Second Level Algorithms

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 07

Lecture 32

Welcome to the 32nd lecture of second level algorithms course. In this lecture, we will see an important application of maximum A steep flow problem to compute ah maximum cardinality bipartite matching in a ah bipartite graph ok. So, let us begin. So, let us recall the problem statement of maximum bipartite matching. input is a bipartite graph G equal to V with the bipartition L union R and E the output should be maximum cardinality or size matching ok. So, we will reduce any instance of maximum bipartite matching to an instance of maximum s-t flow problem such that we will be able to compute a maximum cardinality bipartite matching of G from an optimal s-t flow. So, let us see how the equivalent instance of maximum s-t flow problem look like. So, suppose this is the input bipartite graph of maximum bipartite matching problem, this is L, this is R, both L and R are independent sets and edges only go from L to R, edges go between L and R. To construct an equivalent instance of maximum s-t flow problem, we orient the graphs, orient the edges from left to right.

In the maximum bipartite matching problem, the edges are undirected. In the equivalent maximum s-t flow problem, we direct the edges from L to R. We add a source vertex S, a sink vertex T, we add a directed edge from S to every vertex in L and we add a directed edge from every vertex of R to T. So, this is the graph of the maximum s-t flow problem. We need to define the capacity of every edge.

We define the capacity of every edge to be 1. So, how do you construct the equivalent instance? Let us write it concretely. We direct all from L to R, we add a source vertex s and a sink vertex t. We add a directed edge from S to every vertex of L. We add a directed edge from every vertex of R to T. So, these are the set of vertices and edges of the graph G, let us call it G'. We define the capacity of every edge to be 1.

So, this is the reduced instance of maximum ST flow we compute a maximum s t flow, let us call it f. in polynomial time using either Edmund Karp. or Dinic's or push-relabel algorithm. even take it as a easy exercise to prove that for the reduced instance even Ford Fulkerson method will also output the maximum s-t flow in polynomial time.

We can assume without loss of generality that f is an integral s-t flow. since the capacity of every age is an integer and in this case a maximum s-t flow computed by any of the above algorithms is an integral flow.

This we have discussed when we studied the maximum flow problem and these algorithms. Moreover, since the capacity of every edge in G prime let us call this graph $G^{'}$ is 1. the flow value f_e of any edge e is either 0 or 1.

So, here is a claim. Let f be the set of edges from L to R that carries flow of value 1 in f, then the set of undirected edges corresponding to F is a maximum cardinality matching of G, ok. So, pictorially, this is L, this is R, edges are going across. So, we say that find those edges from L to R which carry 1 unit of flow. Some edges will carry 1 unit of flow; some edges will carry 0 units of flow. So, let us look at the set of edges from L to R that carry 1 unit of flow and ignore the direction of those edges.

That set of edges is a maximum cardinality matching in the bipartite graph. That is the claim. So, let us prove it. So, we are given a maximum s-t flow f, and it is an integral flow. We observe, or we will prove, that no two edges in f can have the same endpoint in L.

Indeed, this must be the case; otherwise, there exists a v in L such that at least two edges in f are incident on v. Then, the flow out at v is at least 2, since every edge in f carries 1 unit of flow. However, the flow in at v can be at most 1, since the in-degree of v is 1, and the capacity of every edge of $G^{'}$ is 1. This violates the conservation of flow property at V, OK. So, hence, no two edges in F can be incident on the same vertex in L. Similarly,

No two edges of f can be incident on a single vertex in R. Hence, the set of undirected edges corresponding to F forms a matching in G. Now, we will prove that it is a maximum cardinality matching. If it is not a maximum cardinality matching in G, then let T be a matching.

In G whose size is more than f. Let E be the set of edges in T directed from L to R. Consider the S-T flow f_1 defined as f_1 of E is 1 for every edge e in A and f_1 of E is 1 for all edges E that share an endpoint with any edge in A. The flow of every other edge is 0.

It can be checked easily that f is a valid And the value of f, which is the cardinality of T, which is strictly more than the cardinality of capital F, which is same as the value of F. But this contradicts our assumption that F is a maximum s-t flow. So, the matching defined by the underlying undirected edges of F should be a maximum matching of G. So, this concludes the proof of equivalence, OK.

So, let us stop here. Thank you.