IEOR 8100: Matchings | Sep 20, 2012

Hopcroft-Karp Bipartite Matching Algorithm and Hall's Theorem

Instructor: *Cliff Stein* | Scribe: *Chun Ye*

1 Hopcroft-Karp Algorithm

Recall that the basic bipartite matching algorithm repeatedly finds an augmenting path and performs the operation $M \bigoplus E(P)$, where P is the augmenting path found at each iteration, until the graph has no more augmenting paths. The running time of the algorithm is $O(mn)$, as an augmenting paths can be found by doing a breath first search and there are at most $\frac{n}{2}$ augmenting paths with respect to the empty matching in any graph. In this lecture, we study a faster bipartite matching algorithm originally proposed by John Hopcraft and Richard Karp in 1973. Their algorithm runs in $O(\sqrt{n}m)$ time.

Consider the following subroutine of the algorithm.

1: Start with a non-maximum matching M.
2: Find $\{P_1, \ldots, P_k\}$ a maximal set of vertex disjoint shortest augmenting paths with respect to M.
3: Set $M = M \bigoplus (E(P_1) \cup E(P_2) \cup \ldots \cup E(P_k))$.

Proposition 1. *The above subroutine correctly computes a matching with a larger size.*

Proof. The suffices to show that for all $2 \leq i \leq k$, P_i is an augmenting path with respect to the matching $M \bigoplus E(P_1) \bigoplus E(P_2) \ldots \bigoplus E(P_{i-1})$. This is because since $P_1, P_2 \ldots P_i$ are vertex disjoint, performing $M \bigoplus E(P_1) \bigoplus E(P_2) \ldots \bigoplus E(P_{i-1})$ does not modify the part of the graph where P_i augments the matching M. $\square$

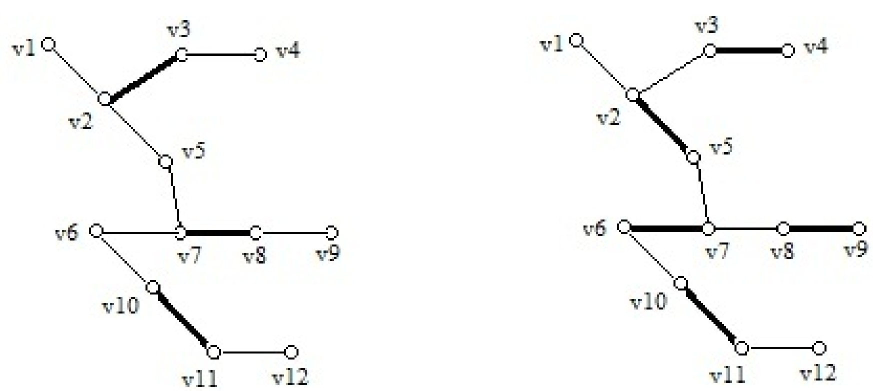
We can find a maximal set of vertex disjoint shortest augmenting paths with respect to M in $O(m)$ time. To do so, we first perform a modified breath first search: let $G(A, B)$ be the underlying bipartite graph, let $S \subseteq A$ be the set of unmatched vertices in A. The algorithm does a simultaneous breath first search starting at every vertex $v \in S$ that alternates between non-matching and matching edges. One can do so by adding a source node, having an edge between the source and each vertex in S, and running a BFS from the source. We stop the BFS at the k$^{\text{th}}$ level where $k + 1$ is the smallest distance from the source to where we hit a free vertex in B. Let X denote the set of all unmatched vertices in B that we found at the k$^{\text{th}}$ level. We then perform the following greedy algorithm: for every vertex in $u \in X$, we trace back along its predecessor vertices until we hit a vertex $v \in S$. If v is unmarked, then we record the path betweeen u and v and mark the vertex v as taken.

Next, we prove the following proposition, which allows us to meaure how much progress we have made after one iteration of the subroutine.

Proposition 2. *Let l be the length of a shortest augmenting path with respect to M. Let $P_1, \ldots, P_k$ be a maximal set of vertex disjoint shortest augmenting paths. Let $M' = M \bigoplus (E(P_1) \cup \ldots \cup E(P_k))$. Let P be a shortest augmenting path with respect to M', then $|P| > l$.*

Proof. We consider two cases.
Case 1: P is vertex disjoint from $P_1, \ldots, P_k$.
It is clear that here $|P| > l$, otherwise it would contradict the fact that $\{P_1, \ldots, P_k\}$ is a maximal set of vertex disjoint shortest augmenting paths.

