20th Lecture: 8/3

Flows in networks. When studying digraphs, it is common to replace the word "edge" by *arc*, which means a directed edge, i.e.: an ordered pair $(v_1,\, v_2)$ of vertices. We shall still write $G = (V,\, E)$, where henceforth G is a digraph and E a set of arcs.

Definition 20.1. A *network* is a digraph $G = (V,\, E)$ with the following three properties:

(i) the digraph is weighted, i.e.: every arc e has a non-negative weight which, in this setting, is called its *capacity* and denoted $c(e)$

(ii) the underlying undirected, unweighted graph is connected

(iii) the digraph is *acyclic*, i.e.: it contains no directed cycles.

It is easy to see[1] that (iii) implies that the digraph contains at least one vertex of indegree zero and one of outdegree zero. A vertex of the former kind is called a *source* and one of the latter kind is called a *sink*.

In fact, without loss of generality we can always assume there is exactly one source and one sink. The idea is, given a network G, to add one *supersource* and one *sueprsink*. One draws one arc of extremely high (think infinite) capacity from the supersource to each actual source and one arc of extremely high capacity from each actual sink to the supersink. The network with the added supervertices will, for all intents and purposes, have exactly the same properties as the original network.

Definition 20.2. Let $G = (V,\, E)$ be a network. A function $f : E \to [0,\, \infty)$ is called a *flow* if it satisfies the following two properties:

(i) $f(e) \leq c(e)$ for every $e \in E$

(ii) for each vertex $v \in V$ which is neither a source nor a sink one has

$$\sum_{e=(w,\, v)} f(e) = \sum_{e=(v,\, x)} f(e). \tag{20.1}$$

In words, (20.1) says that the total flow into v equals the total flow out of v. It is called the *Mass Conservation Law*.

An arc e for which $f(e) = c(e)$ is said to be *saturated* by the flow f.

Definition 20.3. Let f be a flow in a network $G = (V,\, E)$ which has a unique source s and a unique sink t. The *strength* or *value* of f, denoted $|f|$ or $\mathrm{val}(f)$, is the total flow out of s, i.e.:

$$|f| = \sum_{e=(s,\, v)} f(e). \tag{20.2}$$

Note that the Mass Conservation Law implies that that the total flow out of s must equal the total flow into t (informally: everything must flow from s to t and nothing can get

[1] Assume there is no sink. Pick a starting vertex and follow any directed path in G. Since there is no sink we can never get stuck. Since the graph is finite we must eventually hit the same vertex twice, thus forming a directed cycle - contradiction ! A similar argument proves the existence of a source (walk backwards !).

lost). Hence, (20.2) can also be expressed as

$$|f| = \sum_{e=(v,\,t)} f(e). \tag{20.3}$$

Definition 20.4. Let $G = (V,\, E)$ be a network with unique source s and unique sink t. A *cut* is a partition $(S,\, T)$ of V, i.e.: $S \cup T = V$ and $S \cap T = \phi$, such that, moreover, $s \in S$ and $t \in T$.

The *capacity* of a cut $(S,\, T)$ is the sum of the capacities of all the edges crossing *from S to T*, i.e.:

$$c(S,\, T) = \sum_{e=(v,\,w):v\in S,\,w\in T} c(e). \tag{20.4}$$

The Mass Conservation Law implies that "everything coming out of s must eventually flow into t". This immediately implies that the strength of any flow cannot exceed the capacity of any cut. This proves the easy half of the following result:

Theorem 20.5. (Max-Flow Min-Cut Theorem) *Let $G = (V,\, E)$ be a network. The maximum strength of a flow in G equals the minimum capacity of a cut in G.*

To prove the theorem it remains to prove the existence of a flow f and a cut $(S,\, T)$ such that $|f| = c(S,\, T)$. The idea for doing so leads to an algorithm for finding a maximum strength flow, called the *Ford-Fulkerson algorithm*, and incorporates a concept very similar to that of augmenting path encountered in our study of matchings (see Remark 20.8 below):

Definition 20.6. Let f be a flow in a network $G = (V,\, E)$ and let $v_1 v_2 \ldots v_k$ be a path in the underlying undirected, unweighted graph. This is said to be an *f-augmenting path* if, for each $1 \leq i \leq k-1$, the following holds:
(i) if $e_i = (v_i,\, v_{i+1})$ is an arc in G, then $f(e_i) < c(e_i)$,
(ii) if $e_i = (v_{i+1},\, v_i)$ is an arc in G, then $f(e) > 0$.
An arc of type (i) is said to be directed *forwards* along the path, while an arc of type (ii) is said to be directed *backwards*.

