

## Algorithm Design and Analysis

บทที่ 10  
Network flow Part I

## Network Flow Problem

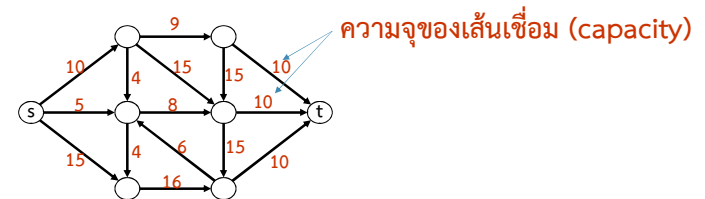
## Network Flow Problem

ในทฤษฎีกราฟ Flow network หรืออาจจะรู้จักในชื่อ Transportation network คือกราฟแบบมีทิศทาง (Directed graph) ที่แต่ละเส้นเชื่อมจะมีความจุ (Capacity) และแต่ละเส้นเชื่อมจะรองรับ flow โดยปริมาณของ flow บนเส้นเชื่อมจะไม่สามารถเกินความจุของเส้นเชื่อม

Directed graph ของปัญหานี้จะถูกเรียกว่า network ซึ่ง network จะจำลองโมเดลของการจราจรบนถนน, การไหลของน้ำในท่อ, กระแสไฟฟ้าในวงจร เป็นต้น

## Flow network

- เป็นการจำลองการไหลของวัตถุผ่านเส้นเชื่อม
- เป็นกราฟแบบมีทิศทาง Digraph  $G = (V, E)$  ที่
  - ▶ มีโหนด source  $s \in V$  และ sink  $t \in V$  (ไม่มีเส้นเชื่อมคู่ขนาน ไม่มีเส้นเชื่อมเข้า s และไม่มีเส้นเชื่อมออกจาก t)
  - ▶ มีความจุ capacity  $c(e)$  ของแต่ละเส้นเชื่อม  $e \in E$  และเป็นค่าความจุที่ไม่ติดลบ

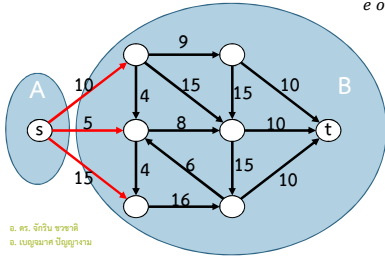


# Minimum cut problem

นิยาม st-cut (cut) เป็นการแบ่งโหนดออกเป็นเซต (A,B) ที่มี  $s \in A$  และ  $t \in B$

นิยาม ความจุของ cut คือผลรวมความจุของเส้นเชื่อมที่ออกจาก A ไป B

$$cap(A,B) = \sum_{e \text{ out of } A} c(e)$$



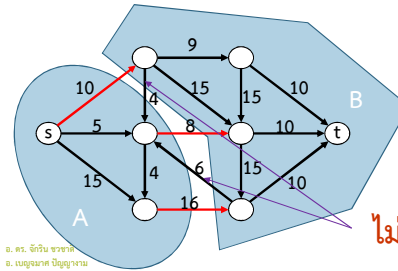
capacity=10+5+15=30

# Minimum cut problem

นิยาม st-cut (cut) เป็นการแบ่งโหนดออกเป็นเซต (A,B) ที่มี  $s \in A$  และ  $t \in B$

นิยาม ความจุของ cut คือผลรวมความจุของเส้นเชื่อมที่ออกจาก A ไป B

$$cap(A,B) = \sum_{e \text{ out of } A} c(e)$$



capacity=10+8+16=34

ไม่นับเส้นเชื่อมจาก B ไป A

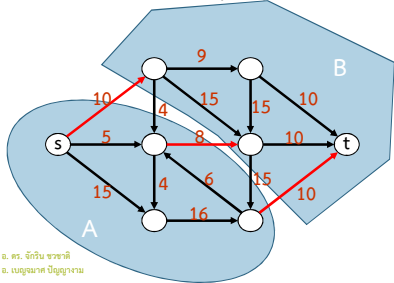
# Minimum cut problem

นิยาม st-cut (cut) เป็นการแบ่งโหนดออกเป็นเซต (A,B) ที่มี  $s \in A$  และ  $t \in B$

