Algorithms
演算法

Graphs (4)
Maximum Flow

Professor Chien-Mo James Li 李建模
Graduate Institute of Electronics Engineering
National Taiwan University

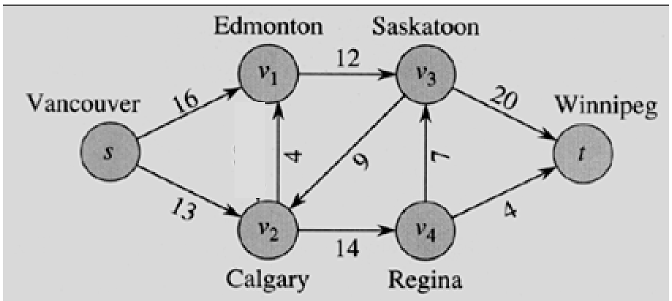
Outline

- **Elementary Graph Algorithms, CH22**
- **Minimum Spanning Trees, CH23**
- **Single Source Shortest Paths, CH24**
- **All-pairs Shortest Paths, CH25**
- **Maximum Flow, CH26***
  - **Flow Networks, 26.1**
  - **Ford-Fulkerson Method 26.2**
  - **Edmond-Karp Algorithm**
  - **Maximum Bipartite Matching 26.3**

***CH 26 is different from 2nd edition**

Flow Networks

- ***Flow Network*** ***G*** **= (*V,E*) is a directed graph**
  - **Each edge (*u,v*) has a *capacity* *c(u,v)*** $\geq$ **0**
  - **If (*u,v*)** $\notin$ ***E*, then *c*(*u,v*) = 0.**
  - **If (*u,v*)**$\in$ ***E*, then reverse edge (*v,u*)** $\notin$ ***E***
- **Two special vertices: *source vertex* *s*, *sink vertex* *t* ,**
  - **each vertex lies on a path from source to sink**
  - ***s*** ~> ***v*** ~> ***t*** **for all *v*** $\in$ ***V***
- **Imagine: vertices are junctions; edges are conduit of different sizes**
  - **capacity is an upper bound on the flow rate = units/time**



(a)

Flow* different from 2nd ed.

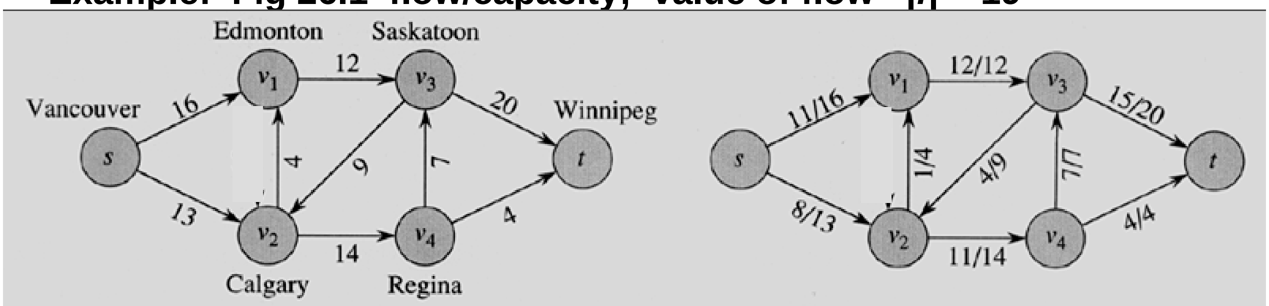
- ***Flow f*** **: *V* x *V*** $\rightarrow \Re$ **, must satisfy**
  - ***Capacity constraint*: For all *u,v*** $\in$**V, 0** $\leq$ ***f(u,v)*** $\leq$ ***c(u,v)***
  - ***Flow conservation*: For all *u*** $\in$ ***V* - {*s, t*}**
    - **total flow into *u* = total flow out of *u***

$$\underbrace{\sum_{v \in V} f(v,u)}_{\text{total flow into u}} = \underbrace{\sum_{v \in V} f(u,v)}_{\text{total flow out of u}}$$

