Graph Theory

1 Introduction

Graphs are an incredibly useful structure in Computer Science! They arise in all sorts of applications, including scheduling, optimization, communications, and the design and analysis of algorithms. In the next few lectures, we'll even show how two Stanford students used graph theory to become multibillionaires.

But first we are going to talk about something else. Namely, sex. The question that we'll address is, on average, who has more opposite-gender partners, men or women?[1] In the popular literature, it seems that most people think the answer is "men". Not surprisingly, this has been the subject of many studies. In one of the largest, researchers from the University of Chicago interviewed a "random sample" of 2500 people over several years to try to get an answer to this question. Their study, published in 1994, and entitled *The Social Organization of Sexuality* found that on average men have 74% more opposite-gender partners than women.

Other studies have found that the disparity is even larger. In particular, ABC news claims that the average man has 20 partners over his lifetime, and the average woman has 6, for a percentage disparity of 233%. The ABC News study, aired on Primetime Live in 2004, claimed to be one of the most scientific ever done with only a 2.5% margin of error. It was called "American Sex Survey: A peak between the sheets"—- hmmmmmm, doesn't sound so scientific. The promotion for the study is even better:

> A ground breaking ABC News "Primetime Live" survey finds a range of eye-popping sexual activities, fantasies and attitudes in this country, confirming some conventional wisdom, exploding some myths – and venturing where few scientific surveys have gone before.

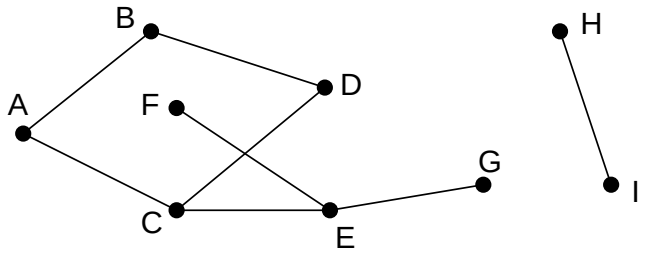
Probably that last part about going where few scientific surveys have gone before is pretty accurate!

Anyway, which do you think is more right, the University of Chicago or ABC News? How would we even attempt to figure out the answer to such a question? Fortunately, this is the kind of question that can be handled with graph theory. Whereas it might be more interesting to interview thousands of people about their sexual practices, we can get the answer a lot more efficiently by modelling the problem as a graph and doing a little analysis on the graph.

[1] Today, we're restricting our analysis to opposite gender partners. We're not making a political statement – it's just a lot easier to do the analysis with graph theory this way.

2 Graphs

Informally, a ***graph*** is a bunch of dots, some of which are connected by lines. Here is an example of a graph:



Sadly, this definition is not precise enough for mathematical discussion. Formally, a graph is a pair of sets (V, E), where:

- V is a nonempty set whose elements are called ***vertices*** (or *nodes*).
- E is a collection of two-element subsets of V called ***edges***.

The vertices correspond to the dots in the picture, and the edges correspond to the lines. Thus, the dots-and-lines diagram above is a pictorial representation of the graph (V, E) where:

$$V = \{A, B, C, D, E, F, G, H, I\}$$
$$E = \{\{A, B\}, \{A, C\}, \{B, D\}, \{C, D\}, \{C, E\}, \{E, F\}, \{E, G\}, \{H, I\}\}.$$

2.1 Definitions

A nuisance in first learning graph theory is that there are so many definitions. They all correspond to intuitive ideas, but can take a while to absorb. Some ideas have multiple names. For example, graphs are sometimes called ***networks***, vertices are sometimes called ***nodes***, and edges are sometimes called ***arcs***. Even worse, no one can agree on the exact meanings of terms. For example, in our definition, every graph must have at least one vertex. However, other authors permit graphs with no vertices. (The graph with no vertices is the single, stupid counterexample to many would-be theorems— so we're banning it!)[2] This is typical; everyone agrees more-or-less what each term means, but disagrees about weird special cases. So do not be alarmed if definitions here differ subtly from definitions you see elsewhere. Usually, these differences do not matter.

[2] Note that we *do* allow graphs without edges however.

Hereafter, we use A—B to denote an edge between vertices A and B rather than the set notation $\{A, B\}$. Note that A—B and B—A are the same edge, just as $\{A, B\}$ and $\{B, A\}$ are the same set.

Two vertices in a graph are said to be ***adjacent*** if they are joined by an edge, and an edge is said to be ***incident*** to the vertices it joins. The number of edges incident to a vertex is called the ***degree*** of the vertex. For example, in the graph above, A is adjacent to B and B is adjacent to D, and the edge A—C is incident to vertices A and C. Vertex H has degree 1, D has degree 2, and E has degree 3.