Case 2: P is not vertex disjoint from $P_1, \ldots, P_k$. We present two proofs. Here is the first proof. Let $P = p_1 p_2 \ldots p_m$. Since P is not vertex disjoint from $P_1, \ldots, P_k$, it must share a M' matching edge with some path $P_i = p_1^i p_2^i \ldots p_l^i$. Let (u, v) be such an edge with u having a smaller index than v in P. Let R_1 denote the subpath of P from p_1 to u and R_2 denote the subpath of P from v to p_m. Let Q_1 denote the subpath of P_i from p_1^i to v and Q_2 denote the subpath of P_i from u to p_l^i. Notice that Q_1 and Q_2 are alternating paths with respect to M. Although R_1 and R_2 are alternating paths with respect to M', they are not necessarily alternating paths with respect to M. If R_1 is an alternating path w.r.t to M, then $R_1 \cup Q_2$ is an augmenting path w.r.t. M. If R_1 is not an alternating path w.r.t. M, then in order for it to become an alternating path w.r.t. M', it must share a M nonmatching edge with some P_j and once we augment the path P_j, we change the nonmatching edge to a matching edge. Consequently, in this case, R_1 must contain at least one free vertex w.r.t M (that is not p_1). Let r denote the free vertex closest to u and let P' be the path from r to u, then $P' \cup Q_2$ is an augmenting path w.r.t M. Similarly, we can always find a subpath Q' of R_2 with v as one of the endpoints such that $Q_1 \cup Q'$ is an augmenting path w.r.t M. Since $P' \cup Q_2$ and $Q_1 \cup Q'$ are augmenting paths w.r.t M, we have that $|P' \cup Q_2| = |P'| + |Q_2| \geq l$ and $|Q_1 \cup Q'| = |Q_1| + |Q'| \geq l$. On the other hand, since P_i is a shortest augmenting path w.r.t. M, we have that $|Q_1| + |Q_2| = l - 1$. Consequently, we get that $|P'| + |Q'| \geq l + 1$, which implies that $|P| = |R_1| + |R_2| + 1 \geq |P'| + |Q'| + 1 \geq l + 2$. The diagram below illustrates an example of the argument



In this example, (u, v) = (v7, v6), R1 = v1 v2 v5 v7, R2 = v6 v10 v11 v12, Q1 = v6 v10 v11 v12, Q1 = v6, Q2 = v7, v8, v9. The diagram on the left represents the graph w.r.t the matching M. The diagram on the right represents the graph w.r.t. the matching M'. To get from M to M', we augmented along the paths Qj = v5 v2 v3 v4 and Qi = v6 v7 v8 v9. Notice that R1 is not an alternating path w.r.t M, but becomes an alternating path w.r.t. M'. P' = v5 v7 is the subpath of R1 such that P' U Q2 is an augmenting path w.r.t. M and Q' = v6 v10 v11 v12 is the subpath of R2 such that Q1 U Q' is an augmenting path w.r.t. M

above.

Here is the alternate proof. Since $M' = M \bigoplus (E(P_1) \cup \ldots \cup E(P_k))$, we have that $M \bigoplus M' = (E(P_1) \cup \ldots \cup$

$E(P_k)$), which implies that $M \bigoplus M' \bigoplus E(P) = (E(P_1) \cup \ldots \cup E(P_k) \bigoplus E(P)$. Consider the connected components of a subgraph H of G whose edge set is $M \bigoplus (M' \bigoplus E(P))$. Since $|M' \bigoplus E(P)| - |M| = k+1$, there are at least $k+1$ components of H that uses more edges from $M' \bigoplus E(P)$ than it uses edges from M. Each of these components corresponds to an augmenting path with respect to M. Consequently, H contains at least $k+1$ vertex disjoint augmenting paths with respect to M, each of which has length at least l. Hence, we conclude that $|M \bigoplus M' \bigoplus E(P)| = |(E(P_1) \cup \ldots \cup E(P_k)) \bigoplus E(P)| \geq (k+1)l$. Since $P_1, \ldots, P_k$ are vertex disjoint, they contribute at least kl distinct edges, which means that P must contribute at least l edges of its own in order for the inequality $|(E(P_1) \cup \ldots \cup E(P_k)) \bigoplus E(P)| \geq (k+1)l$ to hold. Now, since P is not vertex disjoint from $P_1, \ldots, P_k$, it must share a matching edge with some path P_i with respect to the matching M'. Consequently, we may conclude that $|P| > l$. □

Now we may consider the main algorithm.