- ***Value of flow |f|* = *net flow out of source***

$$|f| = \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s)$$

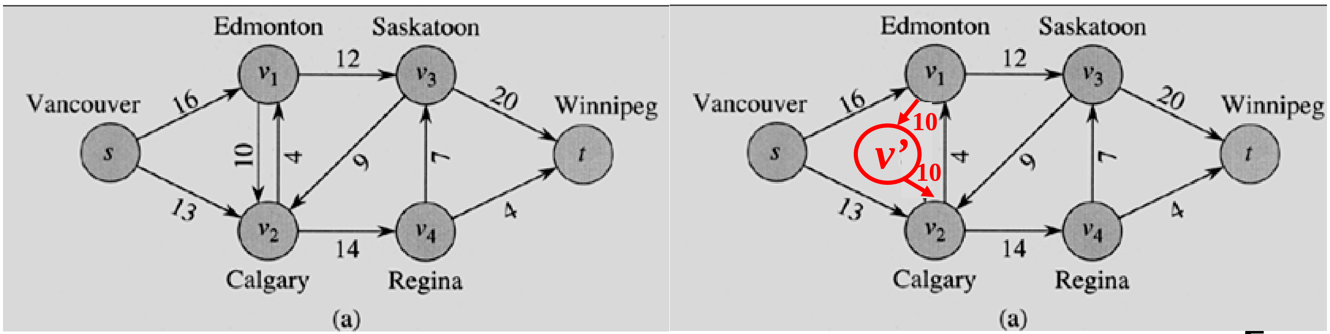
- ***Maximum flow problem***
  - **Given *G, s, t* , and *c*, find a flow whose value is maximum**
- **Example: Fig 26.1 flow/capacity; value of flow= |*f*| = 19**



(a) (b)

Antiparallel Edges

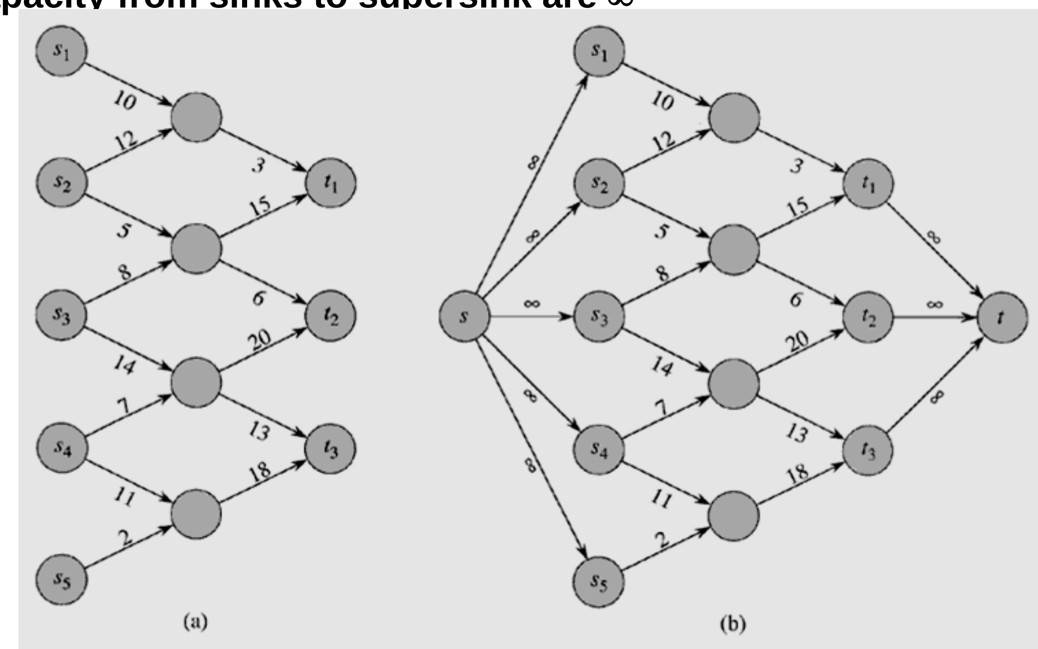
- (v_1,v_2) (v_2, v_1) are *antiparallel edges*
  - violate our assumption
- how to model this ?
  - choose one edge, say (v_1, v_2)
  - create v'
  - replace (v_1, v_2) by two new edges (v_1, v') and (v', v_2)
- Example Fig 26.2



Multiple Sources and Sinks

- What if more than one sources and sinks?
  - Add a *supersource s*, add a *supersink t*
  - Capacity from supersource to source are ∞
  - Capacity from sinks to supersink are ∞

