# MATH 566: Lecture 16 (10/10/2024)

Today: * max flow application: matrix rounding
* residual network
* max flow algorithm

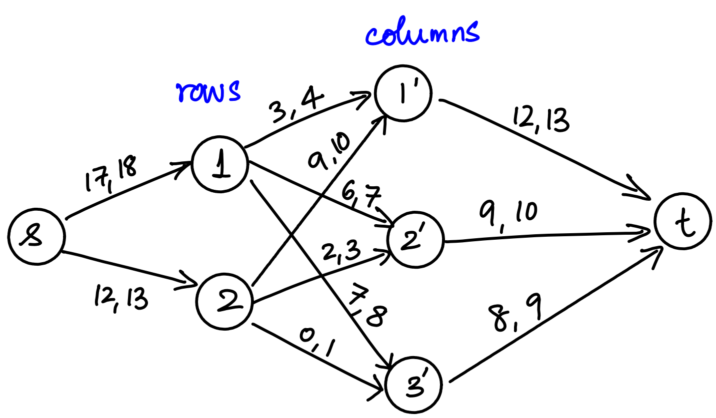
## Matrix Rounding Problem

Given $m \times n$ matrix, $D = [d_{ij}]$ row sums $\alpha_i$ $(1 \le i \le m)$, and column sums $\beta_j$ $(1 \le j \le n)$, where $d_{ij}, \alpha_i, \beta_j \in \mathbb{R}$, the goal is to round each element $d_{ij}$ and row & column sums $(\alpha_i, \beta_j)$, such that the row sums of the rounded elements equal the rounded row sums, and similarly for column sums. For instance, consider the $2 \times 3$ matrix $D$:

| | 1′ | 2′ | 3′ | $\alpha_i$ |
|---|---|---|---|---|
| 1 | 3.1 | 6.8 | 7.3 | 17.2 |
| 2 | 9.6 | 2.4 | 0.8 | 12.8 |
| $\beta_j$ | 12.7 | 9.2 | 8.1 | |

Note that we can round each number up or down — e.g., 3.1 can be rounded to 3 or to 4.
$(\lfloor 3.1 \rfloor = 3, \lceil 3.1 \rceil = 4)$

We formulate this problem as a max flow problem as follows.




$i \xrightarrow{l_{ij},\, u_{ij}} j$

Here,
$N_1 = \{1, \dots, m\}$ : row nodes
$N_2 = \{1', 2', \dots, n'\}$ : column nodes

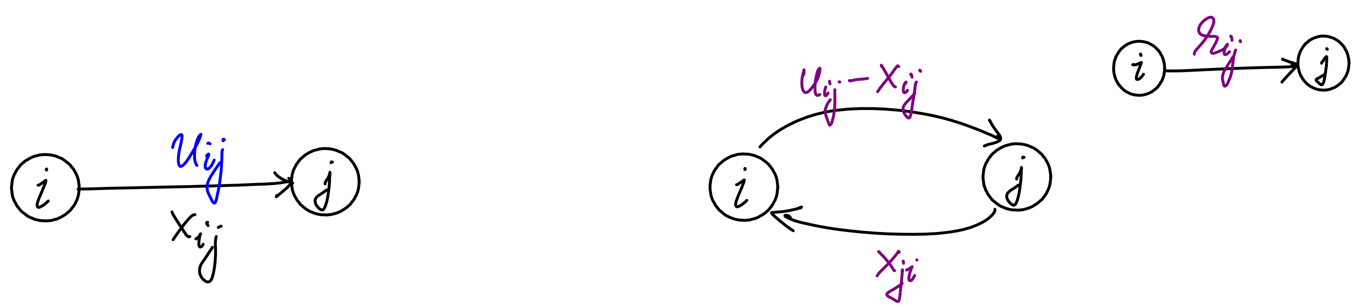
We add $s, t$, arcs $(s,i), i \in N_1$ (row nodes), $(j,t)$ $\forall j \in N_2$ (column nodes), and set $l_{si} = \lfloor \alpha_i \rfloor$, $u_{si} = \lceil \alpha_i \rceil$, $l_{jt} = \lfloor \beta_j \rfloor$, $u_{jt} = \lceil \beta_j \rceil$. We also have $(i,j)$, $1 \le i \le m$, $1 \le j \le n$, with $l_{ij} = \lfloor d_{ij} \rfloor$ (floor), $u_{ij} = \lceil d_{ij} \rceil$ (ceil).