นิยาม ความจุของ cut คือผลรวมความจุของเส้นเชื่อมที่ออกจาก A ไป B

$$cap(A,B) = \sum_{e \text{ out of } A} c(e)$$

Min-cut problem หา cut ที่มีค่าต่ำที่สุด

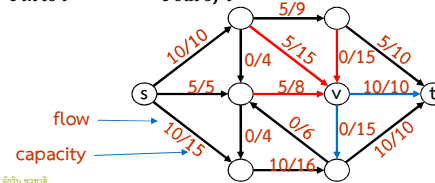


capacity=10+8+10=28

# Maximum flow problem

นิยาม st-flow (flow)  $f$  คือฟังก์ชัน  $V \times V \rightarrow R$  ที่สอดคล้องกับคุณสมบัติต่อไปนี้

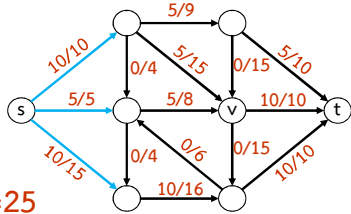
- สำหรับแต่ละเส้นเชื่อม  $e \in E$ :  
 $0 \leq f(e) \leq c(e)$  [capacity]
- สำหรับแต่ละเส้นเชื่อม  $e \in E$ :  
 $f(u,v) = -f(v,u)$  [skew symmetry]
- สำหรับแต่ละโหนด  $v \in V - \{s,t\}$ :  
 $\sum_{e \text{ in to } v} f(e) = \sum_{e \text{ out of } v} f(e)$  [flow conservation]



inflow at  $v = 5+5+0 = 10$   
outflow at  $v = 10+0 = 10$

# Maximum flow problem

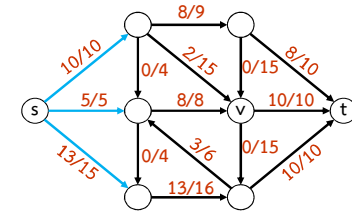
นิยาม ค่าของ flow  $f$  คือ  $val(f) = \sum_{e \text{ out of } s} f(e)$



value=5+10+10=25

# Maximum flow problem

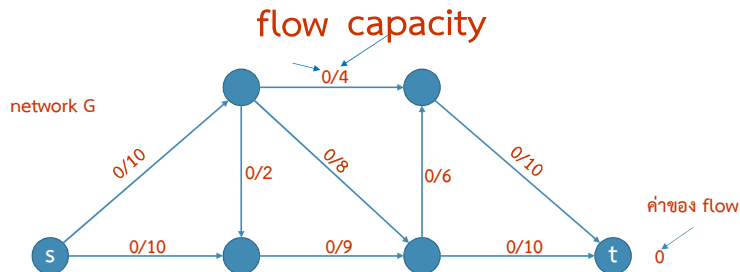
Max-flow problem หา flow ที่มีค่ามากที่สุด



value=5+10+13=28

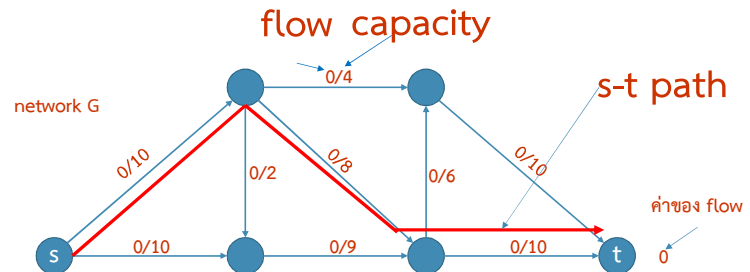
# Greedy algorithm (ลอง)

- เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- เพิ่ม (augment) flow ไปตามเส้นทาง P
- ทำซ้ำจนทำไม่ได้



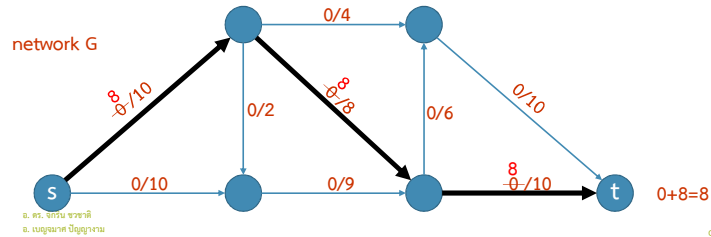
# Greedy algorithm (ลอง)

- เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- เพิ่ม (augment) flow ไปตามเส้นทาง P
- ทำซ้ำจนทำไม่ได้