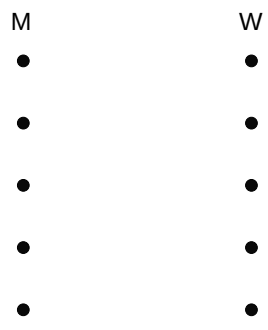
Deleting some vertices or edges from a graph leaves a ***subgraph***. Formally, a subgraph of $G = (V, E)$ is a graph $G' = (V', E')$ where V' is a nonempty subset of V and E' is a subset of E. Since a subgraph is itself a graph, the endpoints of every edge in E' must be vertices in V'.

In the special case where we only remove edges incident to removed nodes, we say that G' is the *subgraph induced on* V' if $E' = \{(x$—$y | x, y \in V' \text{ and } x$—$y \in E\}$. In other words, we keep all edges unless they are incident to a node not in V'.

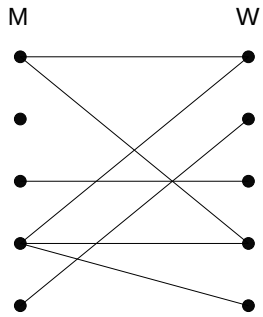
Let's restrict our attention to simple graphs: A graph is *simple* if it has no loops or multiple edges. A *loop* is an edge that has both endpoints at the same node, i.e., an edge of the form A—A. *Multiple edges* are two or more edges with the same pair of endpoints, such as A—B and A—B. A graph with multiple edges is called a *multigraph*.

2.2 Sex in America

Let's model our problem of opposite gender partners in graph theoretic terms. Let $G = (V, E)$ be a graph where the set of vertices V consists of everyone in America. Now each vertex either represents either a man or a woman, so we can partition V into two subsets: M, which contains all the male vertices, and W, which contains all the female vertices. Let's draw all the M vertices on the left and the W vertices on the right:



Now, without getting into a lot of specifics, *sometimes an edge appears* between an M vertex and a W vertex:



Actually, this is a pretty hard graph to figure out. Not only do we not know all the edges, but the graph is *enormous*. If we restrict ourselves to people in the U.S., as in the study, there are about 300 million nodes! Of this 50.8% are women and 49.2% are men. So $|V| = 300$ million, $|M| = 147.6$ million and $|W| = 152.4$ million. We don't even know how many edges there are in this graph!

But it turns out that we don't need to – we just need to figure out the average number of opposite gender partnerships for men and for women. Let A_m be the average number of opposite gender partnerships for men and let A_w be the average number of opposite gender partnerships for women. Our question to resolve is then: Does $\frac{A_m}{A_w} = 1.74$?

But how do we represent A_m, A_w in terms of our graph? It's just the average degree of the male nodes! That is:

$$A_m = \frac{\sum_{x \in M} \deg(x)}{|M|} = \frac{|E|}{|M|}$$

and the average degree of the female nodes is:

$$A_w = \frac{\sum_{x \in W} \deg(x)}{|W|} = \frac{|E|}{|W|}.$$

So,

$$\frac{A_m}{A_w} = \frac{|E|/|M|}{|E|/|W|} = \frac{|W|}{|M|} = 1.0325...$$

So the University of Chicago study was way off. After a huge effort, all they managed to give was a totally wrong answer. The average man has 3% more opposite gender partnerships than the average woman and the answer really has nothing to do with any differences in their behavior. Rather, it just has to do with the relative number of men and women. Collectively, men and women have the same number of opposite gender partnerships, since it takes one of each set for every partnership, but there are fewer men, so they have a higher ratio.

As it turns out, there have been numerous other studies that have missed the same underlying issue. For example, a couple of years ago, the Boston Globe ran a story on a survey of the study habits of students on Boston area campuses. Their survey showed that on average, minority students tended to study with non-minority students more than the other way around. They went on at great length to explain why this "remarkable

phenomenon" might be true. But it's not remarkable at all — using our graph theory formulation, we can see that all it says is that there are fewer minority students than non-minority students. Well, that just follows from the definition of "minority"!