Fig 26.3



Outline

- Elementary Graph Algorithms, CH22
- Minimum Spanning Trees, CH23
- Single Source Shortest Paths, CH24
- All-pairs Shortest Paths, CH25
- Maximum Flow, CH26
  - Flow Networks, 26.1
  - Ford-Fulkerson Method 26.2
  - Edmond-Karp Algorithm
  - Maximum Bipartite Matching 26.3

Ford-Fulkerson Method

- FF method contains three concepts
  - Residual Network
  - Augment Path
  - Cut

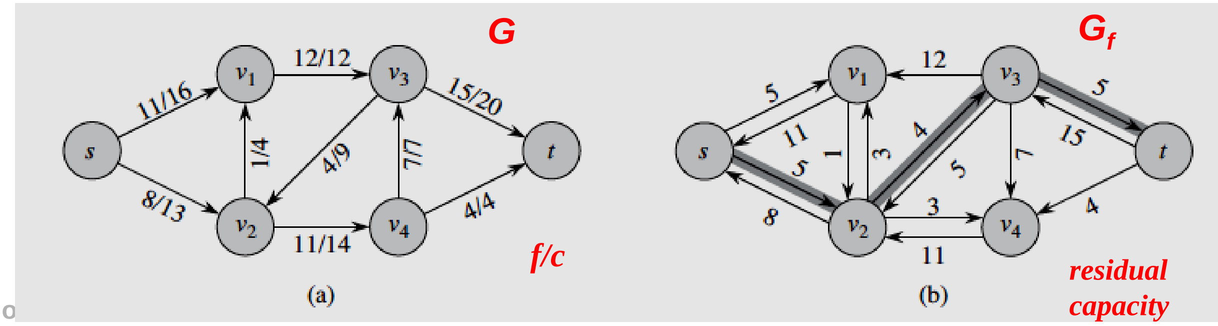
FORD-FULKERSON-METHOD(G,s,t)
initialize flow f to 0
while there exists an augmenting path p in the residual network G_f
    augment flow f along p
return f

Residual Capacity

- Given a flow f in network $G = (V,E)$. Consider a pair of vertices $u,v \in V$
- ***Residual capacity*** = additional flow we can push directly from u to v
  - sending flow back is equivalent to decreasing the flow

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E, \\ f(v,u) & \text{if } (v,u) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

- Example: Fig 26.4
  - $c_f(v_3, v_2) = 9\text{-}4\text{=}5$
  - $c_f(v_2, v_3) = 4$, why?
  - No (v_1, v_3) , why?

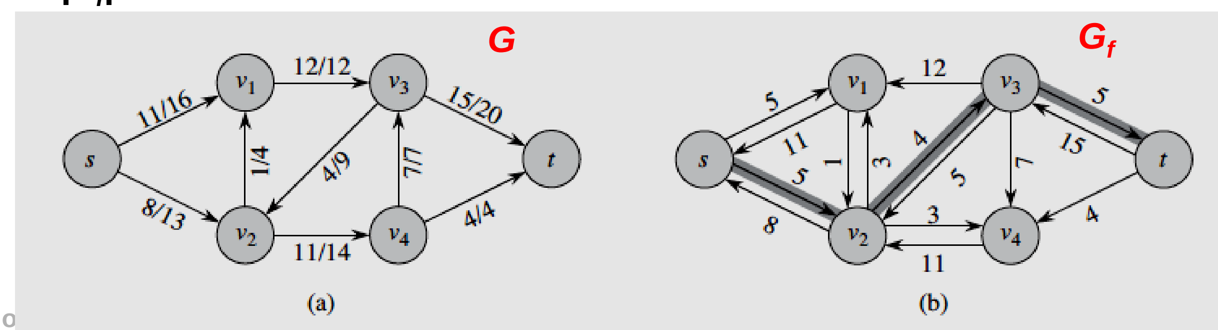


Residual Network

- ***Residual network*** $G_f = (V, E_f)$

$$E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$$

- Similar to a flow network,
  - except that it may contain antiparallel edges (u, v) and (v, u)
- Every edge $(u, v) \in E_f$ corresponds to
  - an edge $(u, v) \in E$, or an edge $(v, u) \in E$, or both
  - therefore $|E_f| \le 2|E|$
- Example: Fig. 26.4
  - $|E| = 9$
  - $|E_f| = 15$

