Math 301: Matchings in Graphs

Mary Radcliffe

1 Definitions and Basics

We begin by first recalling some basic definitions about matchings.

A *matching* in a graph G is a set $M = \{e_1, e_2, \dots, e_k\}$ of edges such that each vertex $v \in V(G)$ appears in at most one edge of M. That is, $e_i \cap e_j = \emptyset$ for all i, j. The *size* of a matching is the number of edges that appear in the matching. A *perfect matching* in a graph G is a matching in which every vertex of G appears exactly once, that is, a matching of size exactly $n/2$. Note that a perfect matching can only occur in a graph with evenly many vertices.

A matching M is called *maximal* if $M \cup \{e\}$ is not a matching for any $e \in E(G)$. A matching is called *maximum* if no other matching in G has a larger size. Note that these are NOT the same thing. Examples are shown in Figure 1(a, b), below.

For another example, consider a path on evenly many vertices, P_{2n}. Note that there are an odd number of edges. Let M_1 be the matching that takes every other edge, starting from the first vertex, and let M_2 be the matching that takes every other edge, starting from the second vertex. Note that both M_1 and M_2 are maximal matchings, since they cannot be extended by adding any more edges. However, M_2 is not a maximum matching, since it has strictly fewer edges than M_1. Of course, M_1 is a perfect matching, and hence it is a maximal matching.

Let M be a matching in a graph G. A path P in G is said to be M-alternating if every other edge in P appears in M. An M-augmenting path is an M-alternating path $P = (v_1, v_2, \dots, v_k)$ such that both v_1, v_k are NOT vertices of M. An example is shown in Figure 1c, below.

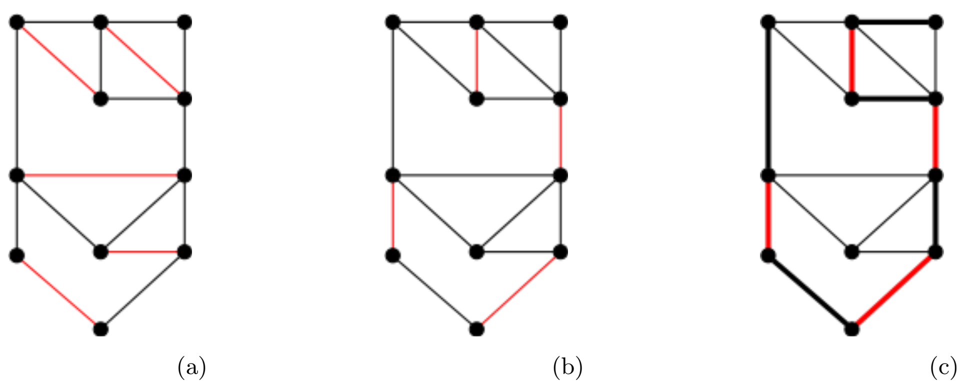

Figure 1: In each of these figures, the red edges indicate a matching. In (a), above, we have a maximum matching in the graph G. Notice that this matching must be a maximum matching, since it includes 5 edges on 11 vertices, and hence no other matching can be larger. In (b), we have a maximal matching; although it is only 4 edges, we cannot add any of the edges of G, since every edge that does not appear in the matching already is adjacent to an edge that does. In (c), the heavy edges indicate an M-augmented path for the maximal but not maximum matching from (b).

Perhaps surprisingly, we can totally characterize maximum matchings in terms of M-augmenting paths.

Theorem 1. *Let M be a matching in a graph G. Then M is a maximum matching if and only if there does not exist any M-augmenting path in G.*

Proof. Suppose that M is a matching in G, such that there does exist an M-augmenting path, say P. Notice that P must have odd length, since its edges alternate between edges in M and edge in $G\backslash M$, and further, it both begins and ends with edges from $G\backslash M$. Let M' be the set of edges in P that are not in M. Then notice that $|M'| > |M|$, and M' is a matching in G, so M is not a maximum matching. Hence, if M is a maximum matching, there cannot be any M-augmenting path in G.

