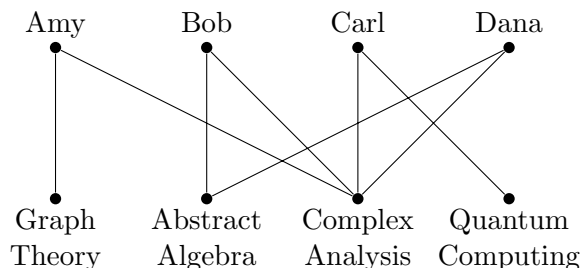# 1   Bipartite graphs

As motivation, consider the following problem. You're in charge of the math department, and so you have to pick who teaches which classes. You start with the following information:

- A list of all the classes that must be offered.

- A list of all the possible instructors that can teach them.

- Each instructor has a list of things they're willing to teach, and a list of thing they're not willing to teach.

In the simplest version of the scheduling problem, each class is taught exactly once, and each person teaches exactly one class. (This is called the **matching problem**; we will return to it later in the semester.)

We can model this data as a graph: make all the classes and all the instructors vertices, and put an edge from each instructor to the classes they're willing to teach. Something like this:

This is an example of a bipartite graph. A **bipartite graph** is a graph G whose vertex set $V(G)$ can be split into two parts A and B, such that every edge has one endpoint in A and one endpoint in B.

Bipartite graphs show up in graph theory for two reasons:

1. Sometimes, our data is inherently "bipartite". In the example above, we can take the bipartition $A = \{\text{Amy}, \text{Bob}, \text{Carl}, \text{Dana}\}$ and

$$B = \{\text{Graph Theory}, \text{Abstract Algebra}, \text{Complex analysis}, \text{Quantum Computing}\}$$

and we know all edges go between A and B because of how the graph is defined.

---

[1]This document comes from an archive of the Math 3322 course webpage: http://misha.fish/archive/3322-fall-2024

2. Sometimes, a graph will turn out to be bipartite because of some hidden property of the graph that we didn't see coming ahead of time. Often, this helps us understand the graph better once we do discover that it's bipartite.

How can we discover that a graph is bipartite? Here is an algorithm we can use. It will try to build a bipartition, and either it will succeed, or it will fail and we will know the graph is not bipartite.

1. Pick any vertex v. Arbitrarily put v on side A of the bipartition: this is fine, because the two sides are symmetric, so we can't make the wrong choice here.

2. Put every neighbor of v on side B of the bipartition. This is the only choice we can make: if any neighbor of v were on side A, we would have an edge between two vertices of A.

3. For each vertex we just put on side B, take their neighbors, and put those on side A. Again, this is the only choice we can make to avoid having an edge between two vertices of B.

4. Repeat these steps: when we assign a vertex to one side, assign all its neighbors to the other side.

   What if some of its neighbors have already been assigned to a side? Well, if they were assigned to the side we were going to put them on anyway, there's no problem. But if we discover that a vertex has a neighbor that's already been assigned to the *same* side, then we're not getting a valid bipartition!

   If this ever happens, we know the graph is not bipartite. At each step, we were making the only choice we could, and we still got stuck.

5. If the graph is connected, we'll eventually assign all vertices to a side in this way. If the graph is not connected, repeat for every connected component! Like many other problems, this one can be solved separately for different connected components.

This algorithm might look similar to our method of finding all distances from v. In fact, if we compare what the two algorithms do, we can realize that this algorithm puts w on side A if $d(v, w)$ is even, and on side B if $d(v, w)$ is odd.

That's the story from the point of view of algorithms. From the point of view of theorems and proofs, there's a useful characterization of bipartite graphs: the theorem below.

Theorem 1.1. *The following are equivalent for a graph G:*

1. *G is bipartite.*

2. *G has no closed walks of odd length.*

3. *G has no cycles of odd length.*

Before we go on to prove this theorem, let's ask: why is it useful? Because it always gives us something to use about G.

If we're assuming that a graph is bipartite, or proving that the graph is bipartite, the definition is useful: it means we can assume that a bipartition exists, or try to construct one.

