CS 435 : Linear Optimization — Fall 2006

Lecture 18: Review of graph algorithms (contd.)

Lecturer: *Sundar Vishwanathan* — Scribe: *Vipul Shingde*
Computer Science & Engineering — Indian Institute of Technology, Bombay

1 Recap of previous lecture

In previous lectures we studied following definitions and theorem:

Definition 1 *A path is alternating w.r.t. a matching M if its edges alternate between those in M and those not in M.*

Definition 2 *An augmenting path is an alternating path that starts from and ends on free (unmatched) vertices.*

Theorem 1 *A matching M is maximum if and only if it does not contain any augmenting path.*

Algorithm for maximum matching: We start with a single edge in the matching M. In each iteration we first find an augmenting path P. Then we exchange the matched and unmatched edges in P to get a matching of one greater size. We keep doing this till a matching with no augmenting path is found. Theorem 1 guarantees that this matching is maximum.

Modified BFS Algorithm
An augmenting path in G w.r.t. a matching M, is found using the following algorithm:

Let L be set of unmatched vertices in G w.r.t. M.
For each $u \in L$
1. Start BFS with root node u.
2. Till BFS is complete
   Let BFS be currently at vertex v
   if v is at even depth
      The children of v in the BFS tree will be unvisited vertices along unmatched edges incident on v.
   else
      if v is unmatched
         Augmenting path $u \rightarrow v$ found.
      else
         The child of v in the BFS tree will be an unvisited vertex along the matched edge incident on v.

2 Proof of correctness of modified BFS algorithm

If there exists an augmenting path P from u, then applying modified BFS with u as root node must find an augmenting path for the algorithm to be correct. In fact, the given alogrithm finds the shortest augmenting path P from u.
Let $\{u_0, u_1, ...u_{k-1}\}$ be the order of vertices in a shortest augmenting path P, with $u = u_0$ as one end-point.

Consider the following lemma

Lemma 2 *The vertex u_i will appear in the i^{th} level of the BFS tree.*

PROOF: By induction on i.

Base Case: $i = 0$
Trivially true from the algorithm.

Inductive Step: Assuming the induction hypothesis holds for $i = l$, consider $i = l + 1$.
if $l = 2 * m + 1$
Modified BFS algorithm will mark one matching edge(if any) incident on each vertex in even levels. Since (u_l, u_{l+1}) is a matched edge, u_{l+1} will appear in level $l + 1$ unless it has already appeared. If it has already appeared, then we can easily show that P will not be the shortest augmenting path.
else if $l = 2 * m$
Modified BFS Algorithm will mark all unmatched edges from each vertex in level $2 * m$, and as all alternating paths of length 2*m have been marked $\Rightarrow$ all alternating paths starting from length $l = 2 * m + 1$ are marked.

□

3 Time Complexity

Consider graph $G(V, E)$ and let $|V| = n$ and $|E| = m$.
Time complexity = Number of iterations times the time to find augmenting path in a matching.
Since the matching increases in size in every iteration, the number of iteration is bounded by n. In each iteration we may have to perform $O(n)$ searches (from unmatched vertices) before finding an augmenting path. A single run of modified BFS takes $O(m)$ time.

Hence time complexity = $O(m * n^2)$.

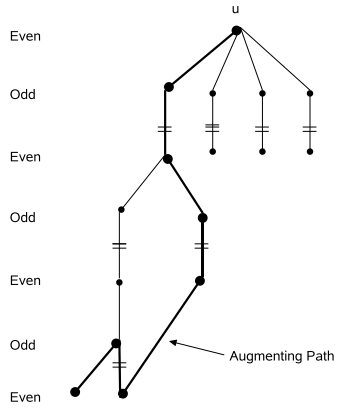

Figure 1: Modified BFS on General graph

[1]With some modifications a better algorithm with run-time of $O(n^{3/2} * m)$ can be obtained

4 Extending the algorithm for general graphs

The above modified BFS algorithm works correctly for bipartite graphs. For general graphs(which can contain odd cycles), the algorithm needs to be modified. Non-bipartite graphs can contain even-even edges(as shown in the figure above). Hence if we apply modified BFS on non-bipartite graphs, it might be possible that the alogrithm might not find an augmenting path. Such an example is shown in the Fig. 1. The augmenting path is shown in dark.
A solution to this problem is to shrink the odd-cycles to a vertex. This approach will be discussed in detail in next lecture.