For the converse, suppose that M is a matching having no M-augmented path in G. Let M_2 be a maximum matching in G. Note that by the above argument, there is no M_2-augmenting path in G.

Let H be the subgraph of G having $E(H) = \{e \in M \text{ but } e \notin M_2\} \cup \{e \in M_2 \text{ but } e \notin M\}$. Let us consider the possible components of H. Note that every vertex has degree 0, 1, or 2 in H. Vertices of degree 0 are disregarded, as their components are trivial. If a component has all vertices of degree 2, it must be an even cycle alternating between edges of M and edges of M_2. If a component has a vertex of degree 1, it must be a path, alternating between edges of M and edges of M_2. Note that since neither M nor M_2 has an augmented path in G, we must have that such a component begins with an edge from one matching, and ends with an edge from the other matching.

In either case, every nontrivial component of H has exactly half of its edges from M and exactly half of its edges from M_2. Hence, we must have that $|M\backslash M_2| = |M_2\backslash M|$. Hence,

$$|M| = |M \cap M_2| + |M\backslash M_2| = |M \cap M_2| + |M_2\backslash M| = |M_2|,$$

and thus M is also a maximum matching in G. □

Much of the time, we will specifically be interested in the following question. Suppose that A and B are subsets of G. We say that A can be *matched to* B if there exists a matching $M = \{e_1, e_2, \ldots, e_k\}$ such that each e_i has one vertex in A and the other in B, and every vertex in A and B appears in the matching. In order to create such a matching, we could simply focus on the collection of edges between A and B, and disregard all other edges in G. As such, we shall first spend some time focusing on bipartite graphs.

2 Matchings in Bipartite Graphs

Throughout this section, we shall be thinking of graphs G taking the following structure: the vertex set of G can be partitioned into two subsets, A and B, such that every edge in G has one endpoint in A and the other in B. We will use the notation $G(A, B)$ to denote a bipartite graph with partite sets A and B. This, of course, is just a bipartite graph. Recall also the notation $N(v) = \{u \in V(G) \mid u \sim v\}$, the set of neighbors of v. Given a set $S \subset V(G)$, we write $N(S) = \cup_{v \in S} N(v)$, that is, $N(S)$ is the set of vertices that are adjacent to at least one vertex in S. We sometimes write $N_G(S)$ if the graph is not clear. We have the following characterization of matchings in G.

Theorem 2 (Hall's Theorem)**.** *Let $G = G(A, B)$. Suppose that $|A| \le |B|$. Then there exists a matching M of size $|A|$ in G if and only if for every subset $S \subset A$, we have that $|N(S)| \ge |S|$.*

In particular, if $|A| = |B|$, then G has a perfect matching under this condition.

We will see in later work that Hall's Theorem is equivalent to many other useful combinatorial theorems, like the Max Flow/Min Cut theorem, Dilworth's Theorem, and several others. We shall refer to the condition that for every subset $S \subset A$, we have that $|N(S)| \ge |S|$ as *Hall's condition.*

Proof. Write $A = \{u_1, u_2, \ldots, u_k\}$ and $B = \{v_1, v_2, \ldots, v_\ell\}$, with $k \le \ell$.

First, suppose that there exists a matching M of size $|A|$ in G. Since G is bipartite, every edge of M includes one vertex from A. By possibly relabeling B, suppose that $M = \{e_1, e_2, \ldots, e_k\}$, where $e_i = \{u_i, v_i\}$ for each i.

Let $S \subset A$, with $S = \{u_{s_1}, u_{s_2}, \ldots, u_{s_t}\}$. Then $v_{s_j} \in N(S)$ for all $1 \leq j \leq t$, and hence $|N(S)| \geq t = |S|$. Therefore, the forward direction is proved.