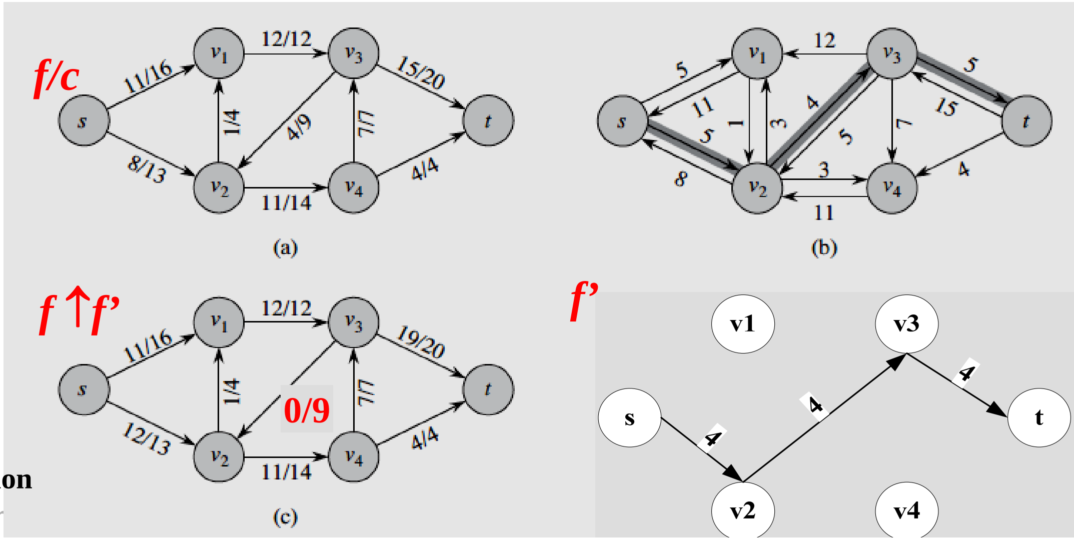


Augmentation of Flow, $f \uparrow f'$

- Given a flow f in G and a flow f' in G_f
  - ($f \uparrow f'$)= *augmentation of f* by f' equation 26.4

$$(f\uparrow f')(u,v)=\begin{cases} f(u,v)+\underbrace{f'(u,v)}_{\text{increase}}-\underbrace{f'(v,u)}_{\text{cancellation}} & ,\text{if } (u,v)\in E \\ 0 & \text{otherwise}\end{cases}$$

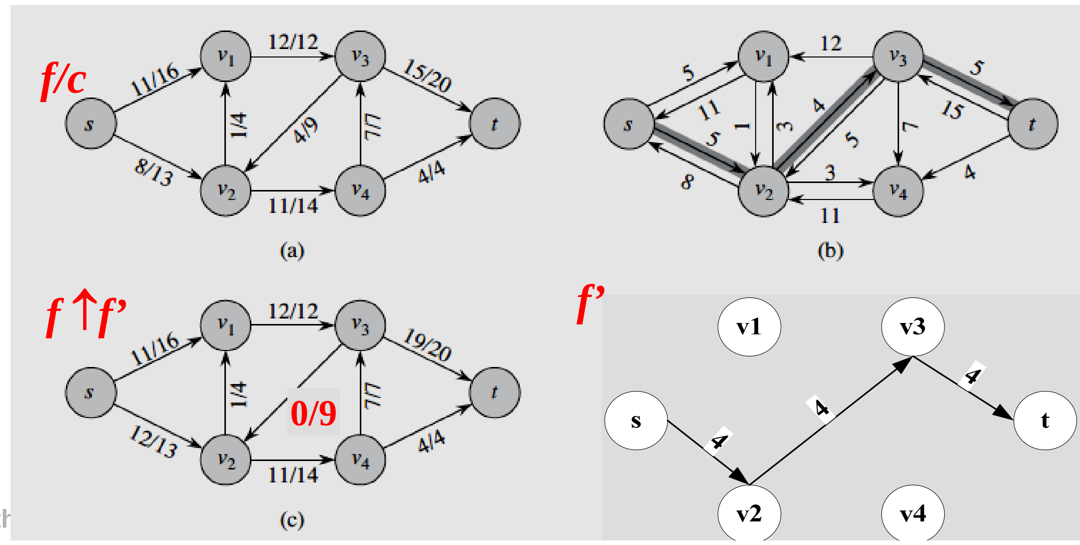
- *Cancellation :*
  - pushing flow on the reverse edge in G_f decreases the flow in G