### Greedy algorithm (ลอง)

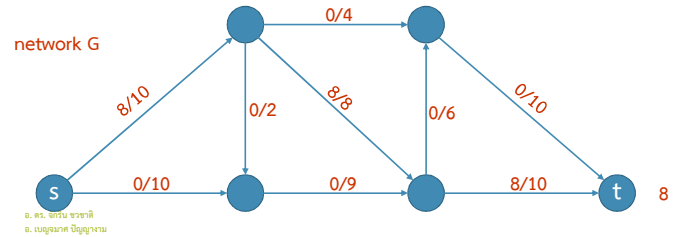
- เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- เพิ่ม (augment) flow ไปตามเส้นทาง P
- ทำซ้ำจนทำไม่ได้



อ.ดร. จารณ ชลชาติ  
อ.เบญจมาศ ปัญญางาม

### Greedy algorithm (ลอง)

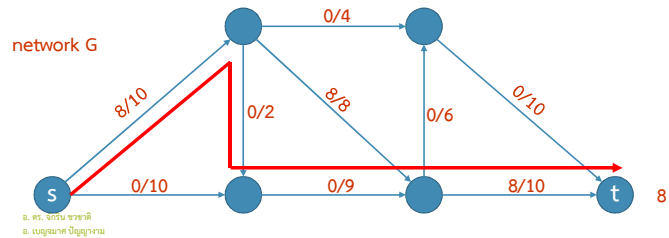
- เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- เพิ่ม (augment) flow ไปตามเส้นทาง P
- ทำซ้ำจนทำไม่ได้



อ.ดร. จารณ ชลชาติ  
อ.เบญจมาศ ปัญญางาม

### Greedy algorithm (ลอง)

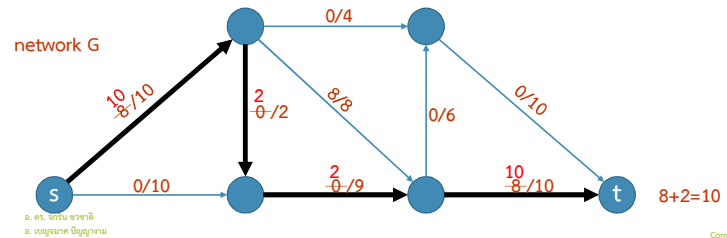
- เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- เพิ่ม (augment) flow ไปตามเส้นทาง P
- ทำซ้ำจนทำไม่ได้



อ.ดร. จารณ ชลชาติ  
อ.เบญจมาศ ปัญญางาม

### Greedy algorithm (ลอง)

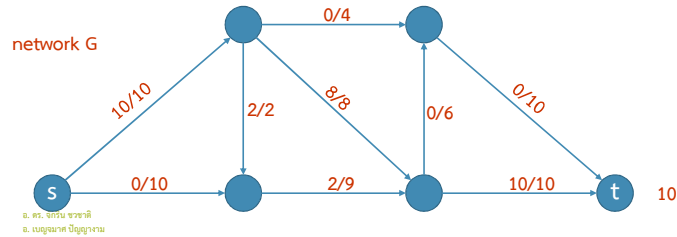
- เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- เพิ่ม (augment) flow ไปตามเส้นทาง P
- ทำซ้ำจนทำไม่ได้



อ.ดร. จารณ ชลชาติ  
อ.เบญจมาศ ปัญญางาม

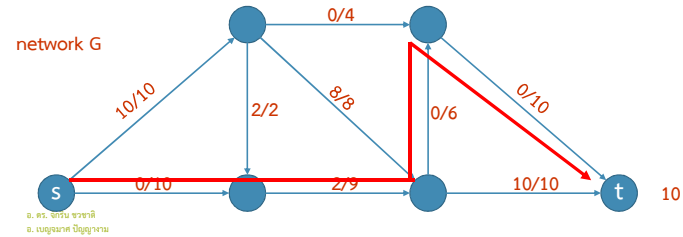
### Greedy algorithm (ลอง)

- ❑ เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- ❑ หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- ❑ เพิ่ม (augment) flow ไปตามเส้นทาง P
- ❑ ทำซ้ำจนทำไม่ได้