If we're assuming that a graph is not bipartite, it's more useful to use condition 3 of the theorem, and assume that G has an odd cycle. This gives us a concrete fact about G to start from.

Finally, if we're proving that a graph is not bipartite, it's more useful to use condition 2, and try to construct a closed walk of odd length. (That's marginally easier than trying to construct a cycle; we don't have to worry about repeated vertices.)

Proof of Theorem 1.1. Suppose G has a bipartition (A, B), but also a closed walk $(v_0, v_1, \ldots, v_{2k+1})$ with $v_0 = v_{2k+1}$. Let's say $v_0 \in A$; this assumption is **without loss of generality** because the proof would be identical if we put $v_0 \in B$.[2]

Then $v_1 \in B$, or else the edge $v_0 v_1$ would have both endpoints in A. By the same logic, $v_2 \in A$, $v_3 \in B$, and so on. In a way that formally ought to be done by induction, we conclude: $v_i \in A$ if i is even and $v_i \in B$ when i is odd. Therefore $v_{2k+1} \in B$; but $v_{2k+1} = v_0$, and we already decided that $v_0 \in A$. This is a contradiction! Therefore G cannot have both a bipartition and a closed odd walk: condition 1 implies condition 2.

Suppose G is not bipartite. Then if we follow our bipartition algorithm starting from some vertex v, we'll eventually fail: both endpoints of an edge xy will both be put on the same side of the bipartition. By our observation about distances, this means that $d(v, x)$ and $d(v, y)$ are both even or both odd.

Consider the following closed walk: follow a shortest path from v to x, then take the edge xy, then follow the reverse of a shortest path from v to y. This has total length $d(v, x) + 1 + d(v, y)$, which is an odd number. So if G is not bipartite, it has a closed walk of odd length. This means that condition 2 implies condition 1: if G has no closed walks of odd length, it is bipartite.

Condition 2 implies condition 3, because all cycles are closed walks: condition 2 is just a more general statement. So we'll be done with the proof if we show that condition 3 implies condition 2. To do this, suppose G has a closed walk of odd length; we'll show that G also has a cycle of odd length.

Use the extremal principle, and pick a shortest closed walk of odd length: $(v_0, v_1, v_2, \ldots, v_{2k+1})$. That length is not 1: (v_0, v_1) is never a closed walk in our graphs, since a vertex is never adjacent to itself.

So the length is at least 3, and either it is a cycle, or else we have $v_i = v_j$ for $0 \le i < j < 2k + 1$. This gives us two shorter closed walks:

- Closed walk $(v_i, v_{i+1}, \ldots, v_j)$, with length $j - i$.
- Closed walk $(v_0, v_1, \ldots, v_i, v_{j+1}, \ldots, v_{2k+1})$, with length $2k + 1 - (j - i)$.

These two lengths add up to $2k+1$, which is odd. So at least one of the lengths is odd—and shorter than $2k + 1$. This contradicts our assumption that our closed walk of odd length was as short as possible. So the closed walk we took must already be a cycle, completing our proof. $\square$

---

[2] This logic is very common. Often, we'll just say a shorter phrase: "Without loss of generality, $v_0 \in A$."

2 Graph zoo

The previous section was full of new ideas and long proofs; here, we'll just look at some graphs that have names. These graphs show up over and over again, so they're useful to know.

2.1 The essential graphs

The first is the **path graph** P_n. It has n vertices $v_1, v_2, \ldots, v_n$ and $n-1$ edges: $v_i v_{i+1}$ for $i = 1, 2, \ldots, n-1$. Here is a diagram:


Among other reasons, the path graph is useful to know because it gives us a way to think about paths. We've been saying that a path is a special kind of walk: a sequence of vertices. But we can also think of a path as a subgraph: a subgraph that is isomorphic[3] to P_n for some n.

