

CSC 373 H 1 Y — Summer 2006

University of Toronto — St. George Campus

Lecture Summary for Week 9

This summary is not a replacement for the lecture. If you miss a class, please arrange with a friend to take note for you.

Tutorial this week: Assignment 2 solution.

References for week 8 & week 9 lectures: Sections 7.1-6

Extra reading: Sections 7.8, 7.9, 7.11, 7.12.

Residual Network

The idea is to improve the flow along some st -path.

Definition Given a flow f in a flow network G , the *residual network* of f , denoted by G_f , is the flow network with the same set of nodes as G , whose edges and capacities are as follows:

- For each edge e in G , if $f(e) < c(e)$ then e is in G_f with capacity $c'(e) = c(e) - f(e)$ (the “leftover” capacity, e is also called a *forward* edge).
- For each edge $e = (u, v)$ in G , if $f(e) > 0$ then (v, u) is in G_f with capacity $f(e)$. ((v, u) is also called a *backward* edge.)

Notice that if $e = (u, v)$ in G with $0 < f(e) < c(e)$, then both e and (v, u) are in G_f .

Augmenting paths An st -path in G_f is also called an *augmenting path*. The amount of flow that can be increased along an augmenting path P is denoted by $\text{bottleneck}(P)$:

$$\text{bottleneck}(P) = \min\{c'(e) : e \text{ in } P\}$$

The following algorithm output the flow that improves on the flow f by the amount of $\text{bottleneck}(P)$ along the path P :

Augment(f, P)

1. $f' \leftarrow f$ % f' is the output flow.
2. $b \rightarrow \text{bottleneck}(f, P)$
3. For each $e = (u, v)$ in P do
4. If e is a forward edge do
5. $f'(e) \leftarrow f(e) + b$
6. Else % e is a backward edge
7. $f'(v, u) \leftarrow f(v, u) - b$
8. End If
9. End For
10. Output f' .

Lemma Let f' be the output of $\text{Augment}(f, P)$. Then f' is a flow, and $\text{Value}(f') = \text{Value}(f) + b$.

Proof To show that f' is a flow, we have to verify the capacity and conservation conditions. Here we only have to check the capacity condition for the edges in P , and the conservation condition for the nodes in P .

Exercise Verify the capacity and conservation conditions for f' .

To show that $\text{Value}(f') = \text{Value}(f) + b$, we look at the set of all edges that comes out of s . There is exactly one (forward) edge in P in this set, and the flow on that edge is increased by b . QED.

Max-Flow Min-Cut Theorem Let f be a flow in a flow network G with source s , sink t . The following are equivalent:

1. f is a maximum flow in G .
2. The residual network G_f has no augmenting path.
3. $\text{Value}(f) = c(A, B)$ for some cut (A, B) .

Proof We will show that $(1) \implies (2) \implies (3) \implies (1)$.

$(1) \implies (2)$: From the Lemma above.

$(2) \implies (3)$: The cut (A, B) is defined as follows: A is the set of all nodes reachable from s in G_f . By (2), t is not in A . Let $B = V - A$.

Now for all edges e from A to B we have $f(e) = c(e)$, and for all edges e from B to A we have $f(e) = 0$. Therefore

$$f^{out}(A) = c(A, B), \quad \text{and} \quad f^{in}(A) = 0$$

Consequently $c(A, B) = f^{out}(A) - f^{in}(A)$, so $c(A, B) = \text{Value}(f)$.

$(3) \implies (1)$: This follows from the Corollary 3 we proved last week that

$$\text{Value}(f) \leq c(A, B)$$

for all flow f and cut (A, B) . QED.

Ford-Fulkerson Algorithm

1. For each e in G do $f(e) \leftarrow 0$ End For % start with a 0 flow
2. While there is an st -path in G_f do
3. $P \leftarrow$ an st -path in G_f
4. $f' \leftarrow \text{Augment}(f, P)$
5. $f \leftarrow f'$
6. Update G_f .
7. End While
8. Return f .

Theorem Assume that all capacities are integers, then in each iteration of the While loop:

All flows on the edges are integers

Proof We prove the Theorem by induction (on the iteration).

Base case We start with a 0-flow. At this time all flows on the edges are 0, so the statement is true.

Induction step Consider the iteration $(i + 1)$. By the induction hypothesis for iteration i , the flows on the edges before we start the $(i + 1)$ iteration are integers, so the residual network has integer capacities. Therefore the bottleneck of the augmenting path is an integer. As a result, the increase in flow value is an integer; therefore the new flows are all integers.

Corollary Suppose that all capacities of the network are integers. Let $m = |E|$, and

$$C = \sum_{e \text{ out of } s} c(e)$$

Then

- (a) The Ford-Fulkerson algorithm runs in time $\mathcal{O}(mC)$.
- (b) In the output of the Ford-Fulkerson algorithm, all flows on the edges are integers.

Proof (a) Since the value of the flows increase by at least 1 in each iteration, and the maximum flow value is $\leq C$, there can be at most C iterations. Also, each iteration can be done in time $\mathcal{O}(m)$, so the total running time is $\mathcal{O}(mC)$.

Part (b) follows from the Theorem above. QED.

Edmons-Karp Algorithm

The augmenting path in step 3 is found using a BFS. It is a shortest path from s to t (i.e., an st -path with smallest number of edges). It can be shown that with this choice of the path, there can be at most $\mathcal{O}(mn)$ iterations. Thus the total running time of the algorithm is $\mathcal{O}(m^2n)$ (here $m = |E|, n = |V|$).

Push-Relabelling algorithm [Section 7.4]

This algorithm follows a different approach, and can be made to run in time $\mathcal{O}(n^3)$.

NOTE When C is small (e.g., $\mathcal{O}(n)$) then $\mathcal{O}(mC)$ is better than $\mathcal{O}(n^3)$.

Disjoint Paths in Directed Graph [Section 7.6]

Given a directed graph G with two designated nodes s, t . The problem is to find the maximum number of edge-disjoint st -paths.

1. The Flow Network The underlying graph for the flow network is G . The capacity of each edge is set to 1.

2. The Max-Flow Algorithm We run Ford-Fulkerson algorithm. Here $C \leq n$, since there are at most n edges going out of the source s . The running time for the max-flow algorithm is $\mathcal{O}(mn)$.

3. Max-flow and an optimal solution We show that the value of a max-flow is the same as the maximum number of edge-disjoint st -paths.

Exercise Show that the maximum number of edge-disjoint st -paths is \leq max-flow value by showing that given a set of k edge-disjoint st -paths, there is a flow of value k .

Exercise Show that the value of the max-flow returned by the algorithm in step 2 above is \leq the maximum number of edge-disjoint st -paths.