Math 3322: Graph Theory[1] Mikhail Lavrov

Lecture 1: What are graphs?

August 17, 2021 *Kennesaw State University*

1 Useful information about the class

See the course webpage or D2L for information about when things happen, how grading works, and so forth.

- The first homework assignment is due Friday, August 27th by 11:59pm, on D2L.

  The first exam will be given in class on Tuesday, September 28th.

- Homework grades and solutions will be posted on D2L.

  Notes for each lecture will be posted on the course webpage.

- Office hours are Tuesday and Thursday after class (2pm–3pm). You can also reach me by email, or by making an appointment for a different time.

- The official textbook for this class is *A First Course in Graph Theory* by Chartrand and Zhang. It is a useful resource for the topics we'll cover, but you will not strictly speaking need it: I will not assign problems out of the textbook. (I do recommend working on problems from the textbook if extra practice with the concepts would help you.)

2 Examples of graphs

2.1 Tour of the US

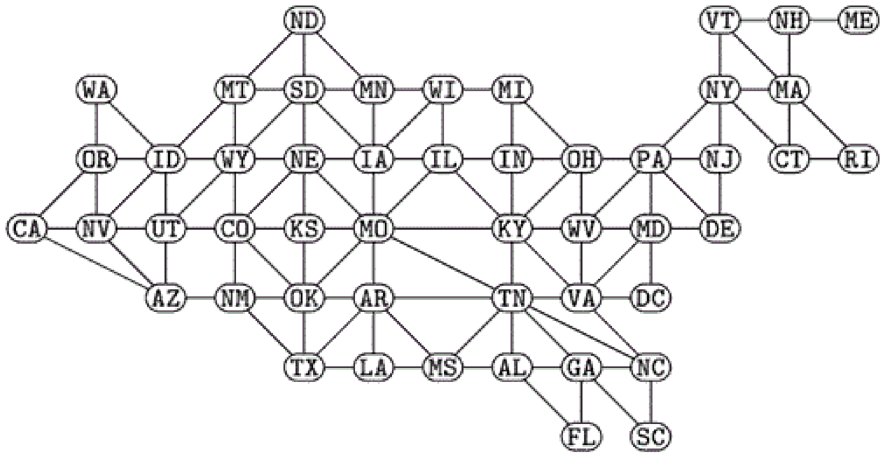
Suppose that you decide to take a tour of the 48 contiguous US states, by car. To make things extra challenging for yourself, you add a condition: you cannot visit any state more than once. We can ask several questions:

- Can you visit all 48 states?
- Can you visit all 48 states, starting from Georgia?
- Can you visit all 48 states, and end adjacent to where you started?

We will not answer any of these questions today (though you can think about them on your own, if you like). Instead, we will talk about graphs: the setting where we can ask this kind of question.

[1] This document comes from the Math 3322 course webpage: http://facultyweb.kennesaw.edu/mlavrov/courses/3322-fall-2021.php

If you want to solve this problem, you do not need all the information on a map of the US. It might be equally convenient (or even more convenient) to represent the US by the following diagram:[2]



(Here, a line is drawn between any two states that are directly connected by at least one drivable road.)

Most of the features of the diagram are irrelevant: it does not actualy matter that Maine is drawn in the top right and California is the leftmost state, and it does not mean anything that the line between Tennessee and Montana is longer than most other lines. We could encode all the information we *need* to solve the problem as follows:

1. Make a list of all the states we want to visit: AL, AZ, AR, . . . , WY.
2. Make a list of all pairs of states with a road between them, such as $\{\mathrm{AL}, \mathrm{FL}\}$ or $\{\mathrm{MO}, \mathrm{TN}\}$.

This is exactly what a graph is!

Formally, a **graph** G is a pair (V, E) where

- V is a set of objects called **vertices**. (These can be anything.)
- E is a set of **edges**; each edge is a pair $\{v, w\}$ of vertices $v, w \in V$ and tells us that v and w are **adjacent**.

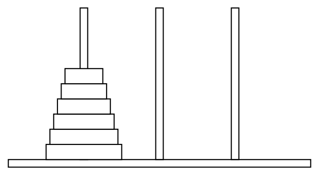
We will often denote the edge between v and w as vw instead of $\{v, w\}$. This is unordered: vw and wv are the same edge.

It is often convenient to represent a graph by a diagram like the one on the previous page, where the edges are drawn as lines connecting the vertices. Sometimes, when we don't need to keep track of the names of the vertices, we'll draw the vertices as simple dots.

[2] Weisstein, Eric W. "Contiguous USA Graph." From MathWorld–A Wolfram Web Resource. https://mathworld.wolfram.com/ContiguousUSAGraph.html

2.2 The Towers of Hanoi puzzle

In the Towers of Hanoi puzzle, you have three pegs, and some number of disks of different sizes stacked on the pegs. Initially, all the disks are placed on one peg, sorted by size (with the smallest disk on top):



You are allowed to make the following moves in this puzzle: lift the top disk on a peg, and put it down on another peg. However, you cannot place a larger disk on top of a smaller one.

The goal of the puzzle is to move all the disks from one peg to another.

To model this puzzle as a graph $G = (V, E)$, we can do the following:

- Let V be the set of all possible states of the puzzle.
- Put an edge between two vertices (two states of the puzzle) if it's possible to get from one to the other by a single move. (This is a symmetric relation: if we can move a disk from one peg to another, we can always move it back.)