For the converse, we proceed by induction on $|A|$. First, note that if $|A| = 1$, the theorem is trivial. Let us then suppose that the theorem is known if $|A| = k - 1$.

Let $G = G(A, B)$ be a bipartite graph with $|A| = k$ satisfying Hall's condition. We consider two cases.

Case 1: for every proper subset S of A, $|N(S)| \geq |S| + 1$.

Consider u_1; wolog, suppose that $u_1 \sim v_1$ (and possibly some other vertices also). Consider the subgraph H of G on $A \backslash \{u_1\}, B \backslash \{v_1\}$; that is, remove both vertices u_1 and v_1 from consideration. Let S be a subset of $A \backslash \{u_1\}$. Then note that $N_H(S)$ is either equal to $N_G(S)$, or has been reduced in size by 1 due to the removal of v_1. In either case, $|N_H(S)| \geq |N_G(S)| - 1 \geq |S|$, and hence H satisfies Hall's condition. We can therefore obtain a matching in H of size $|A| - 1$ by the induction hypothesis; adding the edge $\{u_1, v_1\}$ to such a matching produces a matching in G of size $|A|$.

Case 2: A contains a proper subset S having $|S| = |N(S)|$.

Note that since S is a proper subset of A, by the induction hypothesis, we have that the subgraph of G on $(S, N(S))$ satisfies Hall's condition, and hence, we may find a matching on this subgraph of size $|S|$. Let H be the subgraph of G on $(A \backslash S, B \backslash N(S))$; note that we have removed the same number of vertices from both A and B.

Let T be a subset of $A \backslash S$. Then notice that $N_G(S \cup T) = N_G(S) \cup N_H(T)$, and these two neighborhoods are disjoint. Furthermore, by Hall's condition, we have $|N_G(S \cup T)| \geq |S \cup T| = |S| + |T|$. Therefore,

$$|N_H(T)| = |N_G(S \cup T)| - |N_G(S)| \geq |S| + |T| - |S| = |T|,$$

and hence H satisfies Hall's condition. We thus can find a matching in H of size $|A \backslash S|$. Taking the union of this matching with the matching on S yields a matching in G of size $|A|$, as desired.

□

We note that sometimes, this condition is referred to as Hall's Marraige Condition, due to the following example:

Example 1. Suppose we have a set W of n heterosexual girls and a set M of n heterosexual boys, and we create a bipartite graph $G = G(W, M)$ with an edge from u to v if girl u likes boy v.

The boys wish to ask the girls to prom (ok, in the classical example, they're asking for the girls' hands in marriage, but if a girl likes more than one boy, this seems like a longshot), and because they are high school boys, nobody wants to be rejected. We further assume that if a girl likes a boy, she will always say yes to his prom invitation.

The boys want to figure out which girls to ask to the dance. Note that this is precisely the question of finding a matching in the bipartite graph G described. Hence, Hall's Theorem tells us that the boys can collude on such a strategy precisely when Hall's condition is met, that is, if every subset of girls likes a subset of boys at least as large, such a matching can be found. (Alas, high school is rarely so simple.)

2.1 Stable Matchings

Let us consider a slight variant on this problem, known as the *Stable Marriage Problem*. Here, we shall consider a weighted bipartite graph, as follows.

Example 2. Suppose we are in a similar situation as Example 1, except now, the girls have all ranked the boys in order of preference, and likewise the boys have all ranked the girls in order of preference. We wish to find a matching of boys to girls, so that there is not any *rogue* couple. A rogue couple is a pair (u, v), one girl and one boy, such that u prefers v to their current partner and v also prefers u to their current partner. This rogue couple might be a problem, as they would probably rather ditch their dates and leave the prom together.

This is essentially the structure we wish to avoid. Such a matching can be constructed algorithmically, as follows (in an astonishingly high school way). We shall work through as many time steps as necessary until everyone is paired off. This is known as the Gale-Shapley algorithm.

