

Matching

December 15, 2003

1 Definitions

Definition 1.1 For a graph $G = (V, E)$ a set $M \subseteq E$ of edges is a **matching** if no pair of edges of M has a common vertex.

A matching is **perfect** if it covers all the vertices of G . For a weight function w , which assigns real weight to the edges of G , a matching M is a **maximal weight matching**, if M is a matching and $w(M) = \sum_{e \in M} w(e)$ is maximal.

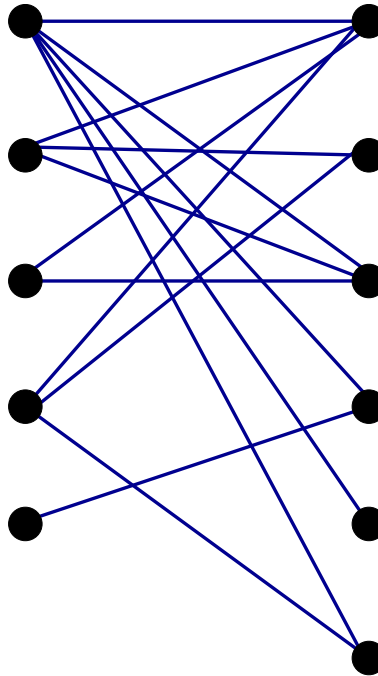
Definition 1.2 If there is no weight on the edges, we consider the weight of every edge to be one, and in this case, we are trying to compute a **maximum size matching**.

Problem 1.3 Given a graph G and a weight function on the edges, compute the maximum weight matching in G .

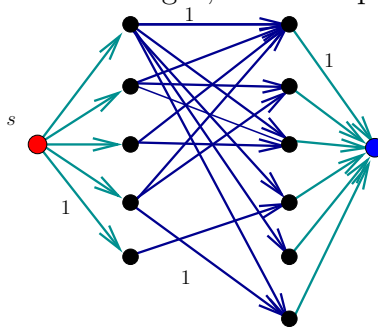
There is a simple reduction showing how to compute a maximum size matching in a bipartite graph. How???

1.1 Unweighted matching in a bipartite graph

Consider the bipartite graph G :



We add a source and destination vertices, connected them to the left and right side respectively, orient the edges from left to right, set the capacity of all edges to 1. We get



It is now easy to argue that the maximum flow in this new graph corresponds to the maximum size matching in the original graph, where an edge has a flow 1 if it is in the matching.

2 Matchings and Alternating Paths

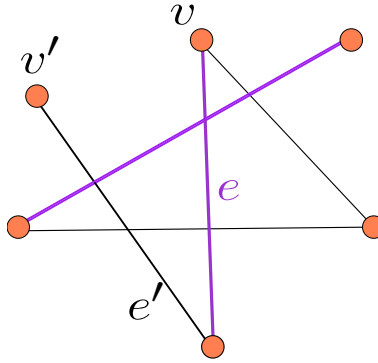
M - a matching.

$e \in M$ is a **matching edge**.

Any edge $e' \in E(G) \setminus M$ is **free**.

$v \in V(G)$ is **matched** if it is adjacent to an edge in M .

A vertex v' *which* is not matched, is **free**.



An *alternating path* is a simple path that its edges are alternately matched and free. *Alternating cycle* is defined similarly. The *length* of a path/cycle is the number of edges in it.

For an alternating path/cycle π , its *weight* is

$$\gamma(\pi) = \sum_{e \in \pi \setminus M} w(e) - \sum_{e \in \pi \cap M} w(e)$$

namely, it is the total weight of the free edges in π minus the weight of the matched edges. This is interesting because of the following lemma.

Lemma 2.1 *Let M be a matching, and let π be an alternating path/cycle with positive weight such that*

$$M' = M \oplus \pi = (M \setminus \pi) \cup (\pi \setminus M)$$

is a matching, then $w(M')$ is ????
bigger. Namely, $w(M') > w(M)$.

Proof: $w(M') = w(M) + \gamma(\pi)$. ■

Definition 2.2 An alternating path is *augmenting* if it starts and ends in a free vertex.

Observation 2.3 *If M has an augmenting path π , then it is not of maximum size (i.e., unweighted case). (Because $M \oplus \pi$ is a larger matching).*

Theorem 2.4 *Let M be a matching, and T be a maximum size matching, and $k = |T| - |M|$. Then M has k vertex disjoint augmenting paths. At least one of length $\leq n/k - 1$.*

Proof: Let $E' = M \oplus T$, and let $H = (V, E')$. Clearly, every vertex in H has at most degree 2. (Why?)

Because every vertex is adjacent to at most one edge of M and T . Thus, H is a collection of paths and (even length) cycles. (why are the cycle even length?)
 Because it is an alternating cycle for M .

Now, there are k more edges of T in $M \oplus T$ than of M . Every cycle have the same number of edges of M and T . Thus, a path in H can have at most one more edge of T than

of M (and in this case it is an augmenting path for M). It follows that there are at least k augmenting paths for M in H .

As for the length of the shortest augmenting path, it follows by observing that if all those (vertex disjoint) augmenting paths were of length n/k then the total number of vertices in H would be $(n/k + 1)k > n$. A contradiction. ■

Theorem 2.5 *Let M be a matching of maximum weight among matchings of size $|M|$. Let π be an augmenting path for M of maximum weight, and let T be the matching formed by augmenting M using π . Then T is of maximum weight among matchings of size $|M| + 1$.*

Proof: Let S be a matching of maximum weight among all matchings with $|M| + 1$ edges. And consider $H = (V, M \oplus S)$.