As a problem, the Towers of Hanoi puzzle looked very different from our driving-around-the-US question. However, as graphs, we are asking some similar questions: in both cases, we are "moving around" the vertices of the graph, traveling along the edges.

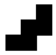
2.3 Circuit board layouts

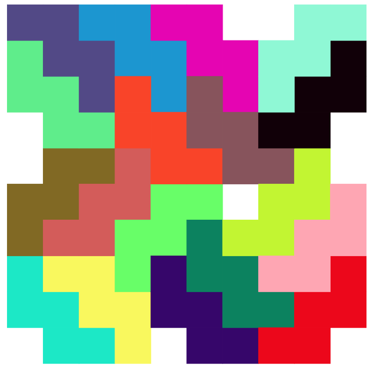
A circuit board is a plastic sheet with many components on it, some of which are connected by conductive copper tracks. When we are initially designing a circuit, we keep track of what the components are, and which ones should be connected by wires. This is a graph: the components are the vertices, and the wires are the edges.

When we print the circuit board, we have an additional constraint: we don't want the conductor tracks to cross!

A graph does not keep track of layout: it is the same no matter how you represent it as a diagram. However, graph theory does study the question: does a graph *have* a diagram in which no two edges cross? This is the problem of finding a **plane embedding** of a graph.

2.4 Placing zigzag tiles

You have an infinite supply of  shaped tiles. How many of them can you place on a 10×10 grid without overlap? One very good solution (which I suspect to be optimal) is given below:



(If you are printing this page in black and white, you will probably have a hard time looking at this diagram—sorry!)

How did I find this solution? I did it by encoding the problem as a graph and using some tools in Mathematica's graph theory library. It's not obvious how to represent this as a graph theory problem, but here is what I ended up doing:

- Let the vertices of be all the ways to place a *single*  tile on the 10×10 grid.
- Put an edge between two vertices if the tile placements they represent are incompatible: the tiles would overlap.

A solution to the puzzle is a set of vertices. Edges represent conflicts, so a conflict-free solution is a set of vertices with no edges between them. This is called an **independent set** in a graph, and we will look at the problem of finding these later on in the class.

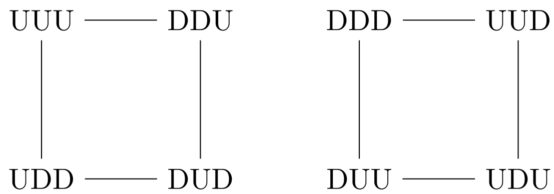
There are 256 vertices: the center of the tile can go anywhere in an 8×8 square of the grid, because it cannot lie along an edge, and there are 4 ways to rotate the tile, once you know where its center goes. This is still a lot for a human, but not out of the reach of computer algorithms.

3 Walks and connectedness

For this problem, I want to introduce one final problem we can model with graphs: the three cups puzzle. It is similar to the Towers of Hanoi puzzle in how we will model it, but it is much simpler.

In the three cups puzzle, you have three cups lined up in a row. In one move, you are allowed to take two consecutive cups (the first and second cup, or the second and third) and flip both of them over. If a cup was already upside down, flipping it will make it right side up again.

The goal of the puzzle is to flip all three cups upside down. But after drawing the diagram of all 8 states, and the adjacencies between them, we quickly see that there's no solution:



(I am using U and D to denote a cup facing Up and a cup facing Down, respectively.)

There are no edges leading from the left part of the diagram to the right part. UUU is on the left, and DDD is on the right, and so we cannot get from UUU to DDD.

We see the same thing show up in other problems. For example, suppose we return to the car-driving problem, but this time we want to take a world tour by car. We run into an obstacle: cars cannot drive on water. So our graph will, once again, have several pieces corresponding to different landmasses. We can get from one vertex (location) to another if they are on the same landmass, but we have no hope if they are on different landmasses.

Our goal for the next lecture is to formalize this idea, which will give us the notions of **connected graphs** and **connected components** of graphs. For now, we want to begin by defining formally what it means to "get from one vertex to another".

We say that a $v - w$ **walk** in a graph G is a sequence of vertices

$$v_0, v_1, v_2, \ldots, v_\ell$$

where $v_0 = v$, $v_\ell = w$, and for $i = 0, 1, \ldots, \ell - 1$, $v_i v_{i+1}$ is an edge. For example, in the cups puzzle, the sequence UUU, DDU, DUD is a UUU − DUD walk.

The number ℓ is the **length** of the walk. (The length is the number of edges used; one fewer than the number of vertices.) The walk UUU, DDU, DUD has length 2, which matches our intuition that it takes 2 steps to get from UUU to DUD.

A special kind of walk is a $v - w$ **path**. A path is a walk in which no vertices are repeated. The above walk is also a path, but UUU, DDU, DUD, UDD, DUD is a UUU − DUD walk which is not a path: the vertex DUD occurs twice.

Initially, we will be more interested in walks, rather than paths. Looking at paths will be interesting to us later on, when we want to say things like "there are two ways to get from UUU to DUD". Indeed, there are only two UUU − DUD paths in the three cup graph; however, there are infinitely many UUU − DUD walks, because we can take any number of redundant steps.