In each time step:

1. Every boy who doesn't have a date yet looks at his preferences, and asks his favorite girl who is not yet crossed off.
2. Each girl looks through her prom proposals (if she has any), and picks her favorite boy to say yes to. If she already has a date, but someone she likes better asks her, she dumps her original date in favor of the new guy, and rejects everyone else.
3. Every boy who has been rejected crosses the girl who did the rejecting off his list.

Let us convince ourselves that this algorithm will produce a stable set of prom dates. First, we should convince ourselves that the algorithm produces a matching at all. Note that once a girl gets asked to prom, she MUST say yes to someone. Once she is asked, she will have a date, and her only possible action is to "trade up" to a more preferable suitor. Hence, we cannot end up with two people both having no date, because (even if they don't like each other that much), the rules say that any unbound boy MUST ask a girl at each round, and so eventually the boy would ask the girl.

Good, so everyone has a date. How about stability?

Let's suppose we had an unstable pair, a girl u and boy v so that u is paired with v' but prefers v, and v is paired with u' but prefers u. Notice that since the boy v prefers girl u to u', he must have asked girl u to prom before asking girl u', and was rejected. That means girl u must have had a date that she had higher preference for than boy v. But she would not have ditched that date for a lower preference option, so she cannot be paired with a boy who she has lower preference for than boy v.

Awesome, so it's also stable! But it is eminently weird. Under this algorithm, for example, we have the following weird dichotomy:

> Every boy goes to prom with his optimal girl, and every girl goes to prom with her LEAST optimal boy.

To understand this weirdness, we first need to understand what is meant by "optimal" here. We don't really mean the person who the boy favors the most; it may be that every boy favors the same girl the most, and hence not everyone can go to prom with the person they like the most. Instead, the optimality is considered with respect to the set of stable matchings. That is, if $\mathcal{M} = \{\text{stable matchings of boys to girls}\}$, then every boy is paired with his optimal mate among those possible in $\mathcal{M}$, and every girl with her least optimal mate.

The proof of this weirdness mostly involves digging in a little to the definition of optimal, and the Gale-Shapley algorithm itself.

Proof. Suppose that there is a boy v who is not paired to his optimal girl u. This implies that at some point in the algorithm, he is rejected by her. Let us choose, then, the boy v and corresponding optimal girl u that is the first rejection of a boy by his optimal girl in the algorithm.

Note that since u rejected v, she must have a current partner, say z, who she prefers to u. Since v was the first boy that was rejected by his optimal mate, it must be that z has yet to be rejected by his optimal mate; that is, z has yet to ask his optimal mate to prom at all. However, since u is not his optimal mate, u must not be a possible match for z in $\mathcal{M}$.

Let $M \in \mathcal{M}$ be a matching in which u and v are paired. Such a matching must exist, since u is the optimal mate to v. But then, u is not paired to z. Suppose instead that y is paired to z. Note that z must prefer u over y, and moreover, u certainly prefers z over v. But then we have a rogue pair (u, z), and hence M is not a stable matching at all. Thus, it must be that under the Gale-Shapley algorithm, every boy is paired to his optimal girl.

For the other part, we wish to show that the opposite fact occurs for the girls. Let M be the stable matching obtained under the Gale-Shapley algorithm. Let u be a girl, and let v be her prom date in M.

Now, let M_1 be any other matching in $\mathcal{M}$, and suppose that u has partner z and v has partner y in M_1. Note that by the first part of this proof, we must have that v prefers u to y, since he was paired with his optimal partner in M. But then, in order to avoid a rogue couple, we must have that u prefers z to v. That is, in any other matching, u gets a "better" boy. $\square$

There are many applications of the Stable Marriage problem, and the Gale-Shapley algorithm (and variants thereupon). Here are some real ones that actually happen:

- NRMP: The National Resident Matching Program. Medical students fill out some forms, indicating preferences on where they would like to serve their residency. Hospitals fill out some other forms, indicating what type of doctors they prefer or need for residents. An algorithm very similar to Gale-Shapley matches up the newly minted doctors to residencies (variant: many hospitals accept more than one resident). Great news for hospitals: because of our weird dichotomy, they always get their optimal doctors! Crappy news for doctors: they always get their LEAST optimal hospitals!
- College admissions. Students here play the role of the boys, desperately seeking acceptance from their most optimal schools. An interesting twist is the wait-list: the "girls" of the algorithm (aka, the schools), get to keep a list of potential students basically on deck for the inevitability of their highest preferences falling through. Notice who "wins" this: it's students. Students should always obtain their optimal schools, whereas schools get the least optimal collection of students.
- Stable roommates. This is a variant of the G-S algorithm in which, rather than having two sets of people that need to be matched up, we have a single pool of people that needs to be paired up. Think of it as the homosexual version of the prom problem: Boys ask other boys. The same basic dynamics are in play, but unfortunately we cannot always guarantee a stable matching (sad).
- Queuing: Which machine should get which jobs? This is an interesting variant, as the preference of machines to jobs might change over time. That is, if you give a machine a long job, it might change its preference to shorter jobs in the immediate future, or some other mechanism for making sure we don't have one machine doing all the heavy lifting.deba

3 Tutte's Theorem

Let us now turn our attention to matchings in non-bipartite graphs. This, in a sense, is like the stable roommates problem above: we don't have two sets to match up, just one big one. We certainly can observe some obvious features of perfect matchings:

- If G has an odd number of vertices, then it has no perfect matching.
- If G has any isolated vertices, then it has no perfect matching.
- If G has a component of odd size, then it has no perfect matching.

In fact, this last point about components of odd size is the critical issue. To develop the theorem, we shall first require some notation. Given any graph G, let $o(G)$ denote the number of odd-sized components of G; these will be referred to simply as odd components for the remainder of this section. Tutte's Theorem gives us a characterization of when G has a perfect matching in terms of odd components.

Theorem 3 (Tutte's Theorem)**.** *Let G be a graph. Then G contains a perfect matching if and only if for every proper subset $S \subset V(G)$ we have $o(G \backslash S) \leq |S|$.*

Proof. We first consider the forward implication. Suppose that G contains a perfect matching M. Let $S = \{v_1, v_2, \ldots, v_k\} \subset V(G)$ be a proper subset of $V(G)$.

Let M' be the set of edges in M that do not include any vertices from S. Let H be a component of $G \backslash S$. Note that the vertices in H come in two flavors: either they are incident to an edge from M', or they are not. Note that we must have evenly many vertices in H that are incident to edges from M', as they are paired up together. Hence, the component H can only be odd if it has an odd number of vertices whose partner in M is in S. There can only be at most $|S|$ many such components, and hence $o(G \backslash S) \leq |S|$.

Let us now consider the backward implication. Let G be a graph on n vertices. We shall proceed by induction on n. Note the base case is when $n = 2$, and the only graph satisfying Tutte's condition is K_2. Hence, we assume that if a graph has $n - 2$ or fewer vertices (note, we may delete two vertices, since no odd graph has a perfect matching, and hence we tacitly assume n is even), then the theorem holds.

So, we have a graph on n vertices, in which Tutte's condition is satisfied. We consider two cases (note: this approach is similar to the approach we took with Hall's Theorem).

Case 1: For every proper subset S of $V(G)$, $o(G \backslash S) \leq |S| - 1$.

Note that as n is even, we must have that $o(G \backslash S)$ and $|S|$ are of the same parity, so in fact we have $o(G \backslash S) \leq |S| - 2$ for every proper subset of $V(G)$.

