

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

Issues

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

Problem Formulation

Directed graph $G = (N, E)$

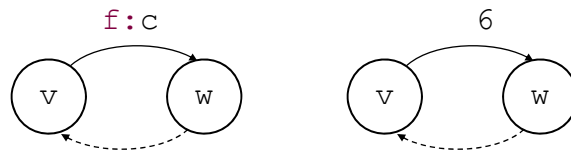
Two special nodes: s t

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

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

If (v, w) not an edge, then

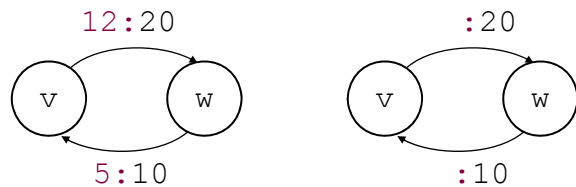
Flow $f(v, w)$ can be



Detail

Assume no "useless" flows between nodes

Positive flow in only one direction



Legal Flow f

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

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

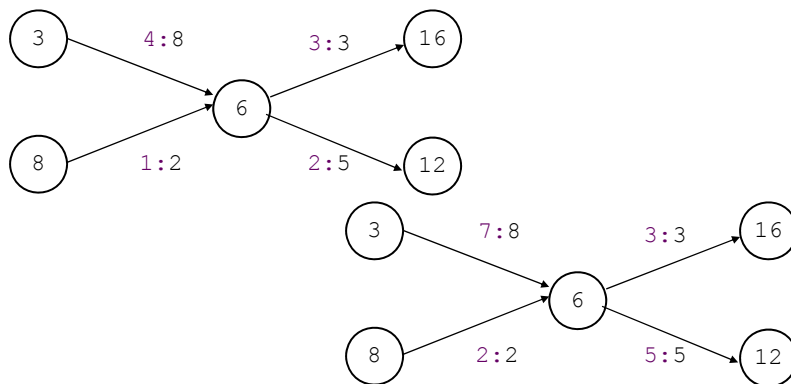
3. For any $v \neq s, t$

$$\sum f(v, w) = 0$$

Inputs = Outputs

Consider node 6

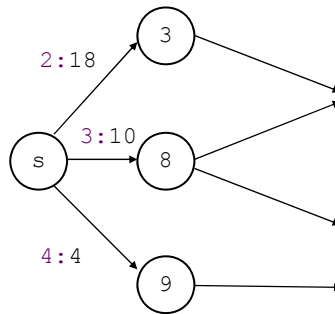
$$f(6, 16) + f(6, 12) + f(6, 8) + f(6, 3)$$



Value of Flow f

Total flow out of source

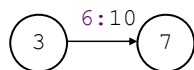
$$|f| = \sum f(s, w)$$



Residual Capacity for a Flow

Residual Capacity between nodes v and w :

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



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

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

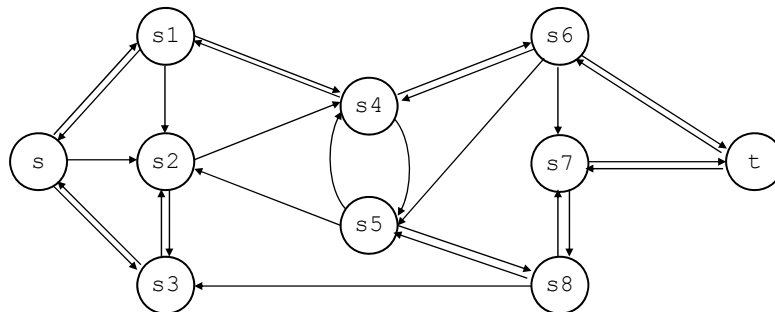
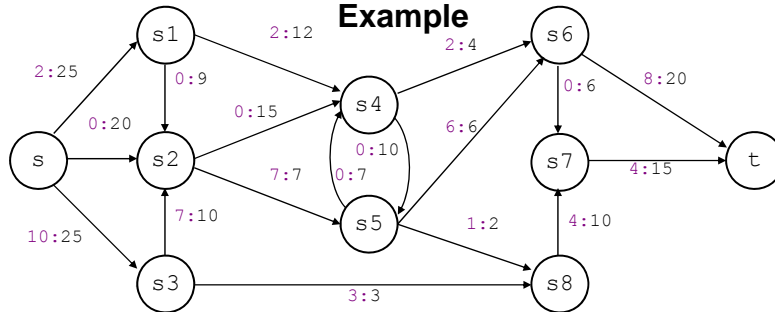
Residual Graph for Flow f