The $x_{ij}$'s in a max flow will be at $l_{ij}$ or $u_{ij}$, which are all integers. Further, flow balance at nodes in $N_1$ and $N_2$ impose the row and column sums, giving a consistent rounding.

---

## Residual Network ("Remaining flow" network)

Algorithms for max flow will employ the concept of residual networks.




The diagram shows the components of the residual network corresponding to arc $(i,j)$. Repeat for all arcs $(i,j) \in A$ to obtain $G(\bar{x})$, the residual network for the flow $\bar{x}$.

Notice that the residual network is defined for a particular flow $\bar{x}$.

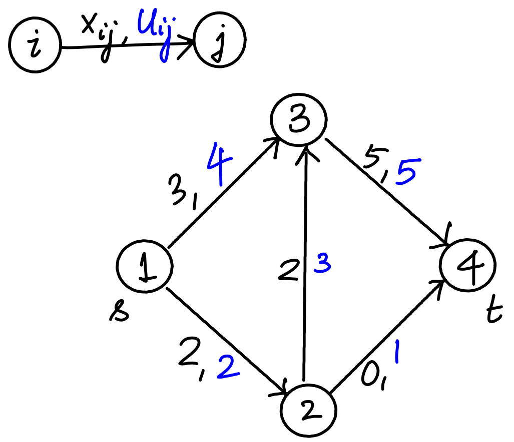
The **residual capacity**, $r_{ij}$, of arc $(i,j)$ is given as

$$r_{ij} = u_{ij} - x_{ij} + x_{ji}.$$

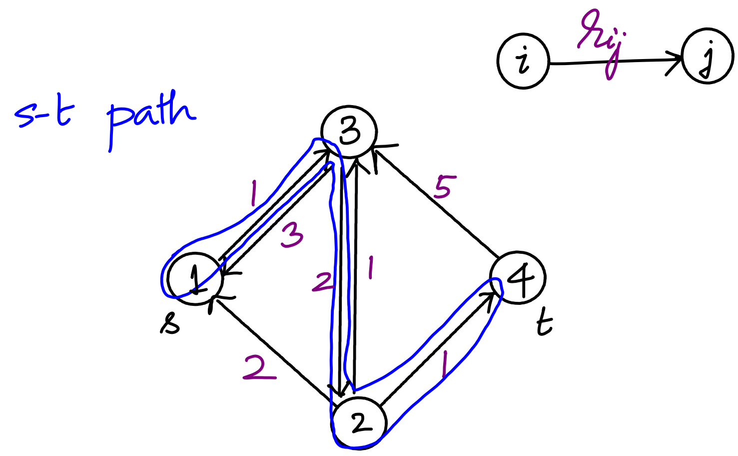
This is the maximum additional flow we can send from $i$ to $j$ using arcs $(i,j)$ and $(j,i)$.

The residual network corresponding to flow $\bar{x}$ is $G(\bar{x})$, and it contains all $(i,j)$ with $r_{ij} > 0$ on $N$, the same node set as the input graph $G$.