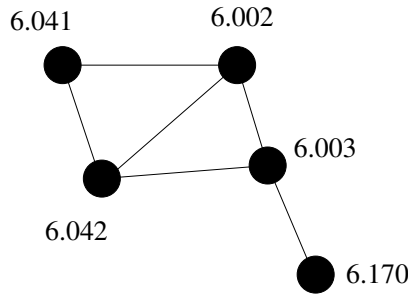
3 Coloring Graphs

In the partner example, we used the notion of an edge to denote an affinity relationship that exists between two nodes. There are many examples in Computer Science where an edge is used to denote just the opposite. Namely, an edge is used to represent a conflict between two nodes. Here is such an example.

Each term, the MIT Schedules Office must assign a time slot for each final exam. This is not easy, because some students are taking several classes with finals, and a student can take only one test during a particular time slot. The Schedules Office wants to avoid all conflicts. Of course, you can make such a schedule by having every exam in a different slot, but then you would need hundreds of slots for the hundreds of courses, and the exam period would run for so long that not only wouldn't there be any time for vacation, but you might never graduate! So, the schedules office would also like to to make the exam period as short as possible.

This is an example of what is a called a *graph coloring problem*: Given a graph G, assign one of k colors to each node such that adjacent nodes have different colors. In general, a graph G is ***k-colorable*** if each vertex can be assigned one of k colors so that adjacent vertices get different colors. The minimum value of k for which a coloring exists is the the ***chromatic number*** $\chi(G)$ of G.

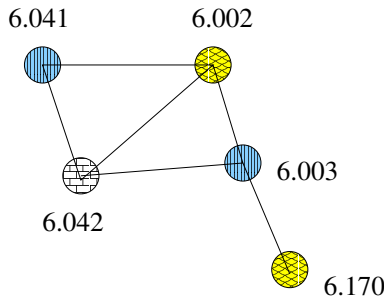
We can recast our scheduling problem as a question about coloring the vertices of a graph. Create a vertex for each course with a final exam. Put an edge between two vertices if some student is taking both courses. For example, suppose we need to schedule exams for 6.041, 6.042, 6.002, 6.003 and 6.170. The scheduling graph might look like this:



6.002 and 6.042 cannot have an exam at the same time since there are students in both courses, so there is an edge between their nodes. On the other hand, 6.042 and 6.170 can have an exam at the same time since no student can be enrolled in both (that is, no student *should* be enrolled in both this semester since they have a timing conflict). Next, identify

each time slot with a color. For example, Monday morning is red, Monday afternoon is blue, Tuesday morning is green, etc.

Assigning an exam to a time slot is now equivalent to coloring the corresponding vertex. The main constraint is that adjacent vertices must get different colors; otherwise, some student has two exams at the same time. Furthermore, in order to keep the exam period short, we should try to color all the vertices using as few different colors as possible. For our example graph, three colors suffice:



This coloring corresponds to giving one final on Monday morning (red), two Monday afternoon (blue), and two Tuesday morning (green).

Can we use fewer than three colors? No! We can't use only two colors since there is a triangle in the graph – so all three vertices participating in the triangle must be colored with different colors.

In general, trying to figure out if you can color a graph with a fixed number of colors can take a long time. It's a classic example of a problem for which no fast optimal algorithms are known. In fact, it is easy to check if a coloring works, but it seems really hard to find it (if you figure out how, then you can get a $1 million Clay prize). However in some cases we can get good upper bounds on the number of colors that are needed.

For example, if k is the maximum degree of any vertex in the graph, then we can easily find a coloring with only $k + 1$ colors.

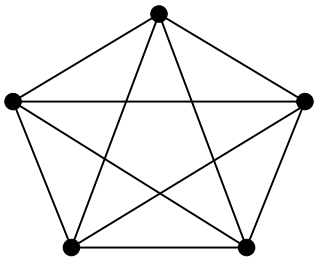
Theorem 1. *A graph with maximum degree at most k is $(k + 1)$-colorable.*

Unfortunately, if you try induction on k, it will lead to disaster. It is not that it is impossible, just that it is extremely painful and would kill you if you tried it on an exam. So, be sure to consider alternatives if you are looking too hard. We already saw that it can help to assume a stronger induction hypothesis. Another option, especially with graphs, is to change what you are inducting on. In graphs, some good choices are n, the number of nodes, or e, the number of edges.

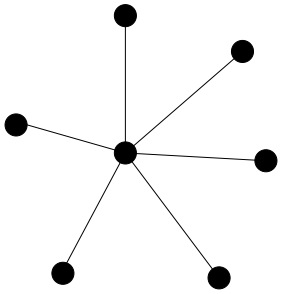
Proof. We use induction on the number of vertices in the graph, which we denote by n. Let $P(n)$ be the proposition that an n-vertex graph with maximum degree at most k is $(k + 1)$-colorable. A 1-vertex graph has maximum degree 0 and is 1-colorable, so $P(1)$ is true.

