Math 3322: Graph Theory[1] Mikhail Lavrov

Lecture 13: Bipartite matchings

October 1, 2024 *Kennesaw State University*

1 The bipartite matching problem

1.1 The setup

Imagine that you're in charge of scheduling classes for the KSU math department. You have 100 sections to schedule, and if you add up the number of classes each professor is willing to teach, that also gets you to 100, so everything seems fine.

Unfortunately, there are other constraints. I would be fine teaching graph theory, but bad at teaching differential equations. My friend in the office next door would be fine teaching differential equations, but bad at teaching graph theory. Neither of us would want to teach a class in a big lecture hall on the Kennesaw campus, because our offices are here on the Marietta campus.

You put all this together into sets of possibilities: maybe "Misha's first class" could be any of

$$\{\text{graph theory}, \text{section 1 of probability}, \text{section 2 of probability}, \text{intro to proofs}, \dots\},$$

and the same for "Misha's second class", and so on for all 100 classes that will be taught. Now your goal is to pick an item from each set so that you end up picking every section that needs to be taught.

1.2 Course scheduling, as a graph

The graph representation of this data is a bipartite graph where the people and their time slots (Misha's first class, Misha's second class, ...) are vertices on one side, while the sections (graph theory, section 1 of probability, ...) are vertices on the other side. There is an edge between "Misha's first class", and every section that Misha's first class could be.

A solution to the class scheduling problem is a subset M of edges: if you say that the first class I'm teaching is graph theory, then that corresponds to putting the edge $\{\text{Misha's first class}, \text{graph theory}\}$ in M. There are two constraints on what M can be:

- Two edges we put in M can't share the same vertex on the first side: "Misha's first class" can only be one thing.
- Two edges we put in M can't share the same vertex on the second side: "graph theory" is only taught once. (If we want multiple sections of a class, we have already represented that by creating multiple vertices for the different sections.)

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

1.3 The matching problem

In general, a **matching** in a graph G is a set of edges M such that no two edges in M have the same endpoint. If you like, you can also think of this as a subgraph of G in which no vertex has degree more than 1, but we'll stick to the set-of-edges version for the most part; assume that's what these notes mean unless they say otherwise.

We can think about matchings in all kinds of graphs. But we'll start with thinking about matchings in bipartite graphs for two reasons. First, lots of applications (like the course scheduling application) will naturally give us a bipartite graph to look for a matching in. Second, the theory turns out to be simpler for bipartite graphs.

We say that a vertex of G is **covered** by M if it's the endpoint of some edge of M, and **uncovered**[2] otherwise. (In the subgraph interpretation: a vertex is covered by M if it has degree 1 in M, and uncovered if it has degree 0.)

At this point, we can be optimistic or realistic. An optimist asks "is the cup full?" but a realist asks "how much water is in the cup?" What am I talking about? Well, there's two questions you can ask about matchings in a graph:

1. You can ask "Is there a perfect matching?" A matching is **perfect** if it covers every single vertex.
2. You can ask "How many edges are in a maximum matching?" A **maximum** matching is one with as many edges as possible.

The answer to the first question is often the important one in theoretical applications, and is actually one of the main results from graph theory that shows up in other areas of math. However, we will start with the second question, because an answer to the second question will also answer the first. A bipartite graph with n vertices on one side and n vertices on the other side has a perfect matching if and only if the maximum matching(s) in that graph have n edges.

1.4 The maximum/maximal distinction

In combinatorial optimization problems like the matching problem, we are trying to find a set S that's as large as possible, subject to some condition. It's common to say that a set S is:

- **maximum** if it really is as large as you can get; there are no better solutions than S.
- **maximal** if you cannot add anything to S without violating the restriction; there might be better solutions, but you cannot get to them without removing something from S first.

Every maximum solution is also maximal, but the reverse might not hold.

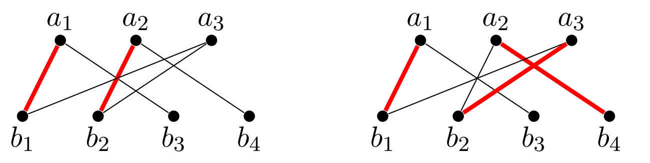
In the same way, "minimum" and "minimal" get used when we're trying to pick a set that's as small as possible, subject to some condition.

In these lecture notes, I will avoid casually using the terms "maximal" and "minimal" to draw this distinction, because the difference is very subtle and easy to miss. I will specifically point out when we're dealing with maximal-but-not-maximum objects. However, no matter which terminology you

[2] You will see "saturated" and "unsaturated", or "matched" and "unmatched", in other sources.

use, it's important to realize that there **is** a difference, and in particular there is a difference in the matching problem.