(v_2, v_3) cancellation

Value of $|f \uparrow f'|$

- (Lemma 26.1) Given a flow network G and a flow f. Let f' be a flow in G_f . Then $f \uparrow f'$ is a flow in G with value $|f\uparrow f'| = |f|+|f'|$
- Example: Fig 26.4
  - $|f| = 19$
  - f' from $s \rightarrow v_2 \rightarrow v_3 \rightarrow t$, $|f'| = 4$
  - $|f \uparrow f'| = 19+4 = 23$



Proof of Lemma 26.1(1)

- **prove *f*↑*f'* is a flow so it obeys the capacity constraint**

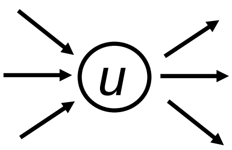
$$
\begin{aligned}
(f \uparrow f')(u,v) &= f(u,v) + f'(u,v) - f'(v,u) && \text{(by equation (26.4))} \\
&\geq f(u,v) + f'(u,v) - f(u,v) && \text{(because } f'(v,u) \leq f(u,v) \text{, why?)} \\
&= f'(u,v) \\
&\geq 0.
\end{aligned}
$$

$$
\begin{aligned}
(f \uparrow f')(u,v) &= f(u,v) + f'(u,v) - f'(v,u) && \text{(by equation (26.4))} \\
&\leq f(u,v) + f'(u,v) && \text{(because flows are nonnegative)} \\
&\leq f(u,v) + c_f(u,v) && \text{(capacity constraint)} \\
&= f(u,v) + c(u,v) - f(u,v) && \text{(definition of } c_f \text{)} \\
&= c(u,v)
\end{aligned}
$$

Proof of Lemma 26.1(2)

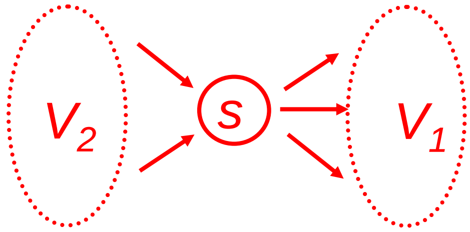
- **prove *f*↑*f'* is a flow so it obeys the flow conservation**

$$
\begin{aligned}
\sum_{v\in V}(f \uparrow f')(u,v) &= \sum_{v\in V}\big(f(u,v) + f'(u,v) - f'(v,u)\big) \\
&= \sum_{v\in V} f(u,v) + \sum_{v\in V} f'(u,v) - \sum_{v\in V} f'(v,u) \\
&= \sum_{v\in V} f(v,u) + \sum_{v\in V} f'(v,u) - \sum_{v\in V} f'(u,v) \\
&= \sum_{v\in V}\big(f(v,u) + f'(v,u) - f'(u,v)\big) \\
&= \sum_{v\in V}(f \uparrow f')(v,u)
\end{aligned}
$$



Proof of Lemma 26.1(3)

- **prove $|f\uparrow f'| = |f|+|f'|$**



$$
\begin{aligned}
\left|f \uparrow f'\right| &= \sum_{v\in V}(f \uparrow f')(s,v) - \sum_{v\in V}(f \uparrow f')(v,s) \\
&= \sum_{v\in V_1}(f \uparrow f')(s,v) - \sum_{v\in V_2}(f \uparrow f')(v,s) \\
&= \sum_{v\in V_1}\big(f(s,v)+f'(s,v)-f'(v,s)\big) - \sum_{v\in V_2}\big(f(v,s)+f'(v,s)-f'(s,v)\big) \\
&= \sum_{v\in V_1} f(s,v) + \sum_{v\in V_1} f'(s,v) - \sum_{v\in V_1} f'(v,s) \\
&\quad - \sum_{v\in V_2} f(v,s) - \sum_{v\in V_2} f'(v,s) + \sum_{v\in V_2} f'(s,v) \\
&= \sum_{v\in V_1} f(s,v) - \sum_{v\in V_2} f(v,s) \\
&\quad + \sum_{v\in V_1} f'(s,v) + \sum_{v\in V_2} f'(s,v) - \sum_{v\in V_1} f'(v,s) - \sum_{v\in V_2} f'(v,s) \\
&= \sum_{v\in V_1} f(s,v) - \sum_{v\in V_2} f(v,s) + \sum_{v\in V_1\cup V_2} f'(s,v) - \sum_{v\in V_1\cup V_2} f'(v,s)
\end{aligned}
$$