Now assume that $P(n)$ is true, and let G be an $(n+1)$-vertex graph with maximum degree at most k. Remove a vertex v, leaving an n-vertex graph G'. Note that G' is the subgraph induced on $V - \{v\}$. The maximum degree of G' is at most k, and so G' is $(k+1)$-colorable by our assumption $P(n)$. Now add back vertex v. We can assign v a color different from all adjacent vertices, since v has degree at most k and $k+1$ colors are available. Therefore, G is $(k+1)$-colorable. The theorem follows by induction. □

Sometimes $k+1$ colors is the best you can do. Consider a graph on n nodes with all possible edges, so $d = n-1$. This is called the *complete graph* K_n or a *clique*, just like a clique of friends, where nodes represent the people and an edge represents the friendship relationship.[3]



Sometimes $k+1$ colors is far from the best that you can do. Consider the n-node star, where the node with the maximum degree has degree $n-1$. The star only needs 2 colors!



4 Why coloring?

Coloring problems come up in all sorts of applications. For example, at Akamai, a new version of software is deployed over each of 20,000 servers every few days. The updates cannot be done at the same time since the servers need to be taken down in order to deploy the software. Also, the servers cannot be handled one at a time, since it would take forever to update them all (each one takes about an hour). Moreover, certain pairs of

[3] When speaking of friends, clique is usually pronounced similar to click. However, for some reason, graph theorists think that the word clique rhymes with geek.

servers cannot be taken down at the same time since they have common critical functions. This problem was eventually solved by making a 20,000 node conflict graph and coloring it with 8 colors – so only 8 waves of install are needed!

At a much smaller scale, the same problem exists with register allocation for variables. Each variable needs to be assigned to a register, but you can't have variables that are active at the same time assigned to the same register. So, the number of colors tells us how many registers are needed.

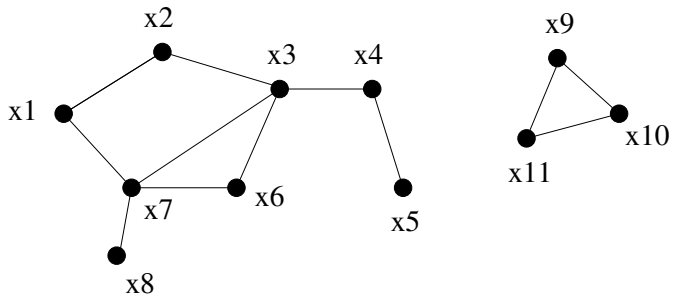
A very famous example of graph coloring is the map coloring problem. In this case, a country or state on a map corresponds to a node and an edge joins two nodes if the corresponding territories on the map share a border. The question is how many colors are needed so that adjacent territories get different colors (if you colored adjacent territories with the same color, how would you be able to tell that they are different territories?). As we mentioned in the first lecture, ultimately, in a very famous result, namely the *4 color theorem*, it was shown that 4 colors suffice. If we have time, in recitation, you'll show the *6 color theorem*.

The last example that we'll mention of a graph coloring application arises in communication graphs. In this problem, we need to assign frequencies to radio stations. If two stations have an overlap in their broadcast area, they can't be given the same frequency. Frequencies are precious and hence an expensive commodity and so you want to minimize the number handed out.

5 Graph theory and communications

We just saw that graph theory comes up in communications problems. In our last example, an edge between to nodes was used to denote a conflict. In the more typical communication problem, an edge between to nodes is used to denote the presence of a communications channel between two end points. The Internet is one eample of such a network, where the nodes correspond to routers (or hubs) and edges correspond to wires or fiber between pairs of routers.

Here is an example of a communications graph:



This network has 11 switches and 12 wires. It doesn't exactly look like the Internet, but it might represent a typical architecture of a single corporate network. The first strange

feature that might pop out at you is that it is not even connected! This could be a little problematic... there is no way for nodes $x_1, \ldots, x_8$ to communicate with nodes x_9, x_{10}, x_{11}. Is this why American productivity isn't increasing at the rate we had hoped for lately? Well, actually, this might be by design. Today some military networks are not allowed to connect to the Internet or other government networks, for fear of being attacked by viruses and worms that might spread through connections. We are able to formalize the notion of being able to communicate in the graph by the notion of a *path*.