Example: Consider the network $G$ with a flow ($x_{ij}$ values) given




$G$ with flow $\bar{x}$




$G(\bar{x})$

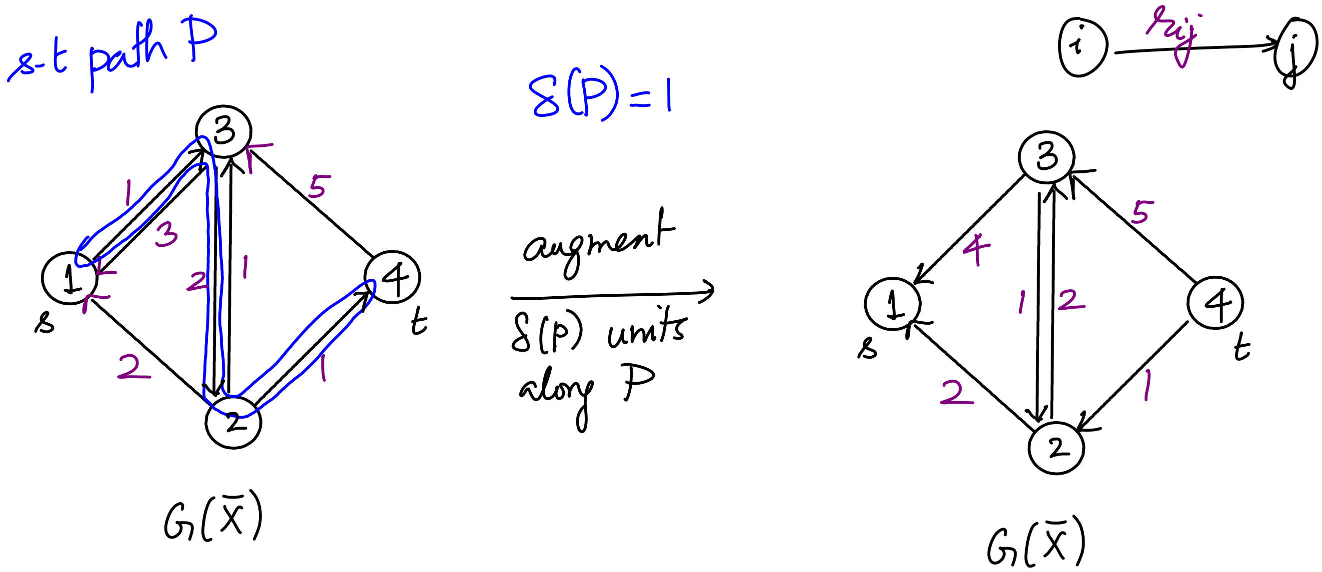
If there is a (directed) path from $s$ to $t$ in $G(\bar{x})$, we could push flow along this path. This idea is central to max flow algorithms — start with some flow $\bar{x}$, find $G(\bar{x})$, and send flow along $s$-$t$ paths in $G(\bar{x})$. How much flow could we send from $s$ to $t$ along such an $s$-$t$ path? We formalize these ideas into definitions now.

**Def** A directed path $P$ from $s$ to $t$ in $G(\bar{x})$ is an **augmenting path**.
The **residual capacity** of the augmenting path $P$ is $\delta(P) = \min\limits_{(i,j)\in P} \{r_{ij}\}$.

Note that $\delta(P) > 0$, as $(i,j) \in P \Rightarrow r_{ij} > 0$ by definition.

To **augment** along $P$ is to send $\delta(P)$ units along each arc in $P$, and modify $\bar{x}$, $G(\bar{x})$, i.e., $r_{ij}$'s, accordingly.

Let's consider the example again.