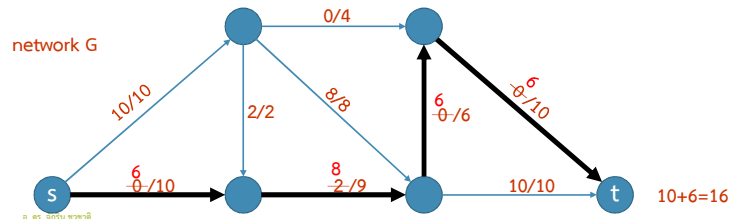
### Greedy algorithm (ลอง)

- ❑ เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- ❑ หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- ❑ เพิ่ม (augment) flow ไปตามเส้นทาง P
- ❑ ทำซ้ำจนทำไม่ได้



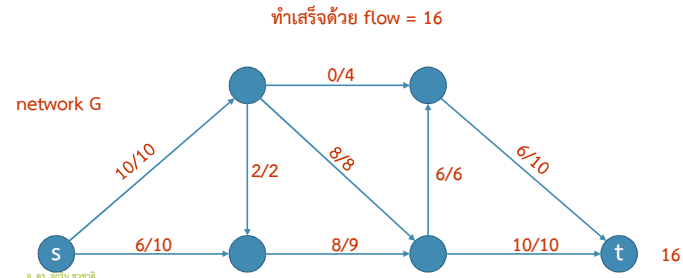
### Greedy algorithm (ลอง)

- ❑ เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- ❑ หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- ❑ เพิ่ม (augment) flow ไปตามเส้นทาง P
- ❑ ทำซ้ำจนทำไม่ได้



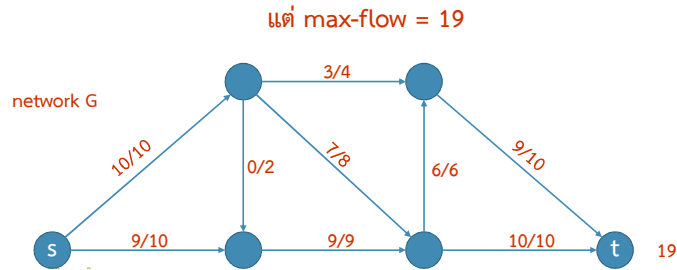
### Greedy algorithm (ลอง)

- ❑ เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- ❑ หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- ❑ เพิ่ม (augment) flow ไปตามเส้นทาง P
- ❑ ทำซ้ำจนทำไม่ได้



# Greedy algorithm (ลอง)

- ❑ เริ่มต้นให้ทุกเส้นเชื่อม  $e \in E$  มีค่า  $f(e) = 0$
- ❑ หา s-t path P ที่แต่ละเส้นเชื่อมมี  $f(e) < c(e)$
- ❑ เพิ่ม (augment) flow ไปตามเส้นทาง P
- ❑ ทำซ้ำจนทำไม่ได้

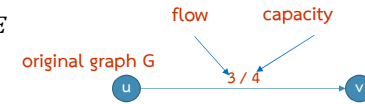


อ.ดร.จักริน ชวชาติ  
อ.เบญจมาศ ปัญญางาม

# Residual graph

Original edge:  $e = (u, v) \in E$

- ❑ Flow  $f(e)$
- ❑ Capacity  $c(e)$

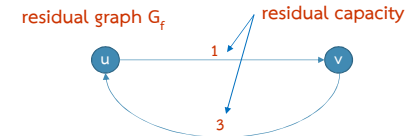


Residual edge ย้อนกลับกับ flow ที่ถูกส่ง

- ❑  $e = (u, v)$  และ  $e^R = (v, u)$

Residual capacity:

$$c_f(e) = \begin{cases} c(e) - f(e) & \text{if } e \in E \\ f(e) & \text{if } e^R \in E \end{cases}$$



อ.ดร.จักริน ชวชาติ  
อ.เบญจมาศ ปัญญางาม

# Residual graph

Residual graph:  $G_f = (V, E_f)$

- ❑ Residual edge มีค่าเป็น residual capacity ที่เป็นบวก
- ❑  $E_f = \{e: f(e) < c(e)\} \cup \{e^R: f(e) > 0\}$

อ.ดร.จักริน ชวชาติ  
อ.เบญจมาศ ปัญญางาม

# Augmenting path

นิยาม augmenting path คือ simple s-t path P ใน residual graph  $G_f$

นิยาม bottleneck capacity ของ augmenting P คือ minimum residual capacity ของเส้นเชื่อมใดๆ ใน P

