Network Flow

Advanced Algorithms (CSE 794)
Instructor: Tamal K. Dey
Scribe: Vineet Harbhajanka*

March 31, 2009

1 Introduction

A flow network is a directed graph G = (V,E) having a capacity, c (u,v) $\geq$ 0, (u,v) $\in$ E. It has a pair of special vertices a *source* s (from where all flow originates) and a *sink* t (at which all flow ends) with flow given by $f : V \times V \rightarrow \Re$

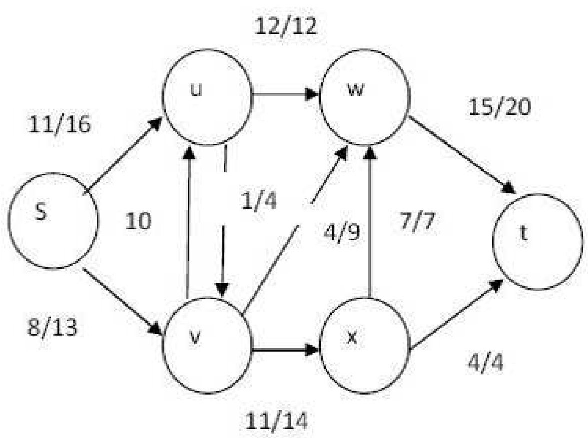

Figure 1: A sample flow network.

1.1 Properties of a Flow

(i) Flow is anti symmetric f(u,v) = - f(v,u).

(ii) For non edges flow can be null or negatives: f(u,v) $\leq$ c(u,v).

(iii) The principle of conservation :Net flow for every vertex other than sink and source is zero.

$$\sum f(u,v) = 0; u \in V - s - t, v \in V$$

*Department of Computer and Information Sciences, The Ohio State University, Columbus, Ohio, USA.

We see that for the source vertex the net flow should be positive, for the sink vertex the net flow should be negative and for every other vertex the net flow should zero. The optimization problem here is maximization of flow following the three principles stated above.

Amount of flow in the network G is given by $\sum$ f(s,v) = $|f|$, $\forall$ v $\in$ V

1.2 Vertex Sets

Next we define flow between vertex sets.

We have f(X,Y): X, Y are disjoint vertex sets and f(X,Y) represents the flow from each vertex of X to each vertex of Y. The flow between vertex sets is given by.

$$f(X,Y) = \sum_x \sum_y f(x,y); x \in X, y \in Y$$

Lemma 1. *Let there be a flow network with directed graph G(V,E), source s, sink t, flow f and the vertices being partitioned into two vertex sets X and Y then:*

(i) Flow from a vertex set X to itself is zero

$$\forall X \subseteq V \; f(X,X)=0$$

(ii) Flow from a vertex set X to another vertex set Y is negative of the flow in the opposite direction

$$\forall X,Y \subseteq V \; f(X,Y)= - f(Y,X)$$

(iii) Flow between vertex sets has distributive property

$$\forall X, Y, Z \subseteq V \text{ with } X \cap Y \text{ is null then}$$
$$f(X \cup Y,Z)=f(X,Z)+f(Y,Z)$$
$$\text{and also, } f(Z, X \cup Y)=f(Z,X)+f(Z,Y)$$

1.3 Other Results

We see that f(s,V) =|f| = f(s,V-s) [as f(s,s)=0]

Let, V= s $\cup$ (V-s)

f(V,V)=f(s,V)+f(V-s,V)

So, |f|=f(s,V)=f(V,V)-f(V-s,V)
=0-f(V-s,V)=f(V,V-s)
=f(V,t)+f(V,V-s-t)
=f(V,t)[as f(V,V-s-t) is zero]

2 Ford Fulkerson Algorithm

It is an iterative algorithm which begins with a flow of zero and then continually tries to push more flow from *s* to *t* through the network till it can augment the flow amount.

PSEUDO CODE

1. Initialize flow f to zero
2. while there exists an augmenting path p
3. do augment flow f along p
4. endwhile

2.1 Residual Network

Residual capacity is defined as c_f(u,v)=c(u,v)-f(u,v). Basically it represents the remaining capacity of flow left unused.

Example :

Say c(u,v)=16 and we have f(u,v)=11
then c_f(u,v)=16-11=5
also f(v,u)=-11 so c_f(v,u)=0-(-11)=11

We represent the residual network as :-

$$G_f = (V, E_f); \forall(u, v) < 0, (u, v) \in E_f$$

2.2 An iteration through Ford Fulkerson Algorithm

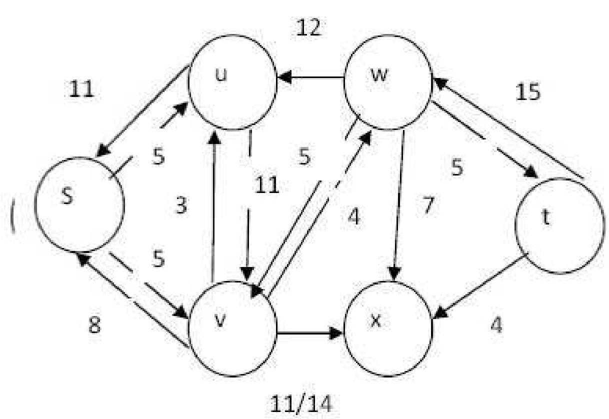

Figure 2: The flow before the iteration.

