Math 3322: Graph Theory[1] Mikhail Lavrov

Lecture 22: More About Matchings

November 2, 2021 *Kennesaw State University*

1 Other bipartite graph theorems

We ended the previous lecture by proving the following theorem:

Theorem 1.1 (König)**.** *In a bipartite graph G, the number of edges in a largest matching is equal to the number of vertices in a smallest vertex cover.*

König's theorem is good for upper bounds on the size of a matching, but occasionally we just want to verify that a matching which is as large as possible exists. For this, we have Hall's theorem.

For a set of vertices S in a graph G, let $N(S)$ be the set of all vertices adjacent to a vertex in S: the **neighborhood** of S. We write $N_G(S)$ if we want to specify which graph we're talking about.

Theorem 1.2 (Hall)**.** *A bipartite graph G with bipartition (A, B) has a matching that covers all of A if and only if it has the following property (known as **Hall's condition**):*

$$\text{For all } A \subseteq S,\ |N(S)| \geq |S|.$$

In particular, if $|A| = |B|$, this is the condition for G to have a perfect matching.

Proof. Where does Hall's condition come from? The logic is: if every element of S is covered by a matching, each vertex $u \in S$ is matched with some vertex $v \in N(S)$. (By "matched with" I just mean that edge uv is in the matching.) So for the $|S|$ different vertices in S, we need at least $|S|$ different vertices in $N(S)$ for them to be matched with. This condition generalizes some special cases:

- When $|S| = 1$, it says that in order for a matching to cover A, every vertex must have a neighbor.
- Checking the condition for $S = \{u, v\}$ rules out the case where u and v both have only one neighbor, and it's the same vertex.
- When $S = A$, Hall's condition implies that we must have $|B| \geq |A|$ to get a matching that covers all of A.

But this is all arguing that Hall's condition is necessary.

To prove it's sufficient, we will use König's theorem. Specifically, we will assume that there is no matching that covers all of A. Then, we will find a subset $A \subseteq S$ that violates Hall's condition.

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

Another way to say "there is no matching that covers all of A" is "there is no matching of size $|A|$", because every edge of a matching covers exactly one vertex in A. If there is no matching of size $|A|$, then by König's theorem, there is a vertex cover U of size at most $|A| - 1$.

Now let $S = A \setminus U$: the set of all vertices in A that are not part of the vertex cover U. If $v \in S$ and w is adjacent to v, then to cover edge vw, we must have $w \in U$ (since we cannot have $v \in U$). Therefore all of $N(S)$ must be contained in U.

Therefore U contains at least the $|A| - |S|$ vertices in A but not in U, and also at least the $|N(S)|$ vertices in $N(S)$ (which are all in B, so we're not double-counting). But we started by assuming that $|U| \leq |A| - 1$. Therefore

$$\big(|A| - |S|\big) + |N(S)| \leq |A| - 1$$

which can be rearranged to $|N(S)| \leq |S| - 1$, proving that S violates Hall's condition. □

There's lots of applications of Hall's theorem to various settings. Sometimes, we find a small set that violates Hall's condition, and immediately conclude a perfect matching is impossible. Sometimes we use it as a lemma for other theorems.

For example, there's the following (coincidentally also proved by König, so let's just call it "Theorem 1.3"):

Theorem 1.3. *If G is an r-regular bipartite graph (every vertex has degree r), then it has a perfect matching.*

Proof. Just to get some intuition, here is an observation first. If G has a bipartition (A, B), then we can count the edges of G in two ways. First, we could sum the degrees of all vertices in A and get $r|A|$: this counts every edge once, because every edge has an endpoint in A. We could also sum the degrees of all vertices in B and get $r|B|$. The two methods must give the same answer, so $r|A| = r|B|$, and $|A| = |B|$.

So there is always *hope* for a perfect matching in a regular bipartite graph, and this theorem goes on to say that the perfect matching always exists.

We will prove the theorem two ways: once using Hall's theorem, and once using König's theorem.

Using Hall's theorem. We will pick an arbitrary set $S \subseteq A$, and check that it satisfies Hall's condition.

There are exactly $r|S|$ edges with one endpoint in S. Each one of them has its other endpoint in $N(S)$. This doesn't mean that there are $r|S|$ vertices in $N(S)$: the same vertex in $N(S)$ could be the endpoint of many of the $r|S|$ edges. But actually, it can be the endpoint of at most r of them: so $|N(S)| \geq \frac{r|S|}{r} = |S|$.

