Algorithms Assignment 2

Abhey Arora, BS-1618

28th February 2018

1 Correctness Proof for MPM algorithm

Before writing its correctness I will first write MPM's algorithm.

Algorithm 1 MPM Algorithm

Input:A network graph G with capacities $u(e)$ for each edge.
Output:Maximum flow in the graph G from s(source) to t(sink)
Initialization:Initialize with $f(e) = 0$ for each edge

while s and t are connected in Residual Graph G_f **do**
  Compute the layered network L_f
  while s and t are connected in Layered graph L_f **do**
    Compute potential for each vertex in L_f.
    Add vertex to P if it has least potential yet.
    Let m be the potential of all vertices in P.
    Take all vertices in P one by one and let u be the current vertex in P.
    Push m units of flow from u to t and update the residual capacities.
    Pull m units of flow from s to u and update the residual capacities.
    Delete all vertices that are not on any path from s to t and all their incoming and outgoing edges in L_f. This clearly involves all vertices of U.
  end while
  Update flow variable and update residual graph
end while

As, we are performing the inner loop till the time s and t are connected in the current layered graph we are saturating at least one edge from each path from s to t. This is what we call a ***blocking flow***. As, we are computing blocking flows till the time s and t are disconnected in the Residual Graph we are finishing the existence of augmenting paths in the residual graph. This means that if our algorithm stops it will stop at a flow which is maximum due to the Theorem that **A flow is maximum if and only if there is no augmenting path from s to t.**

Now, we will prove that our algorithm stops. As each iteration of inner loop removes at least one vertex with the minimum potential. So, each time we go inside the inner loop we can stay inside for at most n iterations. Now, let $d(f)$ be the number of hops it takes for the shortest path from s to t in the current residual graph. Now, we claim that the value of $d(f)$ increases each time one inside loop finishes. First of all, as only backward edges are introduced, they can only increase the length of paths. So, if we have to have a path of length $d(f)$ in current residual graph it should be made only of edges originally in residual graph. This is not possible because we augment all paths present in the original residual graph whose lengths were $d(f)$ or more than that connecting s and t. So, no such path exists anymore. As, the largest value of $d(f)$ is n. This can go on for $O(n)$ iterations. So, in total worst case iterations of inner loop are $O(n^2)$, n times for each outer loop.

Therefore, the algorithm halts and halts with the correct answer.

2 Complexity Analysis for MPM algorithm

We will calculate the complexity of inner loop, i.e., time taken to calculate a blocking flow and multiply it by n because that is the maximum number of iterations of that. Since at any iteration all the incoming or the outgoing arcs at the reference node will become saturated there can be at most n iterations.
If information about incoming and outgoing arcs at a node is kept in linked lists then the amount of effort in distributing the flow during the i^{th} iteration is $O(|V_i| + |E_i|)$, where $|E_i|$ is the number of arcs deleted and $|V_i|$ is the number of arcs receiving extra flow but not saturated which in worst case is $i - 1$. Therfore, the total effort is

$$O(\sum_i (i - 1 + |E_i|)) = O(n^2 + m) = O(n^2)$$

This is because sum of $|E_i|$'s is at most m and i goes from 1 to n in the worst case.
It takes us $O(n^2)$ to compute a blocking flow. Therefore, it takes us $O(n^3)$ to compute the maximum flow.