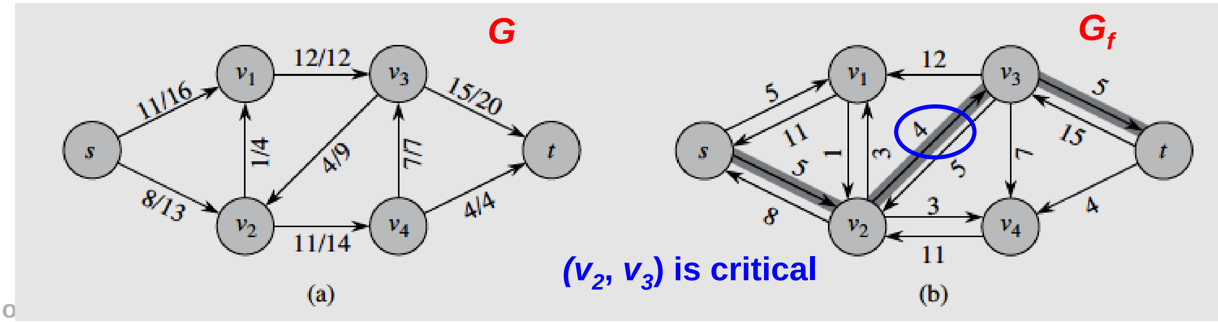
Cont'd

- **exercise 26.2-1**

$$
\begin{aligned}
|f \uparrow f'| &= \sum_{v\in V_1} f(s,v) - \sum_{v\in V_2} f(v,s) + \sum_{v\in V_1\cup V_2} f'(s,v) - \sum_{v\in V_1\cup V_2} f'(v,s) \\
&= \sum_{v\in V} f(s,v) - \sum_{v\in V} f(v,s) + \sum_{v\in V} f'(s,v) - \sum_{v\in V} f'(v,s) \\
&= |f| + |f'|
\end{aligned}
$$

Augmenting Path

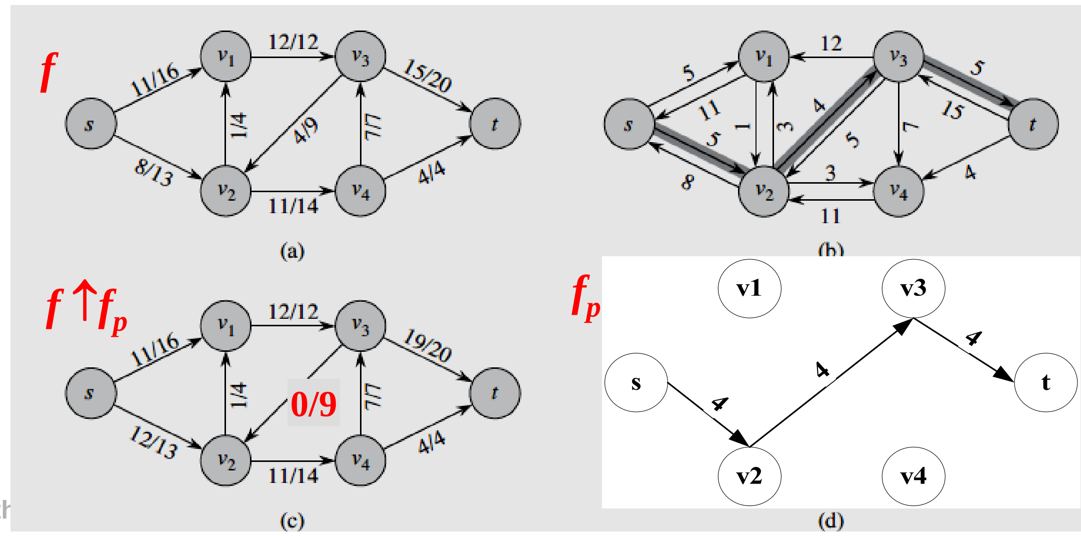
- ***Augmenting path* p is a simple path from s ~> t in G_f**
  - **p admits more flow along each edge**
    - **a sequence of pipes through which we can push more flow**
  - **How much more flow can we push from s to t along p?**
    - ***residual capacity of p*** $c_f(p) = \min\{c_f(u,v) : (u,v) \text{ is on } p\}$
    - **smallest residual capacity of all edges on this path**
- **Example: Fig 26.4**
  - **Augmenting Path p = <s, v_2, v_3, t >**
  - **$c_f(p)$ = 4**
- **(u, v) is called *critical on augmenting path p* if $c_f(u, v) = c_f(p)$**