Take the following two matchings in the same graph:



$M_1 = \{a_1b_1, a_2b_2\}$ $\qquad$ $M_2 = \{a_1b_1, a_2b_4, a_3b_2\}$

If we go through vertices a_1, a_2, a_3 in order, and pick the first neighbor in b_1, b_2, b_3, b_4 to match each of them to, we'll get matching $M_1 = \{a_1b_1, a_2b_2\}$. Vertices a_3, b_3, b_4 are uncovered, but don't have uncovered neighbors; we can't add any edges to M_1.

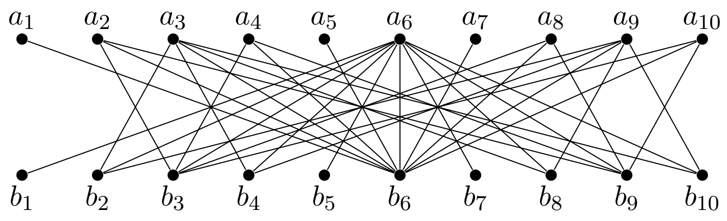
However, if we remove edge a_2b_2 from M_1, then instead we can add edges a_2b_4 and a_3b_2, getting the larger matching M_2. In fact, this is one of the possible maximum matchings in the graph.

This tells us that the bipartite matching problem is not straightforward. By contrast, consider the problem of finding a spanning tree in a graph. There, if you try to find the smallest set of edges that connect all the vertices, you'll never get stuck in a suboptimal solution; you'll always end up at $n - 1$ edges for n vertices. That was a much easier problem.

2 Vertex covers

2.1 The multiples-of-six graph

Consider the bipartite graph with vertices $a_1, a_2, \ldots, a_{10}$ on one side, vertices $b_1, b_2, \ldots, b_{10}$ on the other side, and edges defined by the following rule: a_i and b_j are adjacent exactly when the product ij is divisible by 6. Here is a diagram:



What is the largest matching in this graph?

You will not have too much trouble finding a matching with 6 edges. For example, you can take the matching made by three "X" shapes in the graph: $M = \{a_2b_3, a_3b_2, a_5b_6, a_6b_5, a_8b_9, a_9b_8\}$. It seems difficult to find a better matching, but how can we know that it's impossible?

If we try to find other matchings, we encounter a bottleneck. To get a product divisible by 6, we need a multiple of 2 and a multiple of 3, but the multiples of 3 are much harder to come by. Every single edge must include one of the multiples of 3: a_3, a_6, a_9, b_3, b_6, or b_9. However, we can't use any of these vertices twice in a matching. This means we can have at most 6 edges in a matching: one for each of these vertices.

2.2 Vertex covers

This argument generalizes.

Let a **vertex cover** of G be a set of vertices U such that every edge of G has at least one endpoint in U (possibly both). For example, in the graph above, $\{a_3, a_6, a_9, b_3, b_6, b_9\}$ is a vertex cover. It is not the only one:

- The set of *all* the vertices in a graph is always a vertex cover.
- In a bipartite graph like the one above, the set of all vertices on one side is a vertex cover: for example, $\{a_1, a_2, a_3, \dots, a_{10}\}$ is a vertex cover.
- Every edge needs a multiple of 2, so $\{a_2, a_4, a_6, a_8, a_{10}, b_2, b_4, b_6, b_8, b_{10}\}$ is a vertex cover.

However, $U = \{a_3, a_6, a_9, b_3, b_6, b_9\}$ is the smallest vertex cover.

Vertex covers are important because of the following claim:

Claim 2.1. *If M is a matching and U is a vertex cover in the same graph, then $|M| \le |U|$.*

Proof. In this proof, we will think of M as a subgraph of G: it will have all the vertices of G, but only the edges that are edges of M. Consider the sum

$$\sum_{u \in U} \deg_M(u).$$

Each term $\deg_M(u)$ of this sum is at most 1, because no vertex can have degree more than 1 in M: that's what being a matching means. So the value of sum is at most $|U|$.

Each edge of M has at least one endpoint in U, because that's what being a vertex cover means, contributing at least 1 to $\deg_M$ of that endpoint. So the value of the sum is at least $|M|$.

Therefore $|M| \le |U|$. □

Therefore vertex covers are a way to get upper bounds on the maximum size of a matching in a graph. For example, a simple observation: if G is a bipartite graph with bipartition (A, B), then the maximum number of edges in a matching is at most $\min\{|A|, |B|\}$. That's because both A and B are vertex covers.