คุณสมบัติที่สำคัญ: ให้  $f$  เป็น flow และให้ P เป็น augmenting path ใน  $G_f$  แล้ว  $f'$  เป็น flow และ  $val(f') = val(f) + bottleneck(G_f, P)$

```

AUGMENT( $f, c, P$ )
 $b =$  bottleneck capacity of path P
FOREACH edge  $e \in P$ 
  IF ( $e \in E$ )  $f(e) = f(e) + b$ 
  ELSE  $f(e^R) = f(e^R) - b$ 
RETURN  $f$ 

```

อ.ดร.จักริน ชวชาติ  
อ.เบญจมาศ ปัญญางาม

# Ford-Fulkerson algorithm

## Ford-Fulkerson augmenting path algorithm

- ▶ เริ่มต้นด้วย  $f(e) = 0$  สำหรับทุกเส้นเชื่อม  $e \in E$
- ▶ หา augmenting path P ใน residual graph  $G_f$
- ▶ Augment (เติม) flow ไปตาม path P
- ▶ ทำซ้ำจนทำไม่ได้

FORD-FULKERSON( $G,s,t,c$ )

FOREACH edge  $e \in E: f(e) = 0$

$G_f$  = residual graph

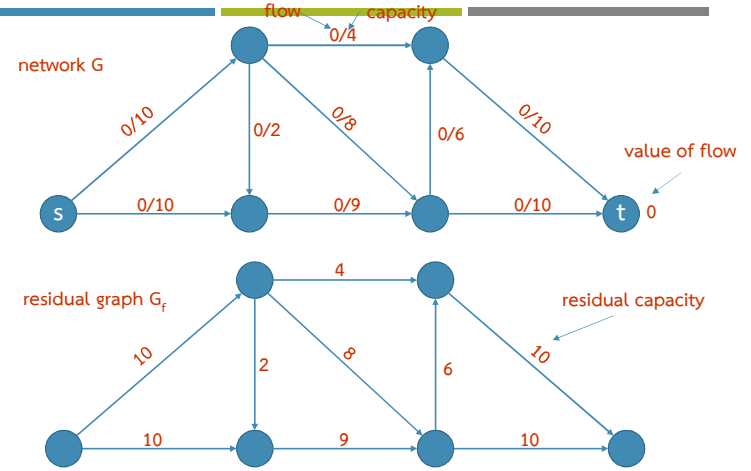
while(there exists an augmenting path P in  $G_f$ )

