

The complexity of Edmonds-Karp

Course notes for Search and Optimization

Spring 2006

Peter Bro Miltersen

February 6, 2006

Version 3.0

The theorem below may replace Lemma 26.8 and Theorem 26.9 of Cormen *et al* for the purpose of analyzing the complexity of the Edmonds-Karp algorithm for the Max Flow Problem.

Theorem As f changes throughout the execution of the Edmonds-Karp algorithm, so does the residual network G_f . This sequence of residual networks satisfies the following properties:

1. The distance between s and t in G_f *never decreases*. After each iteration of the while-loop (i.e., after each augmentations), it either increases or stays the same.
2. The distance between s and t in G_f can stay the same for at most $|E|$ iterations of the while-loop (i.e., $|E|$ augmentations) before increasing.

Before we prove the theorem, let us note the consequence for the complexity of Edmonds-Karp: As the distance between s and t can never be more than $|V| - 1$ and it starts out as at least 1, it follows from the theorem that there can be at most $O(|V||E|)$ iterations of the while-loop when executing Edmonds-Karp. As each iteration can be performed in time $O(|E|)$, the time complexity of Edmonds-Karp is $O(|E|^2|V|)$.

Proof We prove the theorem by considering a residual network G_f during the execution of Edmonds-Karp that is either the very first residual network (i.e., $f \equiv 0$, so G_f is actually equal to the original network G) or a residual network where the distance between s and t has just increased, i.e., the distance is now strictly bigger than in the previous residual network. We shall keep track of the residual networks G_f that occur in the executing of Edmonds-Karp for the next few iterations of the while-loop and show that the distance between s and t will not decrease and that after at most $|E|$ iterations, it will strictly increase.

We shall define a *labeling* of the vertices V that we shall keep fixed throughout the proof: We label each vertex v with the *length of the shortest path from s to v* in the network G_f we start out with. Let the label of v be denoted $\text{level}(v)$. The network G_f will change throughout the argument as we perform new iterations of the while-loop, but we shall *not* change the labelling: It will remain the same. Thus, the labels will not retain their original semantics, but we shall argue that the following *invariant* will be maintained anyway:

Invariant: All arcs of G_f are either arcs (u, v) with $\text{level}(v) = \text{level}(u) + 1$ (we shall refer to these arcs as *good arcs*) or arcs (u, v) with $\text{level}(v) \leq \text{level}(u)$. We shall refer to these arcs as *bad arcs*.

Thus, there will be no arcs “skipping a level”, i.e., there will be no arcs (u, v) with $\text{level}(v) > \text{level}(u) + 1$. Clearly, this property is true when we start where the labels denote the lengths of shortest paths from s to the vertices.

If the shortest path from s to t has length d when we start out, we have that $\text{level}(t) = d$. Note that as long as the invariant is maintained, there can be no path from s to t of length strictly smaller than d (and property 1. of the theorem we are trying to prove follows), since such a path would have to “skip a level” and *if* the shortest path from s to t is still d , no shortest path from s to t can contain bad arcs, for the same reason: The bad arcs are “wasting time” and we would have to make up for this wasted time by using an arc skipping a level.

Now perform an iteration of Edmonds-Karp and let us look at what happens to G_f . The augmenting path we choose must be a shortest path from s to t , i.e., it contains only good arcs. In the next residual network, at least one good arc has disappeared, namely the arc(s) with the minimum capacity $c_f(e)$ among the arcs e on the augmenting path. Furthermore, for each arc (u, v) on the path, the reverse arc (v, u) will be present in the next G_f (it may or may not be present in the old G_f). These are the only differences between the old G_f and the new G_f . Note that the arcs (v, u) added do not violate the invariant and are in fact bad arcs, as they all go from one level to the previous one, as they are all the reverse of some good arc.

Thus, after one iteration of Edmonds-Karp we still have the invariant and the distance from s to t has not decreased. Now, if it has in fact *increased*, we have proved the theorem. Otherwise we perform another augmentation and the argument above ensures us that the invariant is still maintained and the distance from s to t has not decreased. If it has *increased*, we are happy, and otherwise we perform another iteration, still maintaining the invariant. And so on and so forth.

How many iterations can we perform before the distance increases? Each iteration *makes at least one good arc disappear*. This good arc can never come back, as we only add bad arcs while the invariant is maintained, as explained above. When we start out, there are at most $|E|$ good arcs. Thus, we can have at most $|E|$ augmentations¹ before the distance from s to t becomes strictly bigger than d . This completes the proof.

¹actually, even a bit fewer, as we need d good arcs to have a path of length d from s to t .