c) for every k, and the sum of the degrees is even, then the sequence is graphic. This is an alternative to the Havel–Hakimi algorithm for determining if a sequence is graphic.