Next is the **cycle graph** C_n. (Here, we require $n \geq 3$ for the definition to work.) It has n vertices $v_1, v_2, \ldots, v_n$ and n edges: the edges of P_n, plus the edge $v_1 v_n$. Here is a diagram:
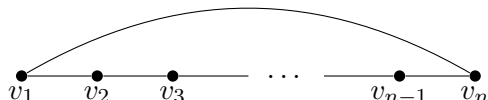

This has a similar relationship to cycles: we can think of a cycle in a graph G not just as a special kind of closed walk, but as a subgraph isomorphic to C_n for some n.

The **complete graph** or **clique** K_n is the graph with the most edges for its vertex set. It has n vertices $v_1, v_2, \ldots, v_n$ and all possible edges $v_i v_j$ with $i \neq j$. There are $\binom{n}{2} = \frac{n(n-1)}{2}$ edges: that's how many ways there are to choose two endpoints. Here are diagrams of K_3, K_4, K_5, and K_6:
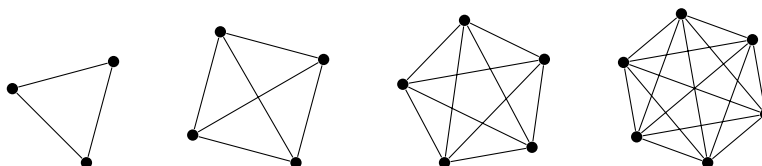
The **complete bipartite graph** $K_{m,n}$ is a bipartite graph with as many edges as it could possibly have. It has vertex set $\{v_1, v_2, \ldots, v_m, w_1, w_2, \ldots, w_n\}$; its bipartition will be $A = \{v_1, v_2, \ldots, v_m\}$ and $B = \{w_1, w_2, \ldots, w_n\}$. The only allowed edges have the form $v_i w_j$, and $K_{m,n}$ has all of these edges.

Here are some examples. From left to right, these are $K_{1,4}$, $K_{3,3}$, and $K_{5,2}$:


---
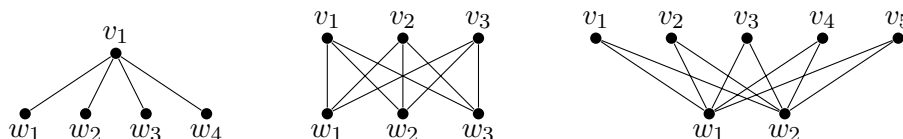
[3]We'll talk about isomorphisms formally later on in the semester. For now, think of "is isomorphic to" as a fancy way to say "looks like".

2.2 More graphs it is useful to know

The graphs in the previous section are graphs every graph theorist knows. Here are a couple more that are less foundational, but which we'll often use in examples.

The **cube graph** Q_3 has vertices $\{(0,0,0),(0,0,1),(0,1,0),(0,1,1),(1,0,0),(1,0,1),(1,1,0),(1,1,1)\}$ and an edge between any two vertices that differ in one coordinate. It looks like a cube when we draw a diagram. It generalizes to Q_d: a graph with 2^d vertices defined by the same rule, but with vertex set $\{0,1\}^d$. Q_3 is shown below on the left:
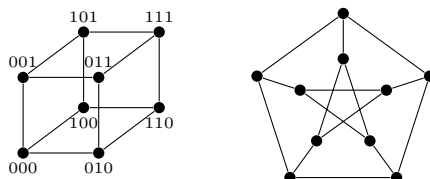
The **Petersen graph**, which we've already seen, can be defined by the following rule. Its vertices are pairs of elements of $\{1,2,3,4,5\}$: they are $\{1,2\},\{1,3\},\ldots,\{4,5\}$. Two vertices $\{a,b\}$ and $\{c,d\}$ are adjacent if they have no elements in common: the intersection $\{a,b\}\cap\{c,d\}$ is empty. (See also Exercise 4.)