$$R = (N, E')$$

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

Capacities are residual capacities for f

Example



Algorithm Design & Analysis

Capacity of a Path

Minimum capacity edge
Add a flow of along the path

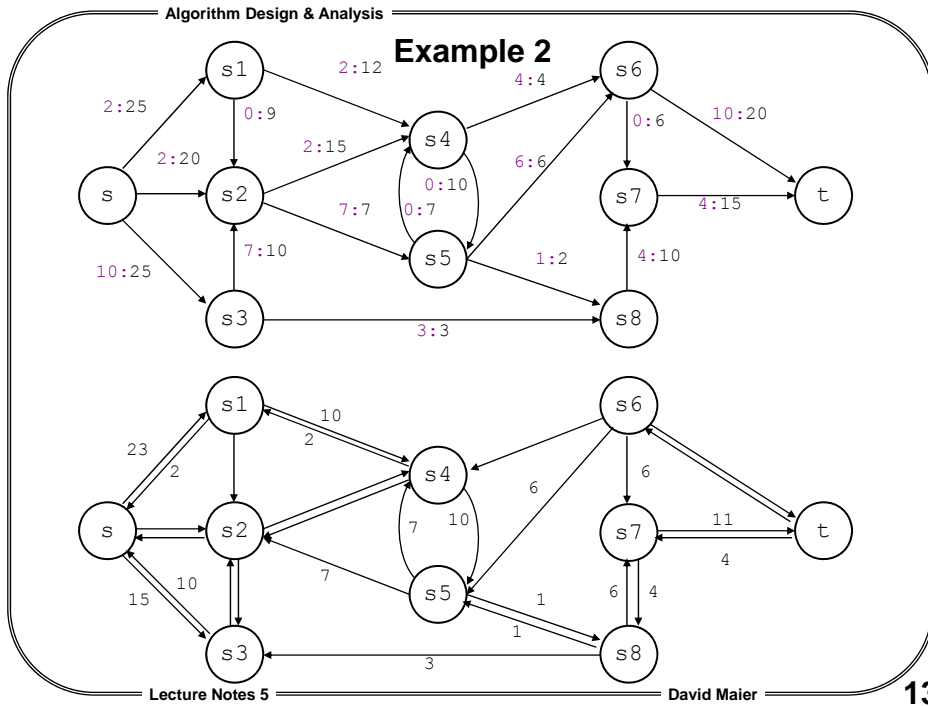
Lecture Notes 5 David Maier **11**

Algorithm Design & Analysis

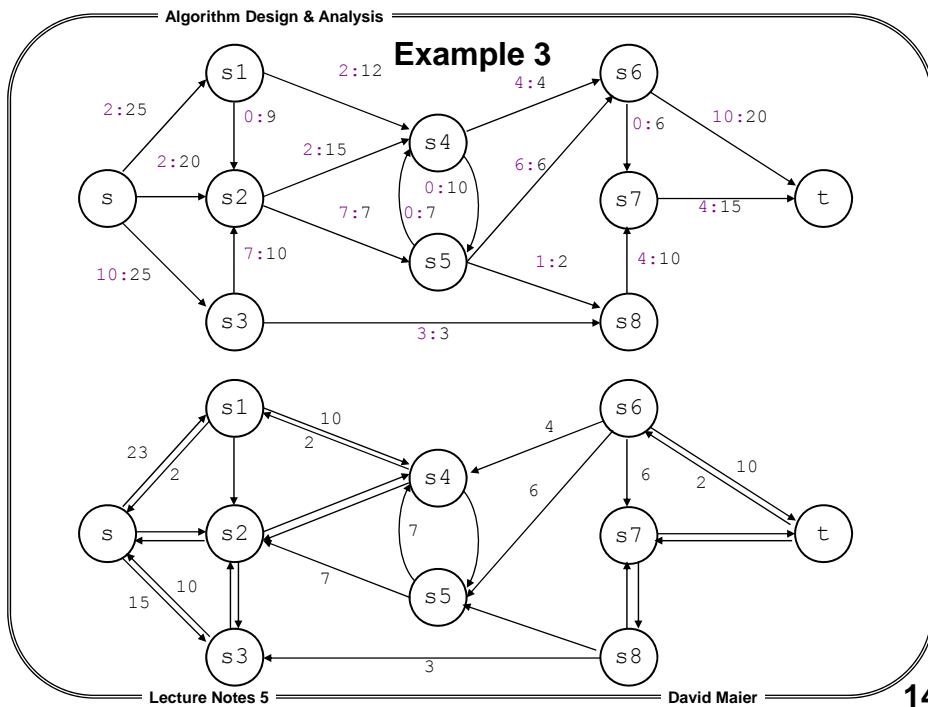
Is the Result a Legal Flow?

