Second Level Algorithms

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 06

Lecture 27

Welcome to the 27-th lecture of the second-level algorithms course. We have been studying the proof of correctness of the push-relabel algorithm. We have already shown that the algorithm maintains the loop invariant, and hence, if the algorithm terminates, then it outputs a maximum s-t flow. In the last lecture, we proved a key lemma which shows that for every vertex v that has a positive excess flow, there must be a path in G from s to that vertex using only those edges which carry a positive amount of flow. Using that key lemma, we will see today how we can bound the number of relabel and push operations.

So, let us begin. The first observation or corollary from the lemma that we have proved is a bound on the height of any vertex. So, here is the corollary. The height of every vertex is at most twice the number of vertices minus one. Consider any vertex v on which we perform a relabel operation.

Recall that we perform a relabel operation only when a vertex has a positive excess flow. So, if we perform a relabel operation on v, at that point in the execution of the algorithm, v must have an excess flow. We must have an excess when we perform a relabel operation on v. Now, we use the lemma that we proved in the last lecture: that for every vertex which has a positive excess, there must be a path from s to v in the graph where every edge carries a positive amount of flow, or equivalently, there must be a v to s path.

In the residual graph. From the lemma, we conclude that there are must be a path from v to s in the residual graph. The length of the path can be at most $n-1$. Since the algorithm maintains the invariant, the height of s is n. Let the path be v, then $u_1, u_2, \ldots, u_k$, which is s. and the length of the path is k, so k is less than or equal to $n-1$. Okay, so the height of s is n, so the height of u_{k-1} is at most $n+1$ because edges can go either in the same height or uphill, or if it goes downhill, it can go down by at most one height. So, because there is an edge from u_{k-1} to u_k and the height of u_k is n, the height of u_{k-1} is at most $n+1$. This implies that the height of u_{k-2} is at most $n+2$.

This implies the height of u_1 is less than or equal to $n+k-1$, which implies that the height of v is at most $n+k$. k is at most $n-1$. So, this is at most $2n-1$. So, whenever we perform a reliable operation on v, the height of v is at most $2n-1$, and hence—or let me write—whenever we perform a relabel operation on v, the height of V must be at most 2n minus 1. Hence, the height of V after a reliable operation is at most 2n. So, this is what we have to show in the corollary. This bounds the number of reliable operations.

So, this shows that the corollary the total number of reliable operations in any execution of a push-relabel algorithm is at most 2n^2 proof. Every reliable operation increases the height of the vertex on which we have performed the reliable operation by 1. Hence, the total number of reliable operations performed on any vertex is at most twice. We can perform reliable operations on any vertex other than s and t. Or let us write this way: the reliable operation is never performed on s and t. Hence, the total number of reliable operations throughout the run of push reliable algorithm is at most $2n\times(n-2)$. which is at most less than or strictly less than $2n^2$, which proves the corollary. So, the number of free label operations is at most $2n^2$. Next, we prove that the number of push operations is at most n cube. So, to prove this we will divide push operations into two types.

One is saturating push. Another is non-saturating push. Saturating push are those push operations where the amount of excess flow pushed is the residual capacity of the edge along which the push operation is performed. In the non-saturating push the amount of excess flow pushed is strictly less than the capacity of the edge along which the push operation is performed. we will show that the number of saturating push and the number of non-saturating push both of them is $O(n^3)$ thereby proving this theorem.

So, let us first bound the number of saturating pushes. the total number of saturating pushes throughout the run of the push reliable algorithm is $O(mn)$ which is $O(n^3)$. Proof.

The crucial thing is to bound the number of times we can perform a saturating push along any edge. So, consider any edge e equal to uv in any residual graph. First, observe that the saturating push along the edge e makes edge e disappear in the residual graph of the next iteration.

Observe that a saturating push along the edge e makes e disappear from the residual graph of the next iteration. To perform another saturating push along the edge e, e equal to uv, the edge e must reappear in the residual graph, which happens only when we push flow along vu, the edge in the opposite direction. However, we push flow along downhill edges only. So, between the disappearance and subsequent reappearance, the height of v must increase by at least two. Why? Because, consider the first time I perform a saturating push from u to v, the height of v must be 1 less than the height of u. Now, I need to push some flow from v to u, and push only happens in the downhill edge. So, the height of v must cross the height of u, which means it must increase by at least 2. Hence, the number of saturating pushes along any edge of the residual graph is at most n, because, as we have already seen, the height of any vertex cannot be more than twice n. So, the total number of saturating pushes is at most the number of edges in the residual graph is at most twice the number of edges of the input graph. Hence, twice m, and for every edge, the number of saturating pushes can be at most n. So, twice m n, which is big $O(mn)$, okay. So, this bounds the number of saturating pushes.

So, let us stop here. In the next lecture, we will bound the number of non-saturating pushes. Thank you.