

## Ford-Fulkerson Method

Flow maximization in a network (graph)  
with capacities

Basic idea:

- Find a path from source to target that still has flow capacity (*augmenting path*)
- Add the maximum flow allowed along this path
- Repeat until *no augmenting path*

## Issues

1. How do we account for flow by *cancellation*
2. Does adding an augmenting path lead to a legal flow?
3. Will this process converge?
4. If so, will it lead to *a maximum flow*

### Problem Formulation

Directed graph  $G = (N, E)$

Two special nodes:  $s$  *source*       $t$  *target*

Assume for any node  $v \in N$ , there are paths

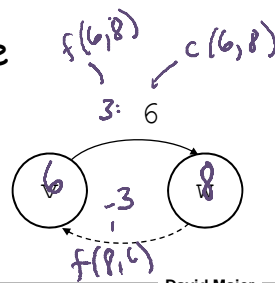
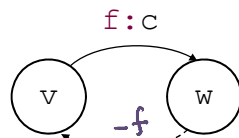
$s \rightsquigarrow v \rightsquigarrow t$

Capacity  $c(v, w) \geq 0$

If  $(v, w)$  not an edge, then

$c(v, w) = 0$

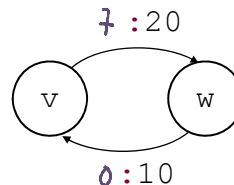
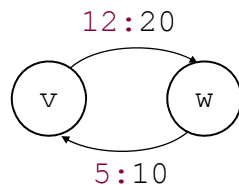
Flow  $f(v, w)$  can be



### Detail

Assume no "useless" flows between nodes

Positive flow in only one direction



## Legal Flow $f$

$$1. f(v, w) \leq c(v, w)$$

$$2. f(v, w) = -f(w, v)$$

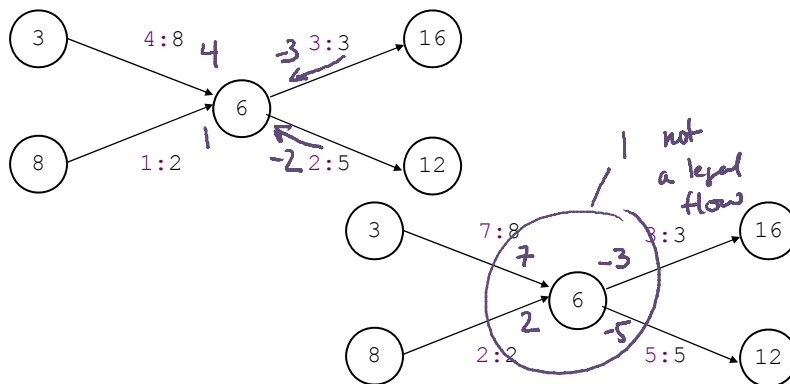
3. For any  $v \neq s, t$

$$\sum_{w \neq v} f(v, w) = 0$$

## Inputs = Outputs

Consider node 6

$$f(6, 16) + f(6, 12) + f(6, 8) + f(6, 3)$$

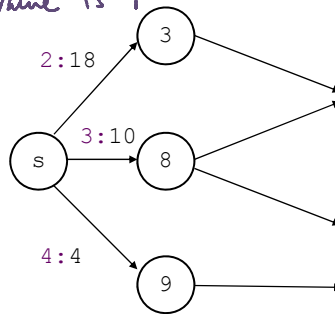


Value of Flow  $f$

Total flow out of source

$$|f| = \sum f(s, w)$$

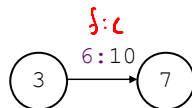
value is 9



Residual Capacity for a Flow

Residual Capacity between nodes  $v$  and  $w$ :

$$r(u, v) = c(u, v) - f(u, v)$$



$$r(3, 7) = c(3, 7) - f(3, 7)$$

4                      10                      6

$$r(7, 3) = c(7, 3) - f(7, 3)$$

6                      0                      -6

Algorithm Design & Analysis

### Residual Graph for Flow $f$

$R = (N, E')$   
 $E' = \{ (v, w) \mid r(v, w) > 0 \}$

Capacities are residual capacities for  $f$

Lecture Notes 5 David Maier **9**

Algorithm Design & Analysis

### Example

The top diagram shows a network with nodes  $s, s1, s2, s3, s4, s5, s6, s7, s8, t$ . Edges and their flow:capacity values are:

- $s \rightarrow s1$ : 2:25
- $s \rightarrow s2$ : 0:20
- $s \rightarrow s3$ : 10:25
- $s1 \rightarrow s2$ : 0:9
- $s1 \rightarrow s4$ : 2:12
- $s2 \rightarrow s4$ : 0:15
- $s2 \rightarrow s5$ : 7:7
- $s2 \rightarrow s3$ : 7:10
- $s3 \rightarrow s8$ : 3:3
- $s4 \rightarrow s5$ : 0:10
- $s4 \rightarrow s6$ : 2:4
- $s4 \rightarrow s5$ : 0:7
- $s5 \rightarrow s6$ : 6:6
- $s5 \rightarrow s8$ : 1:2
- $s6 \rightarrow s7$ : 0:6
- $s6 \rightarrow t$ : 8:20
- $s7 \rightarrow t$ : 4:15
- $s7 \rightarrow s8$ : 4:10

The bottom diagram shows the residual graph with blue residual capacities and red augmenting paths. A red arrow from  $s$  to  $t$  is labeled "add a flow of 2".

Lecture Notes 5 David Maier **10**

Algorithm Design & Analysis

### Capacity of a Path

Minimum capacity edge  $(v,w)$   $r(v,w)$   
 Add a flow of  $r(v,w)$  along the path

Lecture Notes 5 David Maier **11**

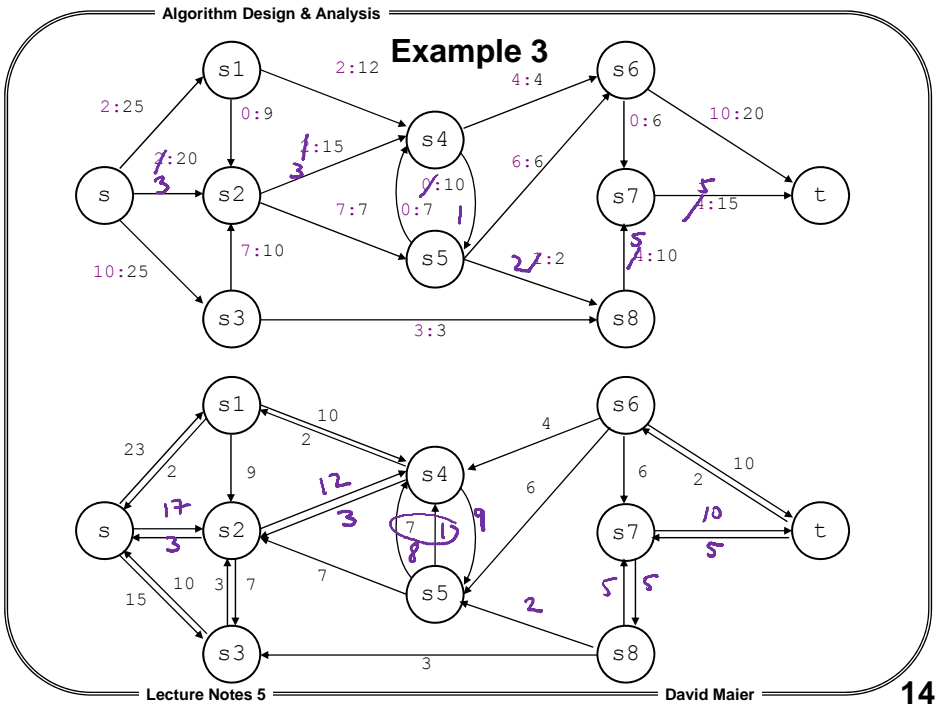
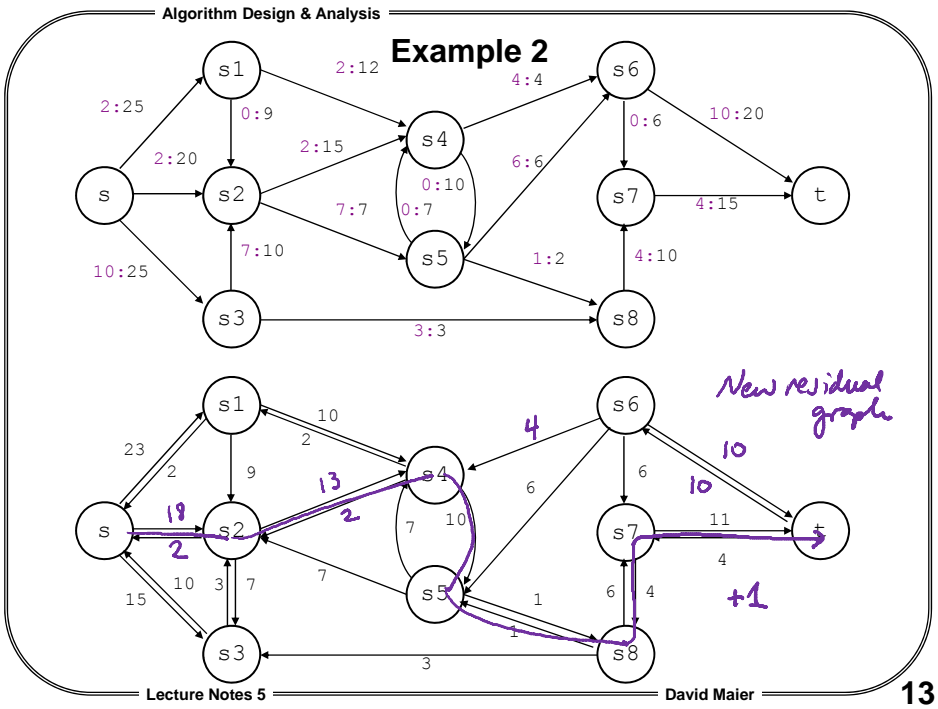
Algorithm Design & Analysis