We can try to find a **minimum vertex cover** (one with as few vertices as possible) to get the best upper bound. However, this upper bound could still be really bad. For example, the largest matching in the complete graph K_{100} has only 50 edges: at that point, you've used up all the vertices! However, the smallest vertex cover in K_{100} has 99 vertices: you have to use all but one of the vertices, because if you skip two vertices, you don't cover the edge between them.

It will turn out that in bipartite graphs, the minimum vertex cover *is* equal in size to the maximum matching. This is known as König's theorem; we will prove it in the next lecture. It is the reason we are studying matchings in bipartite graphs first.

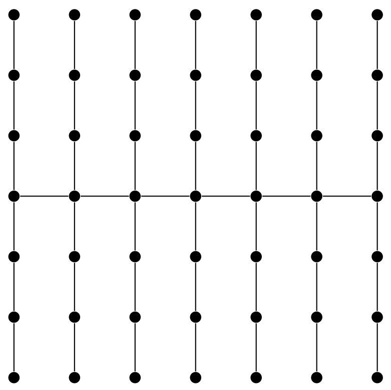
3 Practice problems

1. Generalize the multiples-of-six graph to a bipartite graph with $2n$ vertices: $a_1, a_2, \dots, a_n$ on one side and $b_1, b_2, \dots, b_n$ on the other side.

   What is the largest matching, and the vertex cover that proves that it is the largest?

2. If we want to modify the multiples-of-six graph (the un-generalized one, with 20 vertices) so that $\{a_3, a_6, a_9, b_3, b_6, b_9\}$ is *still* a vertex cover, but there are as many edges as possible, what is the best thing to do?

3. In the graph below, find a matching M and a vertex cover U with $|M| = |U|$.



4. An **independent set** in a graph G is a set $I \subseteq V(G)$ such that there is no edge between any two vertices in I. We will learn more about independent sets later.

   For now, prove the following: a set $U \subseteq V(G)$ is a vertex cover if and only if its complement $V(G) - U$ is an independent set.

5. A **dominating set** in a graph G is a set $D \subseteq V(G)$ such that every vertex not in D has a neighbor in D. These are not as important as independent sets or vertex covers; they appear in some applications, but we will not study them this semester.

   Instead, answer the following:

   (a) Is a vertex cover always a dominating set? Prove this, or find a counterexample.

   (b) Is a dominating set always a vertex cover? Prove this, or find a counterexample.

   (c) What kind of independent set is also a dominating set?

6. The "cop-and-highway-robber game"[3] is played as follows.

   The playing board is a bipartite graph. One player (the highway robber) chooses an edge of the graph and occupies it for highway robbery. Simultaneously and without seeing the edge chosen, the other player (the cop) chooses a vertex of that edge to guard. If the cop guards an endpoint of the edge chosen by the highway robber, the highway robber gets caught and loses; otherwise, the highway robber gets away and wins.

[3]Whose name I made up, inspired by the study of cops-and-robbers games on graphs, which are something fancier.

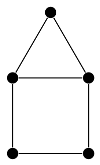
This is a game, like rock-paper-scissors, where both players need to choose their strategy at random; if either player plays predictably, then the other player can exploit this.

(a) Suppose that the graph has a matching M. Find a strategy for the highway robber to win with probability $1 - \frac{1}{|M|}$, no matter what the cop does (and prove that it works).

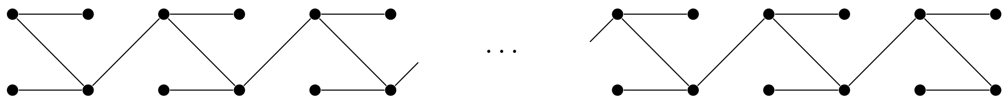
(b) Suppose that the graph has a vertex cover U. Find a strategy for the cop to win with probability $\frac{1}{|U|}$, no matter what the highway robber does (and prove that it works).

(Why is this another proof of Claim 2.1?)

(c) If $|M| = |U|$, these two strategies "meet in the middle" and end up being both optimal. If there is no M and U such that $|M| = |U|$, then it's possible that neither strategy is optimal. For an example of this, see if you can figure out the optimal strategy for both players on the "house graph" shown below:



7. To explore the difference between maximum and maximal matchings, consider the following graph (which can be made arbitrarily large):



(a) Find a perfect matching in the graph above

(b) Find the smallest matching in the graph above which is maximal (there is no larger matching that contains all of its edges).

(c) Let G be a graph (**not** necessarily bipartite) and let M be a maximal matching in G. Prove that G has a vertex cover U with $|U| = 2|M|$.

(d) Prove that the example in this problem is essentially as bad as it gets: if M is a maximal matching, then there is no matching with more than $2|M|$ edges.