- Capacity?
- Skew symmetry?
- Conservation?

Lecture Notes 5 David Maier **12**



13

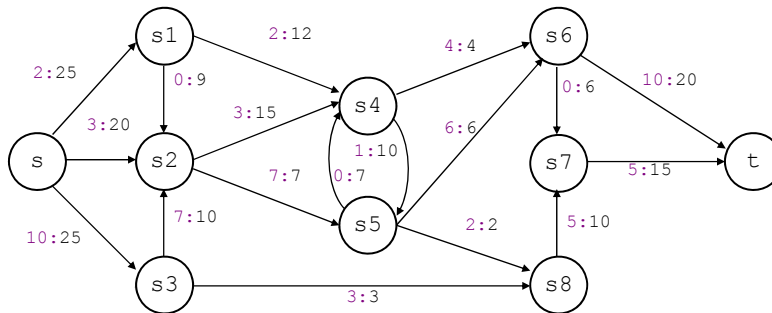


14

Is it a Maximum Flow?

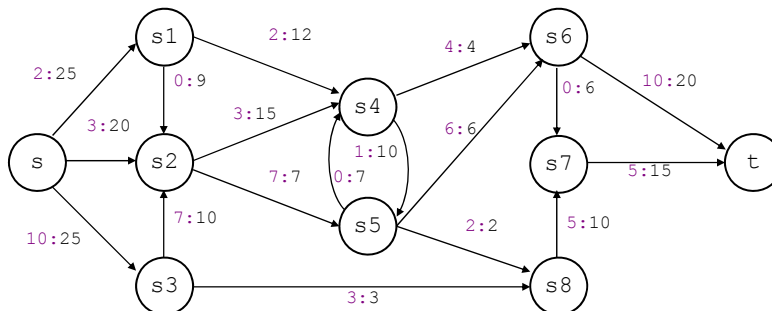
Would seem so:

Have a group of edges that divides s from t and



Cut of a Graph

Divide nodes of N into two groups S, T



Net flow across cut $\sum f(v, w)$

Capacity across cut $\sum C(v, w)$

Results

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

$2 \Rightarrow 3$

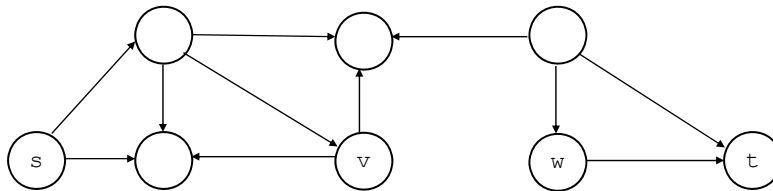
Consider residual graph R with no augmenting path

$$S = \{v \mid \dots\} \quad T = N - S$$

Must have

Claim that for (v, w) with $v \in S, w \in T$, must have

Suppose not. Then $r(v, w) > 0$. Then R has



Basic Implementation

Start with 0 flow

Repeat

Add flow along an augmenting path

Does it always converge?

Yes, if capacities are integers. Flow grows by at least

How long does it take?

If you pick augmenting paths arbitrarily

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

Edmonds-Karp Algorithm

Start with 0 flow

Repeat

Add flow along an augmenting path with

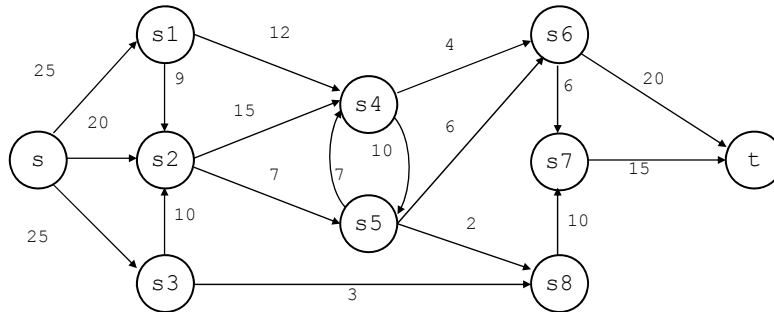
Time complexity no longer depends on value of maximum flow

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

Edmonds-Karp Example



Matrix Multiplication

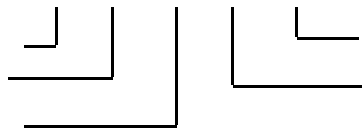
For a divide-and-conquer approach, we want an $n \times n$ matrix to look like a matrix of smaller matrices

$$\begin{pmatrix} 1 & 3 & 21 & 9 \\ 4 & 12 & 36 & 35 \\ 42 & 6 & 10 & 53 \\ 2 & 32 & 12 & 60 \end{pmatrix} \begin{pmatrix} 41 & 2 & 6 & 14 \\ 3 & 11 & 7 & 18 \\ 40 & 13 & 16 & 29 \\ 7 & 19 & 63 & 72 \end{pmatrix}$$

How is a matrix of matrices like a matrix of integers?