Now choose an augmenting path from s to t from G_f

Augmenting path: A path from s to t in G_f here it is (s,v,w,t)

Residual capacity of p: minimum capacity along this path given by

$$minC_f(u,v); (u,v) \in p$$

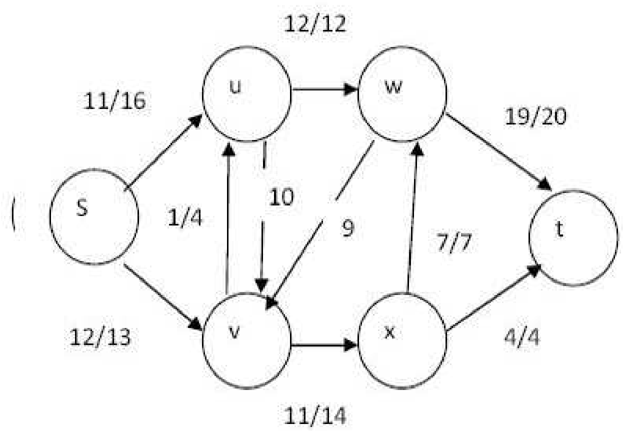

Figure 3: The flow after the iteration.

Lemma 2. *Let G=(V,E) be a flow network with a flow f. Let p be an augmenting path in G_f*

(i) The augmented flow along p is given by

f_p(u,v)=c_f(p);where (u,v) $\in$ p

=-c_f(p);where (v,u) $\in$ p

=0; otherwise

2.3 Cuts in graph

A cut in a graph is a partitioning of vertex set. We define a cut (S,T) in G(V,E) if it partitions V into S, T so that s$\in$ S, t$\in$ T

Netflow for a cut: f(S,T)

Capacity of a cut:

$$C(S,T) = \sum_u \sum_v C(u,v); u \in S, v \in T$$

We can say augmented flow satisfies $|f'| = |f| + |f_p| < |f|$ and it is a flow in G

Theorem 1. ***Max Flow Min Cut Theorem**: Max flow in a network equals the capacity of the minimum cut.*

Proof. The proof is through two lemmas as follows.

Lemma 3. *Let S be a vertex set including source s and T be a vertex set including sink t. Then flow from S to T is equal to the flow in the network.*

f(S,T)=|f|=f(S,V)-f(S,S)=f(S,V)

Or, f(S,T)=f(s,V)+f(S-s,V)=|f|

[Since,2nd term is zero as net flow from vertices other than s,t is 0]

Lemma 4. *Let S be a vertex set including source s and T be a vertex set including sink t. Then capacity of flow from S to T is equal to or more than the flow in the network*

$$|f| \leq c(S,T)$$
$$|f| = f(S,T) = \sum_u \sum_v f(u,v); u \in S, v \in T$$
$$\leq \sum_u \sum_v C(u,v) < C(S,T); u \in S, v \in T$$

□

Theorem 2. *If f is a flow in a network G=(V,E) with source s, sink t then following statements are equivalent*

(1) f is a maximum flow.

(2) Residual network Gf does not have any augmenting path.

(3) Amount of flow is C(S,T) for some cut (S,T)

Proof. : The proof of equivalence of three statements is done by a circular proof of first statement being equivalent to second, second to third and third to first.

Stmt 1 and 2: Taking a contradiction, if 2 is not true this means there is an augmenting path hence 1 is also not true. Hence 1 implies 2

Stmt 2 and 3: If the residual network Gf contains no augmenting paths, then — f — = c(S,T) for some cut (S,T) of G. Let the set S contain all the vertex v which has a path connected from s on G_f. Since there is no augmenting path in G_f, . The partition (S,T) is a cut of G. And, capacity of the edge (u,v) on G_f is 0, or there exists a path from s to v. According to the definition of the residual network, we have f(v,u) = 0 and f(u,v) = c(u,v) in G. Therefore,

$$f(S,T) = \sum_u \sum_v (f(u,v) - f(v,u)); u \in S, v \in T$$

$$= \sum_u \sum_v (f(u,v))$$

$$= \sum_u \sum_v (c(u,v)) = c(S,T)$$

And,

$$|f| = f(s,V) = f(s,V) + f(S-s,V) - f(S,S) = f(S,T)$$

Thus,

$$|f| = f(S,T) = c(S,T)$$

Stmt 3 and 1: If $|f|$ = c(S,T) for some cut (S,T) of G, then f is a maximum flow of G. By the following observation, we get that for any flow f, amount of flow is upper bounded by any c(S,T). thus 3 implies 1 □