1. Lecture notes on bipartite matching

Matching problems are among the fundamental problems in combinatorial optimization. In this set of notes, we focus on the case when the underlying graph is bipartite.

We start by introducing some basic graph terminology. A *graph* $G = (V, E)$ consists of a set V of *vertices* and a set E of pairs of vertices called *edges*. For an edge $e = (u, v)$, we say that the *endpoints* of e are u and v; we also say that e is *incident* to u and v. A graph $G = (V, E)$ is *bipartite* if the vertex set V can be partitioned into two sets A and B (*the bipartition*) such that no edge in E has both endpoints in the same set of the bipartition. A matching $M \subseteq E$ is a collection of edges such that every vertex of V is incident to at most one edge of M. If a vertex v has no edge of M incident to it then v is said to be *exposed* (or unmatched). A matching is *perfect* if no vertex is exposed; in other words, a matching is perfect if its cardinality is equal to $|A| = |B|$.

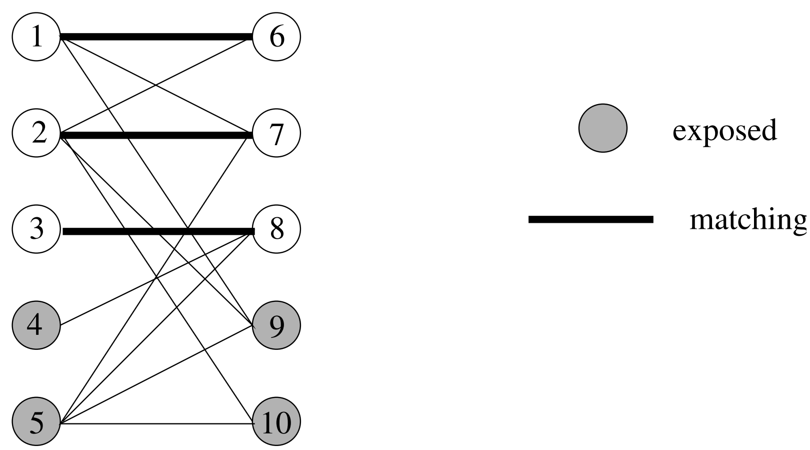

Figure 1.1: Example. The edges $(1, 6)$, $(2, 7)$ and $(3, 8)$ form a matching. Vertices 4, 5, 9 and 10 are exposed.

We are interested in the following two problems:

Maximum cardinality matching problem: Find a matching M of maximum size.

Minimum weight perfect matching problem: Given a cost c_{ij} for all $(i, j) \in E$, find a perfect matching of minimum cost where the cost of a matching M is given by $c(M) = \sum_{(i,j)\in M} c_{ij}$. This problem is also called the *assignment problem*.

Similar problems (but more complicated) can be defined on non-bipartite graphs.

1.1 Maximum cardinality matching problem

Before describing an algorithm for solving the maximum cardinality matching problem, one would like to be able to prove optimality of a matching (without reference to any algorithm). For this purpose, one would like to find upper bounds on the size of any matching and hope that the smallest of these upper bounds be equal to the size of the largest matching. This is a *duality* concept that will be ubiquitous in this subject. In this case, the dual problem will itself be a combinatorial optimization problem.

A *vertex cover* is a set C of vertices such that all edges e of E are incident to at least one vertex of C. In other words, there is no edge completely contained in $V - C$ (we use both $-$ and $\setminus$ to denote the difference of two sets). Clearly, the size of any matching is at most the size of any vertex cover. This follows from the fact that, given any matching M, a vertex cover C must contain at least one of the endpoints of each edge in M. We have just proved *weak duality*: The maximum size of a matching is at most the minimum size of a vertex cover. As we'll prove later in these notes, equality in fact holds:

Theorem 1.1 (König 1931) *For any bipartite graph, the maximum size of a matching is equal to the minimum size of a vertex cover.*

We shall prove this *minmax* relationship algorithmically, by describing an efficient algorithm which simultaneously gives a maximum matching and a minimum vertex cover. König's theorem gives a *good characterization* of the problem, namely a simple proof of optimality. In the example above, one can prove that the matching $(1, 9)$, $(2, 6)$, $(3, 8)$ and $(5, 7)$ is of maximum size since there exists a vertex cover of size 4. Just take the set $\{1, 2, 5, 8\}$.

The natural approach to solving this cardinality matching problem is to try a *greedy* algorithm: Start with any matching (e.g. an empty matching) and repeatedly add disjoint edges until no more edges can be added. This approach, however, is not guaranteed to give a maximum matching (convince yourself). We will now present an algorithm that does work, and is based on the concepts of *alternating paths* and *augmenting paths*. A path is simply a collection of edges $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$ where the v_i's are distinct vertices. A path can simply be represented as v_0-v_1-$\ldots$-v_k.

Definition 1.1 *An* **alternating path with respect to** M *is a path that alternates between edges in* M *and edges in* $E - M$.