### Is the Result a Legal Flow?

- Capacity?   
*residual graph shows available capacity*
- Skew symmetry?   
 $f(v,w) = -f(w,v)$
- Conservation?

Lecture Notes 5 David Maier **12**

# CS 410/584, Algorithm Design & Analysis, Lecture Notes 5



Algorithm Design & Analysis

### Is it a Maximum Flow?

Would seem so:  
Have a group of edges that divides  $s$  from  $t$   
and *can't take more flow*

Lecture Notes 5 David Maier **15**

Algorithm Design & Analysis

### Cut of a Graph

Divide nodes of  $N$  into two groups  $S, T$

$$\text{Net flow across cut } \sum_{v \in S, w \in T} f(v, w) = 2 + 3 + 0 - 1 + 6 + 5 = 15$$

$$\text{Capacity across cut } \sum_{v \in S, w \in T} C(v, w) = 12 + 15 + 7 + 6 + 10 = 50$$

Lecture Notes 5 David Maier **16**

Results

- Net flow across any cut  $|f|$
- $|f|$  is bounded above by *capacity of any cut*
- Max-flow/min-cut theorem
  1.  $f$  is maximum flow
  2. residual graph has no *augmenting paths*
  3.  $|f|$  is capacity of *some S|T*

2  $\Rightarrow$  3

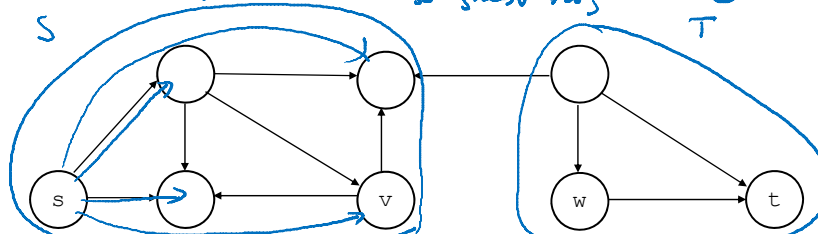
Consider residual graph  $R$  with no augmenting path

$S = \{v \mid s \rightsquigarrow v \text{ in } R\}$   $T = N - S$

Must have  $t \in T$

Claim that for  $(v, w)$  with  $v \in S, w \in T$ , must have  $r(v, w) = 0$  ( $f(v, w) = c(v, w)$ )

Suppose not. Then  $r(v, w) > 0$ . Then  $R$  has an edge  $(v, w)$  and so  $s \rightsquigarrow v \rightarrow w$ , so  $w \in S$



### Basic Implementation

Start with 0 flow

Repeat

↳ Add flow along an augmenting path

*until no such paths*

Does it always converge?

Yes, if capacities are integers. Flow grows by at least *1* each time

How long does it take?

If you pick augmenting paths arbitrarily

$O(|E| \cdot f^*)$

*f\* maximum flow in the graph*

### Edmonds-Karp Algorithm

Start with 0 flow

Repeat

↳ Add flow along an augmenting path with *shortest*

*length*  
*until no paths*

Time complexity no longer depends on value of maximum flow

$O(|N| \cdot |E|^2)$  time

Intuition: Length of shortest path to a node

$v$  in residual graph

• Each addition of flow increases distance to one node

• Distance to a node  $v$  can be increased at most  *$|E|$  times* since  *$|E|$  is length of longest possible path*