Consider a cycle σ in H . The weight $\gamma(\sigma)$ must be...

zero!. Indeed, if $\gamma(\sigma) > 0$ then $M \oplus \sigma$ is a matching of the same size as M which are heavier than M . A contradiction to the definition of M as the maximum weight such matching.

Similarly, if $\gamma(\sigma) < 0$ then $S \oplus \sigma$ is heavier than S . A contradiction.

Similarly, if σ is a path which is a connected component of H with even number of edges, then $\gamma(\sigma) = 0$ by the same argumentation.

Let U_T be all the odd length paths in H that have one edge more in T than in M .

Similarly, let U_M be the odd length paths in H that have one edge more of M than an edge of T .

We know that $|U_T| - |U_M| = ???$

We know that $|U_T| - |U_M| = 1$. Now, consider a path $\pi \in U_T$ and a path $\pi' \in U_M$. It must be that $\gamma(\pi) + \gamma(\pi') = ????$.

Indeed, if $\gamma(\pi) + \gamma(\pi') > 0$ then $M \oplus \pi \oplus \pi'$ would have bigger weight than M while having the same number of edges.

Similarly, if $\gamma(\pi) + \gamma(\pi') < 0$ (compared to M) then $S \oplus \pi \oplus \pi'$ would have the same number of edges as S while being a heavier matching. A contradiction. ■

Thus, $\gamma(\pi) + \gamma(\pi') = 0$. Thus, we can pair up the paths in U_T to paths in U_M , and the total weight of such a pair is zero, by the above argumentation. There is only one path μ in U_T which is not paired, and it must be that $\gamma(\mu) = w(T) - w(M)$ (since everything else in H has zero weight as we apply it to M to get T). Thus, establishing the claim.

The above theorem imply that if we always augment along the maximum weight augmenting path, then we would get the maximum weight matching in the end.

3 Maximum Weight Matchings in A Bipartite Graph

Let $G = (L \cup R, E)$ be the given bipartite graph, with $w : E \rightarrow \mathbb{R}$ be the non-negative weight function. Given matching M we define the graph G_M to be the directed graph, where if

$$l \in L, r \in R$$

$$rl \in M \text{ then } (r \rightarrow l) \in E(G_M) \text{ and } \alpha(r \rightarrow l) = w(rl)$$

$$rl \in E \setminus M \text{ then } (l \rightarrow r) \in E(G_M) \text{ and } \alpha(l \rightarrow r) = -w(rl)$$

Namely, we direct all the matching edges from right to left, and assign them a their weight, and we direct all other edges from left to right, with their negated weight.

An augmenting path π in G must have an odd number of edges. Since G is bipartite, π must have one endpoint on the left side, and one endpoint on the right side.

Observe, that a path π in G_M has weight $\alpha(\pi) = -\gamma(\pi)$ where $\gamma(\pi)$ is the weight function defined by M and G .

Let U_L be all the unmatched vertices in L and let U_R be all the unmatched vertices in R .

Thus, what we are looking for is a path π in G_M going starting U_L going to U_R with maximum weight $\gamma(\pi)$, namely with minimum weight $\alpha(\pi)$.

Lemma 3.1 *If M is a maximum weight matching with k edges in G , than there is no negative cycle in G_M where $\alpha(\cdot)$ is the associated weight function.*

Proof: ???

Assume for the sake of contradiction that there is a cycle C , and observe that $\gamma(C) = -\alpha(C) > 0$. Namely, $M \oplus C$ is a new matching with bigger weight and the same number of edges. A contradiction to the maximality of M . ■

So, we now can find a maximum weight in the bipartite graph G as follows: Find a maximum weight matching M with k edges, compute the maximum weight augmenting path for M apply it, and repeat till M is maximal.

Thus, we need to find a minimum weight path in G_M between U_L and U_R (because we flip weights). How do we do it?

This is just computing a shortest path in the graph G_M which does not have negative cycles, and this can just be done by doing Dijkstra. Indeed, collapse all the vertices of U_L into a single vertex, and all the uncovered vertices of U_R into a single vertex. Let H_M be the resulting graph. Clearly, we are looking for the shortest path between the two vertices corresponding to U_L and U_R in H_M and since this graph has no negative cycles, this can be done using Bellman-Ford algorithm, which takes $O(nm)$ time. We conclude:

Lemma 3.2 *Given a bipartite graph G and a maximum weight matching M of size k one can find a maximum weight augmenting path for G in $O(n^2m)$ time, where n is the number of vertices of G and m is the number of edges.*

We need to apply this algorithm $n/2$ times at most, as such, we get:

Theorem 3.3 *Given a weight bipartite graph G , with n vertices and m edges, one can compute a maximum weight matching in G in $O(n^2m)$ time.*

3.1 Faster Algorithm

It turns out, in fact, that the graph here is very special, and one can use the Dijkstra algorithm. We omit any further details, and just state the result. The interested student can figure out the details (warning: this is not easy).

Theorem 3.4 *Given a weight bipartite graph G , with n vertices and m edges, one can compute a maximum weight matching in G in $O(n(n \log n + m))$ time.*