Definition 1.2 *An* **augmenting path with respect to** M *is an alternating path in which the first and last vertices are exposed.*

In the above example, the paths 4-8-3, 6-1-7-2 or 5-7-2-6-1-9 are alternating, but only the last one is augmenting. Notice that an augmenting path with respect to M which contains k edges of M must also contain exactly $k + 1$ edges not in M. Also, the two endpoints of an augmenting path must be on different sides of the bipartition. The most interesting property of an augmenting path P with respect to a matching M is that if we set $M' = M \triangle P \equiv (M - P) \cup (P - M)$, then we get a matching M' and, moreover, the size of

M' is one unit larger than the size of M. That is, we can form a larger matching M' from M by taking the edges of P not in M and adding them to M' while removing from M' the edges in M that are also in the path P. We say that we have *augmented M along P*.

The usefulness of augmenting paths is given in the following theorem.

Theorem 1.2 *A matching M is maximum if and only if there are no augmenting paths with respect to M.*

Proof: (By contradiction)

($\Rightarrow$) Let P be some augmenting path with respect to M. Set $M' = M \bigtriangleup P$. Then M' is a matching with cardinality greater than M. This contradicts the maximality of M.

($\Leftarrow$) If M is not maximum, let M^* be a maximum matching (so that $|M^*| > |M|$). Let $Q = M \bigtriangleup M^*$. Then:

- Q has more edges from M^* than from M (since $|M^*| > |M|$ implies that $|M^* - M| > |M - M^*|$).
- Each vertex is incident to at most one edge in $M \cap Q$ and one edge $M^* \cap Q$.
- Thus Q is composed of cycles and paths that alternate between edges from M and M^*.
- Therefore there must be some path with more edges from M^* in it than from M (all cycles will be of even length and have the same number of edges from M^* and M). This path is an augmenting path with respect to M.

Hence there must exist an augmenting path P with respect to M, which is a contradiction. $\triangle$

This theorem motivates the following algorithm. Start with any matching M, say the empty matching. Repeatedly locate an augmenting path P with respect to M, augment M along P and replace M by the resulting matching. Stop when no more augmenting path exists. By the above theorem, we are guaranteed to have found an optimum matching. The algorithm terminates in μ augmentations, where μ is the size of the maximum matching. Clearly, $\mu \leq \frac{n}{2}$ where $n = |V|$.

In the example, one would thus augment M along an augmenting path, say 5-7-2-6-1-9, obtain the matching $(1, 9)$, $(2, 6)$, $(3, 8)$ and $(5, 7)$, and then realize that no more augmenting paths can be found.

The question now is how to decide the existence of an augmenting path and how to find one, if one exists. These tasks can be done as follows. Direct edges in G according to M as follows : An edge goes from A to B if it does not belong to the matching M and from B to A if it does. Call this directed graph D.

Claim 1.3 *There exists an augmenting path in G with respect to M iff there exists a directed path in D between an exposed vertex in A and an exposed vertex in B.*

Exercise 1-1. Prove claim 1.3.

This gives an $O(m)$ algorithm (where $m = |E|$) for finding an augmenting path in G. Let A^* and B^* be the set of exposed vertices w.r.t. M in A and B respectively. We can simply attach a vertex s to all the vertices in A^* and do a depth-first-search from s till we hit a vertex in B^* and then trace back our path.

Thus the overall complexity of finding a maximum cardinality matching is $O(nm)$. This can be improved to $O(\sqrt{n}m)$ by augmenting along several augmenting paths simultaneously.

If there is no augmenting path with respect to M, then we can also use our search procedure for an augmenting path in order to construct an optimum vertex cover. Consider the set L (for Labelling) of vertices which can be reached by a directed path from an exposed vertex in A.

Claim 1.4 *When the algorithm terminates, $C^* = (A - L) \cup (B \cap L)$ is a vertex cover. Moreover, $|C^*| = |M^*|$ where M^* is the matching returned by the algorithm.*

This claim immediately proves König's theorem.

Proof: We first show that C^* is a vertex cover. Assume not. Then there must exist an edge $e = (a, b) \in E$ with $a \in A \cap L$ and $b \in (B - L)$. The edge e cannot belong to the matching. If it did, then b should be in L for otherwise a would not be in L. Hence, e must be in $E - M$ and is therefore directed from A to B. This therefore implies that b can be reached from an exposed vertex in A (namely go to a and then take the edge (a, b)), contradicting the fact that $b \notin L$.

To show the second part of the proof, we show that $|C^*| \leq |M^*|$, since the reverse inequality is true for any matching and any vertex cover. The proof follows from the following observations.

1. No vertex in $A - L$ is exposed by definition of L,
2. No vertex in $B \cap L$ is exposed since this would imply the existence of an augmenting path and, thus, the algorithm would not have terminated,
3. There is no edge of the matching between a vertex $a \in (A - L)$ and a vertex $b \in (B \cap L)$. Otherwise, a would be in L.

These remarks imply that every vertex in C^* is matched and moreover the corresponding edges of the matching are distinct. Hence, $|C^*| \leq |M^*|$. $\triangle$

Although the concepts of maximum matchings and minimum vertex covers can be defined also for general (i.e. non-bipartite) graphs, we should remark that König's theorem does not generalize to all graphs. Indeed, although it is true that the size of a maximum matching is *always* at most the minimum size of a vertex cover, equality does not necessarily hold. Consider indeed the cycle C_3 on 3 vertices (the smallest non-bipartite graph). The maximum matching has size 1, but the minimum vertex cover has size 2. We will derive a minmax relation involving maximum matchings for general graphs, but it will be more complicated than König's theorem.