Putting it differently: we have

$$\sum_{v \in S} \deg(v) \leq \sum_{w \in N(S)} \deg(w)$$

because the first sum counts all edges between S and $N(S)$, and the second sum counts all edges incident to a vertex in $N(S)$; this includes all edges counted by the first sum, and maybe more. But since all degrees are equal to r, we get $r|S| \leq r|N(S)|$, so $|N(S) \geq |S|$.

Either way, proving this and applying Hall's theorem gets us a perfect matching.

Using König's theorem. If $|A| = |B| = n$, then there are rn total edges in the graph. Let U be a minimum vertex cover. Each vertex of U can cover at most r edges, so to cover all rn edges, we need $|U| \geq n$. Also, $|U| = n$ is achievable: just take $U = A$, or $U = B$, for example.

Since the minimum vertex cover has n vertices, by König's theorem, the maximum matchng has n edges. This must cover all $2n$ vertices in the graph, so it is a perfect matching. □

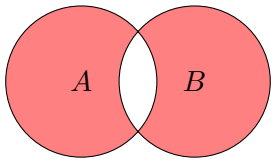
Theorem 1.3 generalizes to bipartite graphs that are "biregular". A (r, s)**-biregular bipartite graph** is a bipartite graph which has a bipartition (A, B) such that every vertex in A has degree r, and every vertex in B has degree s.

By double-counting as we did earlier, we have $r|A| = s|B|$, so if $r \neq s$, then $|A| \neq |B|$. We can't hope for a perfect matching. However, we can show that there is a matching of size at least $\min\{|A|, |B|\}$: a matching that covers the smaller side of the bipartition.

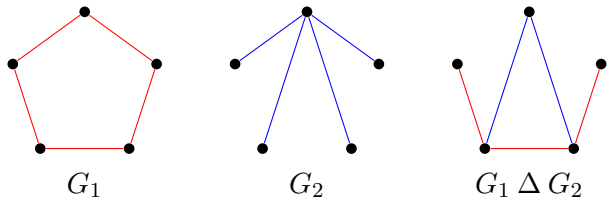
2 Augmenting paths

So far, we've just asked whether a matching exists or not. But if a matching exists, how do we find one?

Let's back up and talk about an operation called **symmetric difference**. At its most basic, it's defined on sets; if A, B are two sets, then $A \,\Delta\, B = (A \cup B) \setminus (A \cap B)$ is the set of all vertices in A or B but not both. Here is a Venn diagram:



We can step this up to graphs. We will only consider graphs G_1, G_2 with the same vertex set V. Then $G_1 \,\Delta\, G_2$ is the graph which also has vertex set V, and whose edge set is $E(G_1) \,\Delta\, E(G_2)$. That is, $G_1 \,\Delta\, G_2$ has every edge which is in exactly one of G_1 or G_2. Here is an example:



Suppose that M_1 and M_2 are two matchings in a graph G. (We will think of them as subgraphs of G, so that we can try to understand $M_1 \,\Delta\, M_2$.) Specifically, M_1 will be a small matching that we've managed to find so far, and M_2 is a hypothetical maximum matching that we haven't found yet.

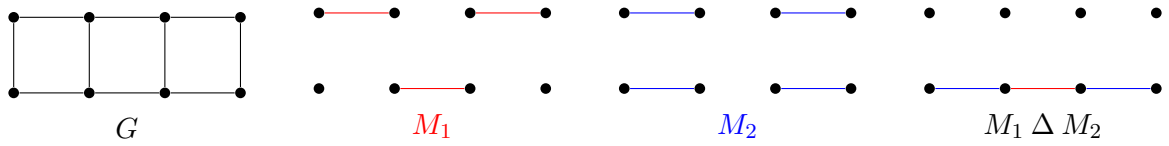
It makes sense to look at $M_1 \,\Delta\, M_2$, because this represents the edges that need to be changed in order to get M_2 from M_1. What can $M_1 \,\Delta\, M_2$ look like?

Remember that as subgraphs of G, M_1 and M_2 have all the vertices of G, and their degrees are either 0 or 1. So in $M_1 M_2$, a vertex can have degree at most 2: it has at most one neighbor in M_1, and at most one neighbor in M_2. Components of such a graph have three types:

- Isolated vertices. Here, nothing changes.
- 2-regular components. These are cycles. They must be even cycles, because they alternate edges from M_1 with edges from M_2. They are not very interesting for us.
- Connected components with some degree-1 vertices. These are all paths (with 2 vertices of degree 1 and the rest of degree 2): any fewer edges and they wouldn't be connected.