2.3 Operations on graphs

The simplest operation on graphs is the **complement** of a graph. If G is any graph, then its complement $\overline{G}$ is a graph with all the same vertices, but exactly the edges that G does not have. For any two vertices u and v, either uv is an edge of G, or it is an edge of $\overline{G}$, but not both. For example, here are diagrams of $K_{3,2}$ and of its complement $\overline{K_{3,2}}$:
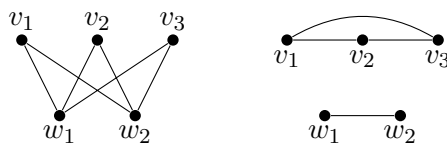
When we take the **union** $G \cup H$ of two graphs, we must be careful. In full generality, $G \cup H$ has all vertices of G and H, and all the edges of both. The most useful versions of this are:

- When G and H have the same vertices, their union just combines the edges: it is the overlap of G and H. We often want to know when a graph is the union of simpler graphs in this way. For example, can you represent K_8 as the union of three bipartite graphs?

- When G and H have no vertices in common, and $G \cup H$ just has all of G and all of H inside it, with no edges between them. This is a common way to denote connected components.

  Sometimes people write $\cup$ to mean a "disjoint union" in which the vertices are relabeled to be different. For example, we might write the graph $\overline{K_{3,2}}$ as $K_3 \cup K_2$ to represent its connected components, even though formally $K_3 \cup K_2$ would just have three vertices.

3   Practice problems

1. Among the small named graphs $P_2, P_3, K_2, K_3, C_3, C_4, K_{1,1}, K_{1,2}, K_{2,1}, K_{2,2}, Q_2$ there are a few "overlaps": two (or more) names for essentially the same graph. (In some cases, with different vertex names, but you can give them the same diagram with different labels.)

   Find all the overlaps.

2. (a) Suppose we take the disjoint union of two copies of K_n; let's say that one of them has vertex set $\{v_1, v_2, \ldots, v_n\}$ as usual, but the other one has vertex set $\{w_1, w_2, \ldots, w_n\}$ instead. Then, we take the complement.

   What is the graph we get?

   (b) Suppose we take the disjoint union of three complete graphs K_a, K_b, and K_c, then take the complement.

   Describe the graph we get directly (by saying what its vertices and edges are). What would be a good name for this graph?

3. The cube graph Q_3 (and, in general, Q_d for any d) is bipartite. What is the bipartition?

   (If you're ambitious, try it for all d; either way, you should start with Q_3 or even Q_2.)

4. Our definition of the Petersen graph says: its vertices are pairs of elements of $\{1, 2, 3, 4, 5\}$: they are $\{1, 2\}, \{1, 3\}, \ldots, \{4, 5\}$. Two vertices $\{a, b\}$ and $\{c, d\}$ are adjacent if they have no elements in common: the intersection $\{a, b\} \cap \{c, d\}$ is empty.

   Below are two diagrams of the Petersen graph (I claim). Prove it! That is, show a way to label the vertices of the graph in each diagram with pairs $\{a, b\}$ such that adjacent vertices are labeled with pairs that have no elements in common.
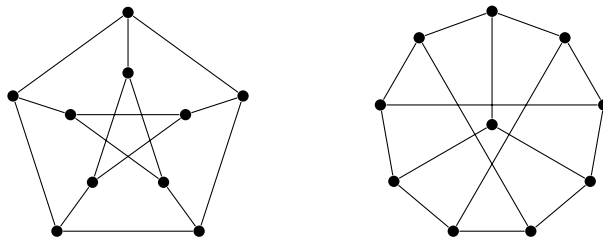
5. Earlier in class, we proved that whenever G has an odd closed walk, it also has an odd cycle.

   The same thing doesn't work if "odd" is replaced by "even". Why not?

   (a) Explain where in the proof the argument fails for even closed walks.

   (b) Give an example of an even closed walk in a graph with no even cycles.

   (Side note: these are the two ways you can disagree with a mathematical argument, and they both have their place. The first one is saying, "Here, you made a mistake. Your claim may still be true, but you haven't given a valid proof." The second one is saying, "I don't see how your proof is wrong, but it must be; here is a counterexample to the claim." The best thing to do, of course, is to give both, if you can!)

6. Prove that if G is a bipartite graph, and H is a subgraph of G, then H is also bipartite.

7. Prove that if G is a bipartite graph on $n \geq 5$ vertices, then the complement of G is not bipartite.

8. For which bipartite graphs G is it true that the complement of G is connected? Prove your answer.