Ring

$(S, +, \cdot, 0, 1)$



Axioms

1. $+$, \cdot associate
2. $+$ commutes
3. \cdot distributes over $+$
4. 0 is identity for $+$
5. 1 is identity for \cdot
6. every a in S has an additive inverse

Example Ring

Integers modulo n , \mathbb{Z}_n

$-[a]$ is

0 is

1 is

\mathbb{Z}_4

$-[1]$ is

$[1] + [3]$

Key Example

$$(M_n, +_n, \cdot_n, 0_n, 1_n)$$

is a ring.

Inverse of $A = (a_{ij})$ is

Get weird
 $R_{2, n/2}$ is

Going Between M_n and $R_{2, n/2}$

A, B, C in M_n where

A', B', C' in $R_{2, n/2}$

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} =$$

$$\begin{pmatrix} & \\ & \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$$

reduce $n \times n$ matrix mult to $n/2 \times n/2$ matrix ops

Complexity

Let $M(n)$ be the number of scalar operations () to multiply two $n \times n$ matrices over

Divide-and-conquer doesn't buy much

$$M(n) =$$

In general, if

$$M(n) =$$

then $M(n) \leq kn^{\log m}$ for some k ,
provided $m >$

Complexity 2

If $m = 8$, as before

$$M(n) = O(n^{\lg 8}) =$$

If $m = 7$

$$M(n) = O(n^{\lg 7})$$

Need 2×2 multiplication with 7 scalar multiplications

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

$$\text{e.g., } c_{22} =$$

Strassen's Algorithm

$$\begin{aligned}
 m_1 &= (a_{12} - a_{22})(b_{21} + b_{22}) \\
 m_2 &= (a_{11} + a_{22})(b_{11} + b_{22}) \\
 m_3 &= (a_{11} - a_{21})(b_{11} + b_{12}) \\
 m_4 &= (a_{11} + a_{12})b_{22} \\
 m_5 &= a_{11}(b_{12} - b_{22}) \\
 m_6 &= a_{22}(b_{21} - b_{11}) \\
 m_7 &= (a_{21} + a_{22})b_{11}
 \end{aligned}$$

Don't want to use commutativity of multiplication

More Algebra

$$\begin{aligned}
 c_{11} &= m_1 + m_2 - m_4 + m_6 \\
 c_{12} &= m_4 + m_5 \\
 c_{21} &= m_6 + m_7 \\
 c_{22} &= m_2 - m_3 + m_5 - m_7
 \end{aligned}$$

$$\begin{aligned}
 &(a_{11} + a_{22})(b_{11} + b_{22}) \\
 &- (a_{11} - a_{21})(b_{11} + b_{12}) \\
 &+ a_{11}(b_{12} - b_{22}) \\
 &- (a_{21} + a_{22})b_{11}
 \end{aligned}$$

Another Use of Matrix of Matrices

Connect matrix inversion with matrix multiplication

Inverse of matrix A , denoted A^{-1} , is the unique matrix such that $AA^{-1} = I$

I is the identity matrix

Inverse doesn't always exist

Example

$$\begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} -3 & 2 \\ 2 & -1 \end{pmatrix} = \begin{pmatrix} -3+4 & 2-2 \\ -6+6 & 4-3 \end{pmatrix}$$

Inversion vs. Multiplication

Assuming matrix inversion behaves reasonably, it has the same order as multiplication

Let $I(n) =$

1. $I(n)$ in $\Omega(n^2)$
2. $I(3n)$ in $O(I(n))$

Time to multiply two $n \times n$ matrices, $M(n)$, is in $O(I(n))$.

Reduce Matrix Multiply to an Inversion Problem

Suppose we have A, B to multiply,
where both are $n \times n$

Construct a $3n \times 3n$ matrix D

$$D = \begin{pmatrix} I_n & A & 0 \\ 0 & I_n & B \\ 0 & 0 & I_n \end{pmatrix}$$

Claim that D^{-1} is

$$D^{-1} = \begin{pmatrix} I_n & -A & AB \\ 0 & I_n & -B \\ 0 & 0 & I_n \end{pmatrix}$$

Check

$DD^{-1} =$

$$\begin{pmatrix} I_n & A & 0 \\ 0 & I_n & B \\ 0 & 0 & I_n \end{pmatrix} \begin{pmatrix} I_n & -A & AB \\ 0 & I_n & -B \\ 0 & 0 & I_n \end{pmatrix}$$