$f = \text{AUGMENT}(f,c,P)$

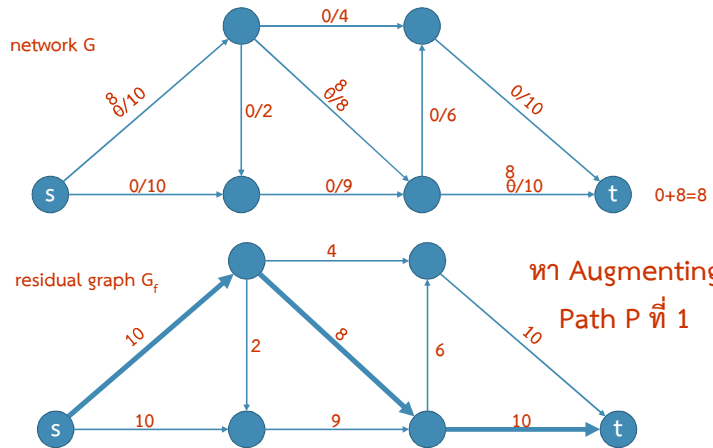
Update  $G_f$

return  $f$

# Ford-Fulkerson algorithm demo

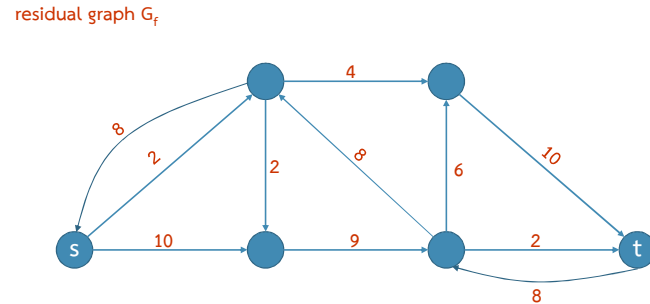