A *path* in a graph $G = (V, E)$ is a sequence of distinct nodes $x_1, x_2, \ldots, x_k$ such that each of x_1—x_2, x_2—$x_3, \ldots x_{k-1}$—$x_k \in E$. For example, x_8, x_7, x_6, x_3, x_4 is a path from x_8 to x_4.

For this class, we'll assume that all the nodes in a path are distinct, but not all texts require this. We'll use the term *walk* to refer to paths with repeated nodes – i.e., a walk is the same as a path except that you can repeat nodes and edges as you move through the graph. So any path is a walk. But $x_8, x_7, x_6, x_3, x_7, x_6$ is a walk which is not a path.

We then say that a graph $G = (V, E)$ is *connected* if for all pairs of nodes $x_i, x_j \in V$, there is a path from x_i to x_j in G.

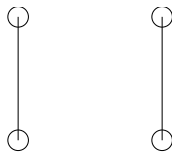
The graph shown here is not connected since, for example, there is no path from x_5 to x_9. But it does have *two* connected components – that is, it consists of two subgraphs that are themselves connected.

A *connected component* of G is a maximal connected subgraph of G. Maximal means that you can't add any nodes or edges to the subgraph without making it be disconnected. For example, x_9, x_{10}, x_{11} is a connected component above. But, $x_1, \ldots, x_7$ is not, since you can add x_8 and still stay connected.

Here is a false proof about connectivity. It exposes a very common flaw made on proofs by induction on graphs – it even has a name – it is known as "build-up error".

False Claim. *If every vertex in a graph has degee at least 1, then the graph is connected.*

There are many counterexamples; here is one:



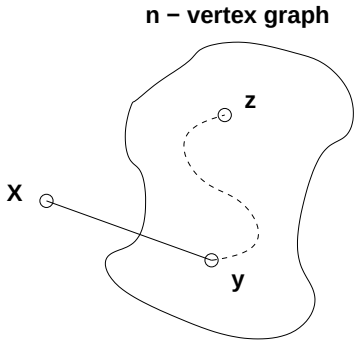
Since the claim is false, there must be at least one error in the following "proof".

Proof. We use induction. Let $P(n)$ be the proposition that if every vertex in an n-vertex graph has degree at least 1, then the graph is connected.

Base case: There is only one graph with a single vertex and it has degree 0. Therefore, $P(1)$ is vacuously true, since the if-part is false.

Inductive step: We must show that $P(n)$ implies $P(n+1)$ for all $n \geq 1$. Consider an n-vertex graph in which every vertex has degree at least 1. By the assumption $P(n)$, this graph is

connected; that is, there is a path between every pair of vertices. Now we add one more vertex x to obtain an $(n+1)$-vertex graph:



All that remains is to check that there is a path from x to every other vertex z. Since x has degree at least one, there is an edge from x to some other vertex; call it y. Thus, we can obtain a path from x to z by adjoining the edge x—y to the path from y to z. This proves $P(n+1)$.

By the principle of induction $P(n)$ is true for all $n \geq 1$, which proves the theorem. □

That looks fine! Where is the bug? It turns out that faulty assumption underlying this argument is that *every $(n+1)$-vertex graph with minimum degree 1 can be obtained from an n-vertex graph with minimum degree 1 by adding 1 more vertex*. Instead of starting by considering an arbitrary $(n+1)$- node graph, this proof only considered an $(n+1)$-node graph that you can make by starting with an n-node graph with minimum degree 1.

The counterexample above shows that this assumption is false; there is no way to build that 4-vertex graph from a 3-vertex graph with minimum degree 1. Thus the first error in the proof is the statement "This proves $P(n+1)$".

More generally, this is an example of "build-up error". Generally, this arises from a faulty assumption that every size $n+1$ graph with some property can be "built up" from a size n graph with the same property. (This assumption is correct for some properties, but incorrect for others— such as the one in the argument above.)

One way to avoid an accidental build-up error is to use a "shrink down, grow back" process in the inductive step: start with a size $n+1$ graph, remove a vertex (or edge), apply the inductive hypothesis $P(n)$ to the smaller graph, and then add back the vertex (or edge) and argue that $P(n+1)$ holds. Let's see what would have happened if we'd tried to prove the claim above by this method:

Inductive step: We must show that $P(n)$ implies $P(n+1)$ for all $n \geq 1$. Consider an $(n+1)$-vertex graph G in which every vertex has degree at least 1. Remove an arbitrary vertex v, leaving an n-vertex graph G' in which every vertex has degree... uh-oh!

The reduced graph G' might contain a vertex of degree 0, making the inductive hypothesis $P(n)$ inapplicable! We are stuck— and properly so, since the claim is false!