Fix an edge uv, and consider $H = G \backslash \{u, v\}$, having $n - 2$ vertices. Let $T \subset V(H)$. Note that $o(H \backslash T) = o(G \backslash (T \cup \{u, v\})) \leq |T \cup \{u, v\}| - 2 = |T|$. Hence, H satisfies Tutte's criterion, and thus H has a perfect matching M'. Taking $M = M' \cup \{uv\}$ yields a perfect matching in G.

Case 2: There exists a proper subset S of $V(G)$ with $o(G \backslash S) = |S|$.

Let S be the largest proper subset of $V(G)$ having $o(G \backslash S) = |S| = k$, and write $S = \{v_1, v_2, \ldots, v_k\}$. We first claim that $G \backslash S$ contains only odd components. Indeed, if H were an even component of $G \backslash S$, then fix any vertex $u_0 \in V(H)$. Note that $H \backslash \{u_0\}$ must have at least one odd component, as it has an odd number of vertices, and hence $|S \cup \{u_0\}| \geq o(G \backslash (S \cup \{u_0\}) \geq |S \cup \{u_0\}|$, yielding a strictly larger set satisfying the hypothesis of the case. Hence, no even component may exist.

Let $G_1, G_2, \ldots, G_k$ be the k components of $G \backslash S$, and each of these is odd. Create a bipartite graph B as follows.

Let $V = S$, and let $U = \{u_1, u_2, \ldots, u_k\}$. Put an edge $u_i v_j$ in B if v_j is adjacent to some vertex in G_i.

Claim: B satisfies Hall's criterion.

Pf. of Claim: Suppose T is a proper subset of U, with $|T| = t$. Suppose that $|N(T)| < |T|$. Let $S' = \{v_{i_1}, v_{i_2}, \ldots, v_{i_r}\} = N(T) \subset S$, with $r < t$. Note then that for each $u_i \in T$, we have that G_i is a component of $G \backslash S'$, since it is adjacent to no other vertices in S. Hence $o(G \backslash S') \geq t > r = |S'|$, a violation of Tutte's condition. Hence, B satisfies Hall's criterion. $\square$

Therefore, there exists a perfect matching M in B. Wolog, relabel the components of $G \backslash S$ so that we have, for every $v_i \in S$, that v_i is adjacent to some vertex of G_i; call this vertex u_i.

Let us first show that we can find a perfect matching in $G_j \backslash u_j$ for all j. If $|V(G_j)| = 1$, then we are done. If not, suppose that $|V(G_j)| \geq 3$. Let $H = G_j \backslash \{u_j\}$. Let $W \subset V(H)$ be a proper subset. Note that $o(H \backslash W) = o(G \backslash (S \cup \{u_j\} \cup W)) - (k - 1)$, since we have all the odd components $G_1, G_2, \ldots, G_k$, excepting G_j counted in the right hand side, which can be rectified by simply subtracting $k - 1$. Moreover, due to maximality of S, we have $o(G \backslash (S \cup \{u_j\} \cup W) < |S \cup \{u_j\} \cup W| = |W| + k + 1$, and hence $o(H \backslash W) < |W| + k + 1 - (k - 1) = |W| + 2$. As $o(H \backslash W)$ and $|W|$ must have the same parity, the strict inequality thus implies that $o(H \backslash W) \leq |W|$. Hence, H satisfies Tutte's criterion, and thus by induction we can form a perfect matching in H.

By performing the above procedure, we thus can form a (possibly empty) perfect matching M_j in $G_j \backslash \{u_j\}$ for all j. Taking $M_1 \cup M_2 \cup \cdots \cup M_j \cup \{u_1v_1, u_2v_2, \ldots, u_kv_k\}$ yields a perfect matching in G.

$\square$

Tutte's Theorem is the natural generalization of Hall's Theorem to the universe of nonbipartite graphs. We note that applications mentioned above for stable matchings also apply in this case, except the matchings will not be on a bipartite graph. Similar other applications can be found for Tutte's theorem, especially in the universe of decomposition of graphs into matchings (so that we have a list of matchings covering every edge of the graph).