SNS COLLEGE OF ENGINEERING

Coimbatore-107

COURSE NAME: ANALYSIS OF ALGORITHM

II YEAR/ IV SEMESTER

UNIT – IV

ITERATIVE IMPROVEMENT

Topic

Flow Networks

Ford Fulkerson Algorithm

o(n) (for ...

Ford Fulkerson Algorithm

Topic 1: Unit - IV. Flow Network

Flow Network is a directed graph where it includes ;

(1) Capacity : Each Edge has a Capacity (maximum allow flow).

(2) Source Node (S) : place where flow Starts.

(3) Sink Node (t) : → place where flow ends.

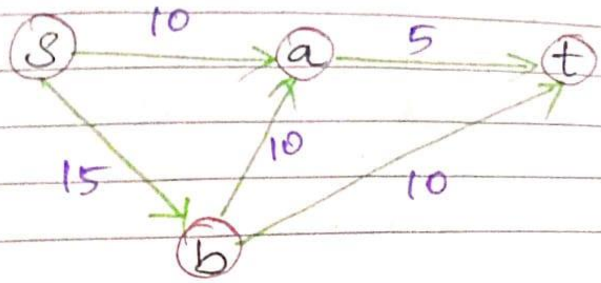
Rules to be maintained during flow:-

(i) Capacity constraint :

Flow on edge must not exceed its Capacity

(ii) Flow Conservation

For all nodes except 'S' and 't', amount of flow Coming in equals the amount going out.



Edge capacities: Source = S; Sink = t.

From	To	Capacity
S	a	10
S	b	15
a	t	5
b	a	10
b	t	10

Step 1:
Initial Flow = 0

Step 2: Augmenting path #1

$S \xrightarrow{10} a \xrightarrow{5} t$

$\Rightarrow$ min (10, 5) = 5

$\Rightarrow$ Send 5 units

Update Residuals:

(i) $S \rightarrow a$ = 10 − min = 10 − 5 = 5

(ii) $a \rightarrow t$ = 5 − min = 5 − 5 = 0

(iii) Add Reverse Edges:

$a \rightarrow S$: 5

$t \rightarrow a$: 5 (a−t : 0 ; we send 5 units of flow from a→t : so remaining capacity becomes zero) So t→a: which create Reverse edge & have equal to flow that was sent already.

∴ $t \to a = 5$

Flow = 5

Step 3: Augmenting path #2 :

$s \xrightarrow{15} b \xrightarrow{10} t$

⇒ min(15,10) = 10

⇒ Send 10 units.

Update Residuals:

$s \to b = 15 - 10 = 5$

$b \to t = 10 - 10 = 0$

Add Reverse edges: (undo) to reverse ed

$b \to s = 10$

$t \to b = 10$

∴ Flow = 5 + 10 = 15

Step 4: Augmenting path #3

$s \to b \to a \to t$

⇒ Update from previous values for $s \to b : 5$ & $a \to t = 0$

⇒ For $b \to a$; No previous value. So Refer from initial table

$b \to a : 10$

We can send any more flow to 't'
So this path is blocked.

Finally it uses flow = 15

Residual Graph Summary

Edge	Residual.
$S \to a$	5
$S \to b$	5
$a \to t$	0
$b \to t$	0
$b \to a$	10
$a \to S$	5
$b \to S$	10
$t \to a$	5
$t \to b$	10

Note :

(i) Forward Edge : Tracks how much capacity is left

(ii) Reverse Edge : Tracks how much flow we can cancel (or) reroute [undo].

(iii) Residual Graph :

This finds more complex paths (like loops) to push more flow if possible.

Flow Network	Ford Fulkerson
=> includes Input graphs with Capacities	Algorithm runs on this Input
=> Has source 's' and Sink 't'	Uses S & t to find flow path
=> Defines edge Capacity limits	Respects those Capacities during Updates
=> Residual Graph is derives from it	updates flow & residuals in each iteration
=> Goal: Max flow from S → t	provides that max. flow.

Algorithm:

```
while (d-flow (rGraph, S, t, visited, parent
{
// Find min Capacity along the path
path flow = INT_MAX
for( v=t; v!=S; v= parent [v])
{
u = parent [v];
if ( rgraph [u][v] < path_flow)
path_flow = rGraph [u] [v]; }
```

```
//update residual capacities
for (v=t ; v!=s ; v=parent [v])
{ u = parent [v];
  rGraph [u][v] -= path flow;
  rGraph [v][u] += path flow;
}
// Add to total flow
max_flow += path flow;
}
```

Time Complexity:

O(max flow * E)

⇒ max flow ⇒ Total maximum flow value

⇒ E ⇒ Number of Edges.

Space Complexity:

O(V+E)

Stores the Graph & Residual Capacities.