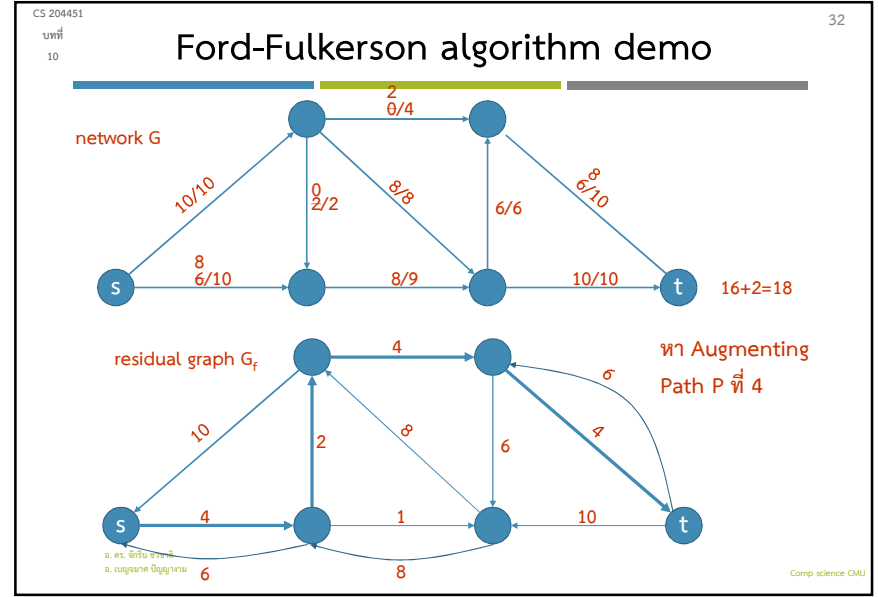
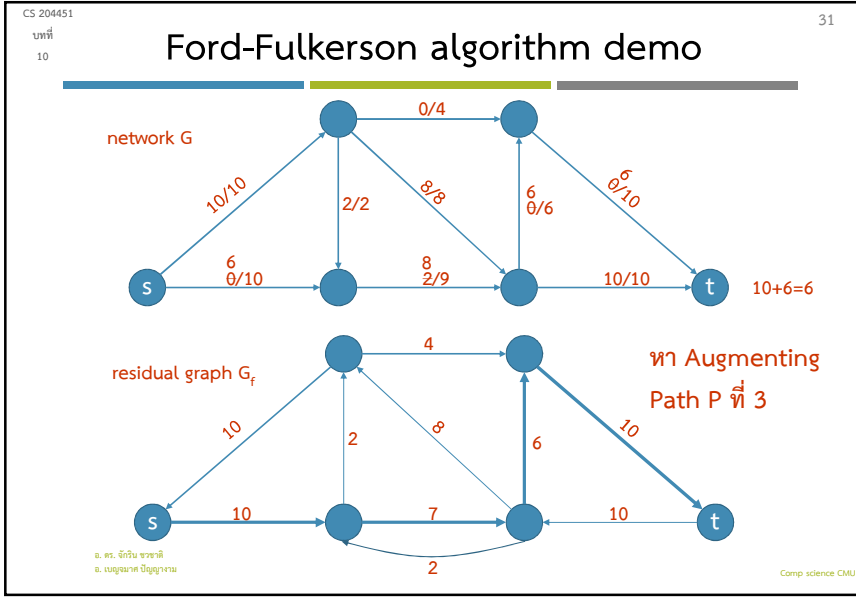
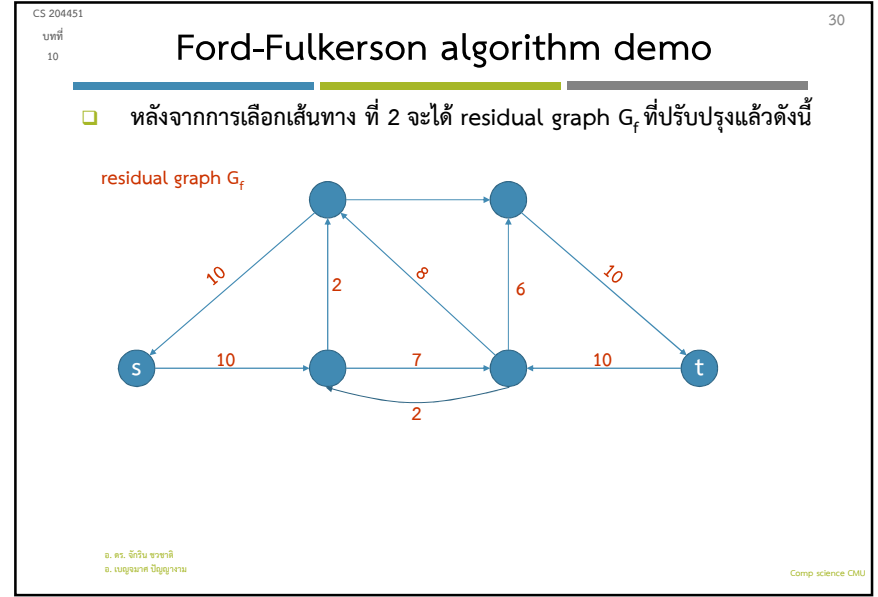
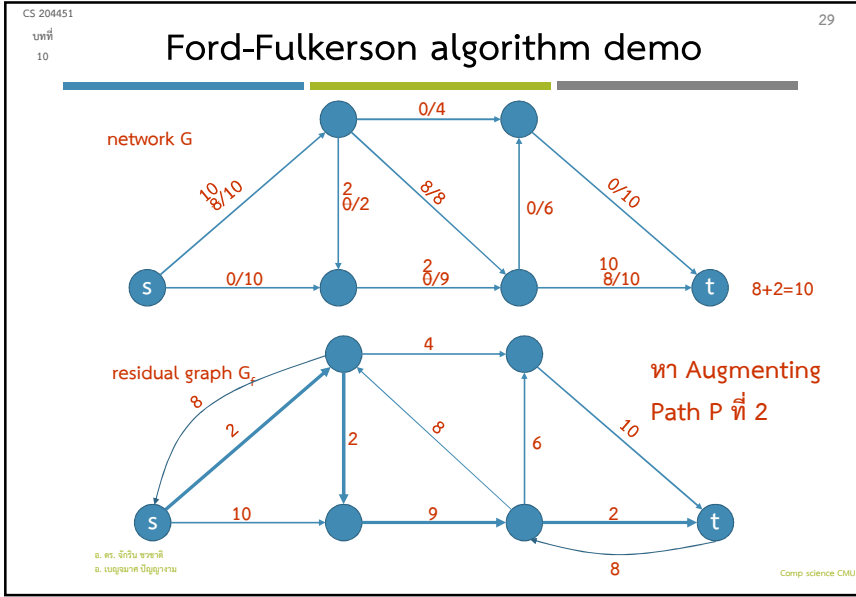
# Ford-Fulkerson algorithm demo