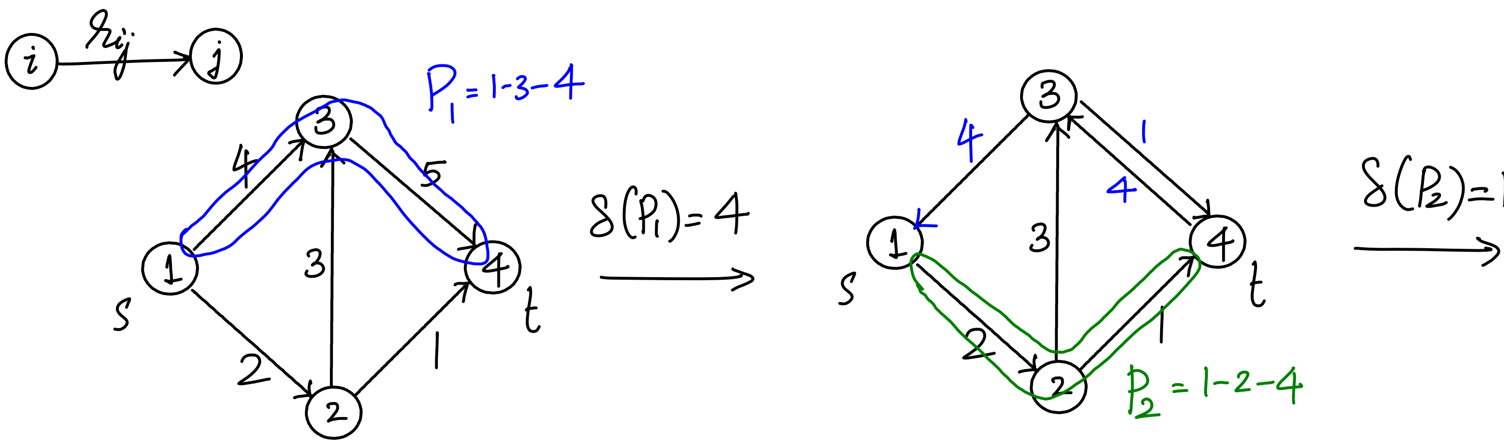

Now there are no more s-t paths in $G(\bar{x})$, which tell us that $\bar{x}$ is optimal!

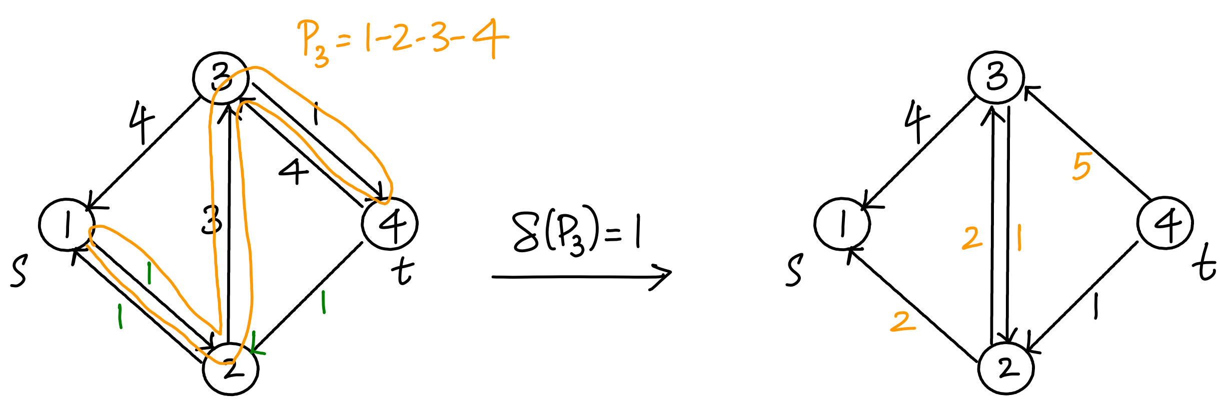
## The Generic Augmenting Path Algorithm (Ford-Fulkerson)

Assume $l_{ij}=0 \;\forall (i,j)\in A$.

```
begin
    x̄ := 0̄;
    initialize G(x̄);
    while G(x̄) has a path P from s to t do
        augment δ(P) units of flow along P;
        update x̄, G(x̄);
    end_while
end_begin
```

**Example** We start with $\bar{x} = \bar{0}$, i.e., the zero flow. Notice that $G(\bar{x}) = G$ when $\bar{x} = \bar{0}$, assuming all $u_{ij} > 0$.




There are no more augmenting paths, and hence the flow is maximum. Notice that the value of the maximum flow is $\delta(P_1) + \delta(P_2) + \delta(P_3) = 4 + 1 + 1 = 6$.