Algorithm 1 Hopcroft-Karp Algorithm

```
Start with M = φ.
while M is not a maximum matching do
    Find {P_1, ..., P_k} a maximal set of vertex disjoint shortest augmenting paths with respect to M.
    Set M = M ⊕(E(P_1) ∪ E(P_2) ∪ ... ∪ E(P_k)).
end while
```

Proposition 3. *The Hopcroft-Karp Algorithm runs in $O(\sqrt{n}m)$ time.*

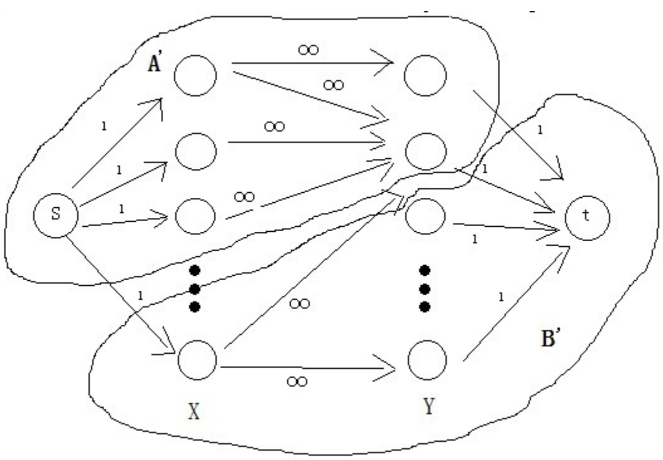
Proof. Since each iteration of the algorithm takes $O(m)$ as argued before, it suffices to show that the algorithm terminates after $O(\sqrt{n})$ iterations. After $\sqrt{n}$ iterations, either the algorithm has terminated because we have found a maximum matching, or we have obtained a matching M where the shortest augmenting path with respect to M has length at least $\sqrt{n}+1$. Let M' be a maximum matching of G, then a subgraph H of G whose edge set is $M \bigoplus M'$ can be decomposed into components containing at least $|M'| - |M|$ vertex disjoint augmenting paths with respect to M. Since each of those augmenting paths has length at least $\sqrt{n}+1$ and $E(H) \leq n$, we get that $|M'| - |M| \leq \frac{n}{(\sqrt{n}+1)} < \sqrt{n}$. Hence, after another $\sqrt{n}$ iterations, the algorithm is guaranteed to find a maximum matching if it hasn't already terminated after the first $\sqrt{n}$ iterations. □

2 Hall's Theorem

Theorem 4. *Given a bipartite graph $G(X,Y)$ where $|X| = |Y|$. G has a perfect matching if and only if for all $A \subseteq X$, $|\delta(A)| \geq |A|$.*

Proof. $\longrightarrow$ If there exists $A \subseteq X$ such that $|\delta(A)| < |A|$, then one cannot match all the vertices in A, which means that G cannot have a perfect matching.
$\longleftarrow$ We prove the contrapositive using flows. First we add a source node s and a sink node t to the graph G and draw an edge from s to each of the vertices in X with capacity 1 and draw an edge from each of the vertices in Y to t with capacity 1. We set the capacities of the edges from X to Y to infinity. Then we run a max flow algorithm to compute a max flow of the modified network. It is clear from the set up that any max flow of the network corresponds to a maximum matching of G. After we have found a max flow f, let A' be a set of vertices that is reachable from s in the residual network with respect to f and

let $B' = G \backslash A'$, then from the max-flow min-cut theorem, we have that (A', B') forms a min cut.



Now, assume G has no perfect matching, then $|f| < n$, which means that $cap(A', B') < n$. We claim that $cap(A', B') = |X \cap B'| + |Y \cap A'|$. This is because there is no edge crossing the min cut from X to Y in the residual network since an edge from X to Y has infinite capacity. Hence, all edges crossing the cut from X to Y are either from s to some vertex of X not reachable from s in the residual network, or from some vertex of Y reachable from s in the residual network to t. The size of the first set of edges is $|X \cap B'|$ and that of the second is $|Y \cap A'|$. Let $A = X \cap A'$, then $|X \cap B'| = n - |A|$. Notice that there can be no edges from $X \cap A'$ to $Y \cap B'$ because we cannot allow any infinite capacity edge to cross the min cut from X to Y (otherwise the cut capacity would be infinity). Hence, we may conclude that $\delta(A) \subseteq Y \cap A'$. Putting it altogether, we have that $n > cap(A', B') = |X \cap B'| + |Y \cap A'| \geq n - |A| + |\delta(A)|$, which means that $|A| > |\delta(A)|$. Hence, we have explicitly found a $A \subseteq X$ where $|A| > |\delta(A)|$ if G has no perfect matching. $\square$