We are specifically interested in components of $M_1 \Delta M_2$ with more edges from M_2 than from M_1. None of the cycle components can do this: every cycle is an even cycle with an equal number of edges from each matching. We need a path with an odd length whose first and last edge come from M_2.

Here is an example of G, two matchings M_1 and M_2, and the symmetric difference $M_1 \Delta M_2$. In this case, $M_1 \Delta M_2$ happens to only consist of the path we want:



In general, we define an **M-augmenting path** for a matching M to be a path that begins and ends at two different vertices not covered by M, and alternates between "edge not in M" and "edge in M". The path we're looking for in $M_1 \Delta M_2$ is exactly an M_1-augmenting path.

Why are M-augmenting paths useful? Because they help us improve a matching M. If M is a matching and P is an M-augmenting path, then $M \Delta P$ is:

- Also a matching. The endpoints of P go from having degree 0 in M to degree 1 in $M \Delta P$. The internal vertices of P go from degree 1 to degree 1, just with a different edge.
- Bigger than M. All the vertices covered by M are still covered by $M \Delta P$, but the endpoints of P (which were not covered by M) are also covered by $M \Delta P$.

So we can use augmenting paths to make matchings bigger.

If $|M_1| < |M_2|$, we always find M_1-augmenting paths in $M_1 \Delta M_2$. That's because they are the only kind of component in $M_1 \Delta M_2$ that has more edges from M_2 than from M_1, and we know that as a whole, $M_1 \Delta M_2$ must have more edges from M_2 than from M_1. This isn't as helpful if we don't have M_2 to reference. But as long as there is some M_2 (even if we don't know what it is), we know that there is some M_1-augmenting path (even if we don't know what it is). Therefore every matching that is not maximum can be improved by finding an augmenting path.

2.1 Augmenting paths in bipartite graphs

In bipartite graphs, this can be done by a graph searching algorithm. Suppose we have a bipartite graph G with bipartition (A, B), and M is a matching in G. We will keep track of two sets: S and T. Initially, S is the set of all vertices in A not covered by M, and T is empty.

Then we repeat the following steps:

1. **Explore!** From each vertex in S, we go to all its neighbors and add them to T (if they were not already there).
2. **Win?** If there is a vertex in T that's not covered by M, stop immediately.
3. **Match!** Otherwise, all the vertices in T *are* covered by M. Take the vertices they're matched to, and add them to S.

What happens if we reach a **win** step? Well, we ended up at an uncovered vertex $b \in B$. If we trace back how we got there, we must have started at an uncovered vertex $a \in A$, and alternated **explore** steps (going to B along an edge not in M) and **match** steps (going back to A along an edge in M). That's precisely an M-augmenting path that starts at a and ends at b.

If we never reach a **win** step, that's also good news. In this case, we can find a vertex cover of size $|M|$: for each edge of M, take its endpoint in B if it is also in T, and otherwise take its endpoint in A. Let's check that this works. For any edge $ab \in E(G)$ with $a \in A$ and $b \in B$:

1. If a is not covered by M, then we added b to T in the very first step, so b is part of our cover.
2. If a is covered by M, and at some point we added a to S, then we would have added b to T in the next step (if not sooner), so b is part of our cover.
3. If a is covered by M, and we never added a to S, then the vertex a is matched to is not part of T (or else we would have added a to S). This is exactly the condition under which a is part of our cover.

So in all cases of an edge ab, one of a or b is in the cover, so it really does cover all the edges.

By the way, this is an alternate proof of König's theorem; if M is a maximum matching, then this algorithm will never find the M-augmenting path we wanted, so it'll find a vertex cover of size $|M|$ instead.

2.2 Augmenting paths in general graphs

Augmenting paths exist in all graphs, but finding them in non-bipartite graphs is hard. Let me try to give some intuition why.

Loosely speaking, the bipartition (A, B) keeps our search procedure organized in the bipartite case: we will always alternate between "take an edge outside M from A to B" and "take an edge of M from B to A". But in general graphs, it's possible that from the same starting point u, there's an even-length path to v (which ends by using an edge in M) and an odd-length path (which ends by using an edge not in M). This means that when we explore a vertex, we might do the wrong thing with it, because we got there by the wrong kid of path.

Look for the "blossom algorithm" if you want to find augmenting paths in general graphs, and prepare for a headache.