CS 435 : Linear Optimization Fall 2006

Lecture 19: Nonbipartite Matching

Lecturer: *Sundar Vishwanathan* Scribe: *Neha Singh*
Computer Science & Engineering Indian Institute of Technology, Bombay

1 Recap

In last class we had seen how to find the matching of maximum size in a bipartite graph. This is done by starting with any single edge as a matching and then iteratively finding augmenting paths and increasing the size of the matching by one at each step until there is no augmenting path in the graph w.r.t that matching. We had proved the following theorems last time:

Theorem 1 *The matching M of the graph G is maximum iff there is no augmenting path in G w.r.t M.*

Theorem 2 *For bipartite graphs, if there exists an augmenting path in the graph G then the we can find it by using the modified bfs algorithm.*

Now let us proceed to finding maximum matching in a non-bipartite graph.

2 Nonbipartite Matching

Till now we have seen how to find the maximum matching for a bipartite graph. The same algorithm for finding an augmenting path in a bipartite graph will be extended for non-bipartite graphs. However modifications need to be made to the algorithm.
As before, we start with an unmatched even vertex, find all its neighboring odd vertices. If any of these odd vertices were unmatched then we have found an augmenting path else we continue from these odd vertices by following the matched edges incident on them. This yields a new set of even vertices and so on. In case of the a bipartite graph edges can be only between even and odd vertices. Thus while following the path, we keep alternating between even and odd vertices. However in the case of a non-bipartite graph, we can have odd-odd and even-even edges also which will give rise to odd circuits in the graph. We consider the problems arising due to even-even edges only, as in the modified bfs algorithm, after reaching an odd vertex, we follow the matched edge and there can be only one matched edge incident on a vertex.

For example consider the figure below. Here there is an even-even edge between v_1 and v_2. We observe that v_1 and v_2 have a common ancestor which has to be necessarily even. This is because in our modified bfs algorithm, we branch off at only the even vertices. At odd vertices, we only follow the matched edge. If we come across an even-even edge between two vertices while searching for augmenting path, the graph must have an odd cycle where the even-even edge is a part of the odd cycle. In the modified algorithm for finding augmenting paths in non-bipartite graphs, whenever we come across such an odd cycle, we shrink it to a single vertex (say v_o) such that all the vertices adjacent to any of the vertex in the cycle are now adjacent to v_o. Such an odd length cycle having a matched and unmatched edges incident on all but one vertex (the base where both incident edges are unmatched) is called a blossom.

Definition 1 *A blossom is a cycle of length 2k+1 with k edges matched . The base of the blossom is the one and the only vertex in the cycle with both incident edges unmatched.*

Naturally for shrinking to represent a valid operation while searching for augmenting path, we must show that by shrinking a blossom does not add or omit augumenting paths in the graph. The exact method and the proof of correctness will be discussed later. First we prove the following lemma.

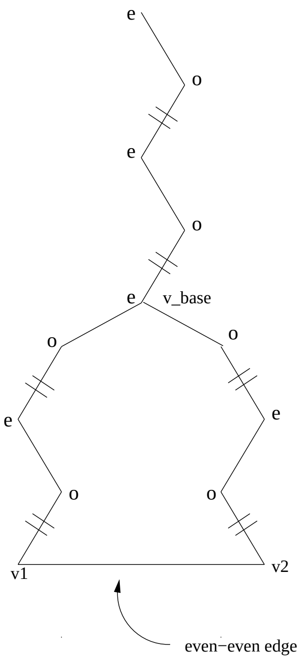

Figure 1: Blossom

LEMMA 3 *Let M be a matching. Let B be a blossom such that the base is unmatched in G. Let $G^{‘}$ be the graph obtained by shrinking B. Let $M^{‘}$ be the edges of M outside B. There is an augumenting path in G w.r.t M iff there is an augumenting path in $G^{‘}$ w.r.t $M^{‘}$.*

PROOF: *Part 1:* ($\Rightarrow$) Given an augmenting path in G w.r.t M to show that there exist an augmenting path in $G^{‘}$ w.r.t $M^{‘}$.

1. Case (i): The augmenting path in G does not intersect the blossom. So the same augmenting path exists in $G^{‘}$ w.r.t $M^{‘}$.
2. Case (ii): The augmenting path intersects the cycle in G.
   Claim: We can find an augumenting path in $G^{‘}$ if we stop as soon as we hit the shrunk vertex(say v_o) while travelling along the augmenting path of G.
   Proof: This is because of the fact that v_o in unmatched in $G^{‘}$. Note that all the vertices in the blossom except the base already have an incident matching edge inside the blossom and the base is assumed to be unmatched.

Part 2:($\Leftarrow$) Given an augmenting path in $G^{‘}$ w.r.t $M^{‘}$ to show that there exist an augmenting path in G w.r.t M.

1. Case (i): The augmenting path in $G^{‘}$ does not contain v_o. Then the same path is an augmenting path in G also.
2. Case (ii): The augmenting path contain v_o. Since v_o is an unmatched vertex in $G^{‘}$, the augmenting path can either start of end at v_o.

   On expanding the blossom, this path will intersect the blossom in the graph G. Thus the path will contain some edge incident on some vertex of the blossom. The vertex through which the path enters the blossom can be either the base or any other vertex of the blossom.

(a) If the path contains the edge incident on the base of the blossom then we have found an augmenting path in G as the base in unmatched.

(b) If the path contains edge incident on any other vertex of the cycle, i.e., other than the base, then extend this path by moving along the cycle to reach the unmatched base and we will have an augmenting path in G.

□

We note that in the figure above the base is unmatched only when the common ancestor of v_1 and v_2 is the vertex from which we started searching for an augmenting path. To use this lemma, we need to do something more once we identify a blossom. This we will see in the next lecture.