The point is that, given an f-augmenting path, we can increase the flow along it by the amount

$$\delta = \min\{\delta_+,\, \delta_-\}, \tag{20.5}$$

where

$$\delta_+ = \min\{c(e_i) - f(e_i) : e_i \text{ is directed forwards along the path}\}, \tag{20.6}$$

$$\delta_- = \min\{f(e_i) : e_i \text{ is directed backwards along the path}\}. \tag{20.7}$$

Note that "increasing the flow by δ along the path" means (i) increasing the flow from $f(e_i)$ to $f(e_i) + \delta$ along every forwards arc and (ii) decreasing the flow from $f(e_i)$ to $f(e_i) - \delta$ along every backwards arc.

We can now complete the proof of Theorem 20.5. Let f be a flow in the network G. If there exists an f-augmenting path from s to t then we can increase the flow along

it and so f cannot have had maximum strength. So suppose f has maximum strength. Let S be the set of vertices reachable from s by an f-augmenting path and T the set of unreachable vertices. Trivially, $s \in S$ and, from what we've just said, also $t \in T$. Hence (S, T) is a cut. Let $e = (v, w)$ be any arc included in this cut, i.e.: $v \in S$ and $w \in W$. If we had $f(e) < c(e)$ then we could increase the flow along e and hence reach w with an f-augmenting path, a contradiction. Thus every arc in the cut must be saturated by f. But Mass Conservation implies that the strength of f must equal the total flow across any cut, hence $|f| = c(S, T)$, v.s.v.

Ford-Fulkerson algorithm for finding a maximum strength flow. Begin with the *everywhere zero flow*, i.e.: $f(e) = 0$ on every arc. Perform a breadth-first search for f-augmenting paths out of s, marking those vertices which are reachable from s by such a path. If one finds an f-augmenting path from s all the way to t, then increase the flow along it by the amount denoted δ in (20.5) and repeat the search procedure for this new flow. If no f-augmenting path from s to t is located, then let S be the set of all marked vertices and $T := V \backslash S$. Then f is a flow of maximum strength and $|f| = c(S, T)$.

Example 20.7. Starting from the everywhere zero flow, we apply the Ford-Fulkerson algorithm to find a maximum strength flow and a minimum capacity cut in the network in Figure 20.1. The table indicates which augmenting path is chosen at each step of the algorithm.

Step	Augmenting path	Increase in flow strength
1	$a \to b \to d \to z$	8
2	$a \to h \to i \to z$	6
3	$a \to g \to i \to z$	6
4	$a \to g \to b \to d \to z$	7
5	$a \to h \to g \to d \to z$	4
6	$a \to g \to d \to z$	1

The flow f at this point is indicated in Figure 20.2. Its strength is the total flow out of a, namely

$$|f| = f(a, b) + f(a, g) + f(a, h) = 8 + 14 + 10 = 32.$$

The vertices reachable from a by an f-augmenting path are a, b, g, d, h. Note that, in the case of h, the only such path is $a \to g \rightsquigarrow h$, where the squiggle indicates a backwards directed arc. So let $S = \{a, b, g, d, h\}$ and $T = V \backslash S = \{i, t\}$. We have

$$c(S, T) = c(d, z) + c(g, i) + c(h, i) = 20 + 6 + 6 = 32.$$

Thus $|f| = c(S, T)$, confirming that we have located both a maximum strtength flow and a minimum capacity cut.

An important remark here is that there can be many different options for the sequence of f-augmenting paths chosen by the F-F algorithm. Below is an alternative sequence for Figure 20.1, which results in a different maximum flow than before. The new maximum flow is illustrated in Figure 20.3.

Step	Augmenting path	Increase in flow strength
1	$a \to b \to d \to z$	8
2	$a \to g \to i \to d \to z$	5
3	$a \to h \to i \to z$	6
4	$a \to g \to d \to z$	6
5	$a \to h \to g \to i \to z$	1
6	$a \to g \to b \to d \to z$	1
7	$a \to g \to b \to d \rightsquigarrow i \to z$	3
8	$a \to h \to g \to b \to d \rightsquigarrow i \to z$	2

Remark 20.8. The augmenting path algorithm for matchings in bipartite graphs is, in fact, a special case of the Ford-Fulkerson algorithm. To see this, let $G = (X, Y, E)$ be a bipartite graph. We can associate to this a network G' as follows:

(i) direct every edge in G from X to Y

(ii) add a vertex s to the left of X and an arc from s to each vertex of X

(iii) add a vertex t to the right of Y and an arc from each vertex of Y to t

(iv) assign a capacity of 1 to every arc.

Then it is fairly easy to see (left as an exercise to the reader !) that, if we start from the everywhere zero flow in the network G', every augmenting path located by the Ford-Fulkerson algorithm will consist of an arc (s, x), followed by what in Definition 19.4 was termed an augmenting path in G, to a vertex $y \in Y$, followed by the arc (y, t). The F-F algorithm will increase the flow from 0 to 1 along forward arcs and decrease it from 1 to 0 along backward arcs, which is equivalent to replacing the current matching in G by the augmented matching. Hence, the final maximum strength flow will assign flow 0 or 1 to every arc and the arcs from X to Y which are assigned flow 1 correspond to a maximum matching in G.