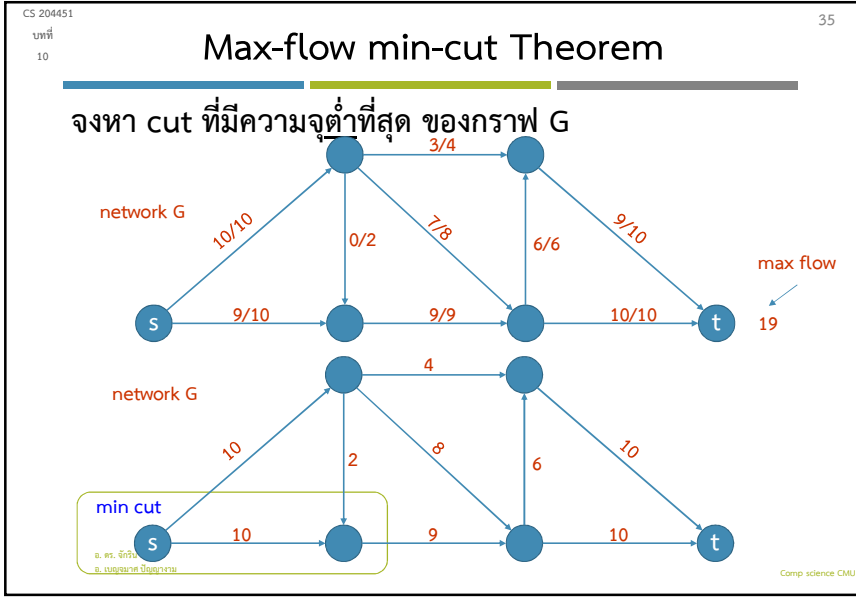
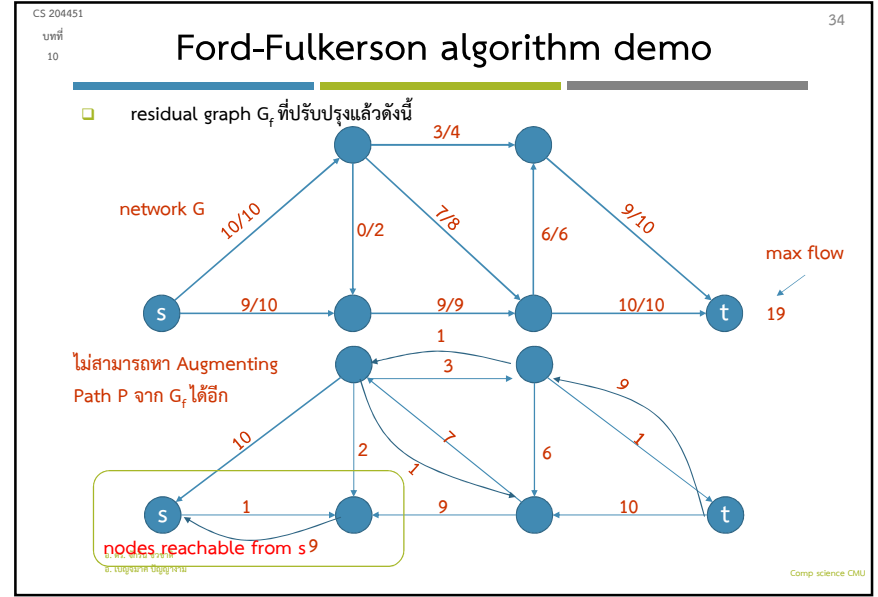
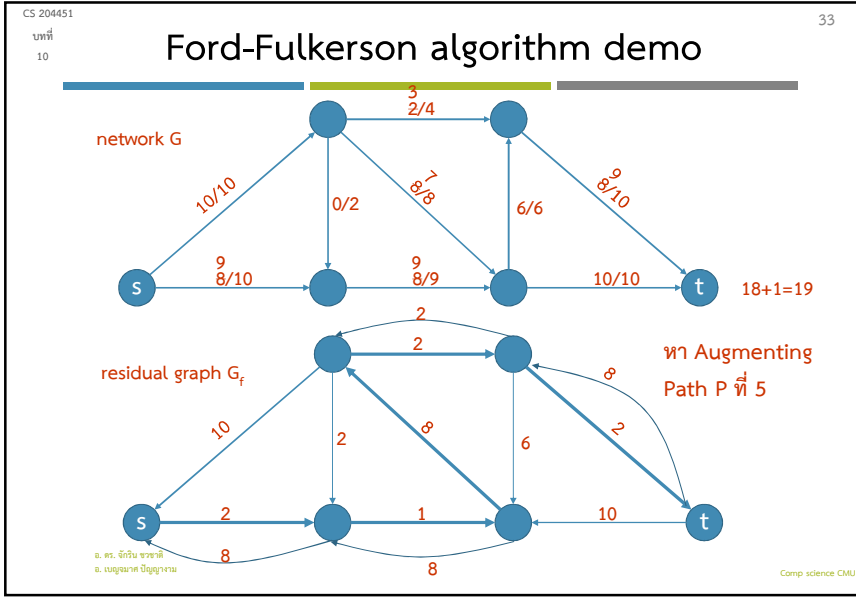
# Ford-Fulkerson algorithm demo

□ หลังจากการเลือกเส้นทาง ที่ 1 จะได้ residual graph  $G_f$  ที่ปรับปรุงแล้วดังนี้









CS 204451  
บทที่ 10

### Max-flow min-cut Theorem

36

Augmenting path theorem: flow  $f$  จะเป็น maximum flow ก็ต่อเมื่อไม่มี augmenting paths

Max-flow min-cut theorem: ค่าของ max-flow = ความจุของ min-cut

เงื่อนไข 3 ข้อต่อไปนี้เทียบเท่ากันสำหรับ flow  $f$  ใดๆ

- จะมี cut(A,B) ที่  $cap(A,B) = val(f)$
- $f$  เป็น max-flow
- ไม่มี augmenting path เมื่อเทียบกับ  $f$

อ.ดร. จักรพันธ์ ช่างชาติ  
อ.เบญจมาศ นิยมสุวรรณ

Comp science CMU