- **(Lemma 26.2)Given flow network G and flow f. Let p be an augmenting path in G_f. f_p is a flow in G_f with value $|f_p| = c_f(p) > 0$**

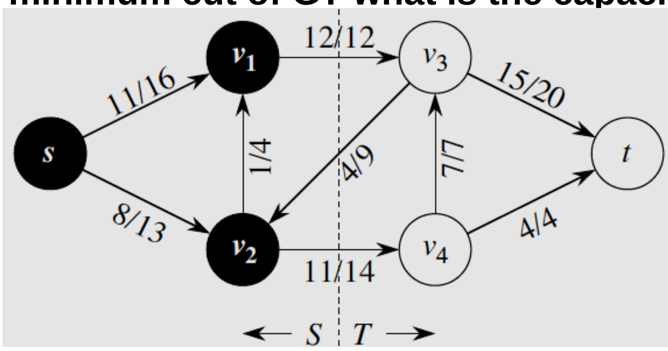
$$f_p(u,v) = \begin{cases} c_f(p) & \text{if } (u,v) \text{ is on } p, \\ 0 & \text{otherwise} \end{cases} \qquad \text{eq. 26.8}$$

- **(Corollary 26.3) Given flow network G and flow f, and augmenting path p in G_f. Then $f \uparrow f_p$ is a flow in G with value $|f \uparrow f_p| = |f| + |f_p| > |f|$**



CUT

- ***Cut*** **(*S*, *T*) of flow network *G* =(*V*,*E*) is a partition of *V* into *S* and *T***
  - ***T* = *V*-*S* such that *s* ∈ *S* and *t* ∈ *T***
- ***Net flow across cut*** **(*S*, *T*) is** $f(S,T)=\sum_{u\in S}\sum_{v\in T} f(u,v)-\sum_{u\in S}\sum_{v\in T} f(v,u)$
- ***Capacity of cut*** **(*S*,*T*) is** $c(S,T)=\sum_{u\in S}\sum_{v\in T} c(u,v)$
- ***Minimum cut of G*** **is a cut whose capacity is minimum over all cuts**
- **Example: Fig 26.5 cut {*s*, v_1, v_2} {v_3, v_4, *t*}**
  - **capacity of cut = 12+14 = 26; net flow cross cut = 12+11-4=19**
  - **what is the minimum cut of G? what is the capacity of the cut?**



Lemma 26.4

- **For any cut (*S*, *T*), the net flow across cut** ***f*(*S*, *T*) = |*f*|**
  - **Proof**

$$\sum_{v\in V} f(u,v)-\sum_{v\in V} f(v,u)=0 \qquad \boxed{\text{flow conservation, } u\in V-\{s,t\}}$$

$$|f|=\underbrace{\sum_{v\in V} f(s,v)-\sum_{v\in V} f(v,s)}_{eq.26.1}+\underbrace{\sum_{u\in S-\{s\}}\left[\sum_{v\in V} f(u,v)-\sum_{v\in V} f(v,u)\right]}_{=0}$$

$$|f|=\sum_{v\in V} f(s,v)-\sum_{v\in V} f(v,s)+\sum_{u\in S-\{s\}}\sum_{v\in V} f(u,v)-\sum_{u\in S-\{s\}}\sum_{v\in V} f(v,u)$$

$$=\sum_{v\in V}\left(f(s,v)+\sum_{u\in S-\{s\}} f(u,v)\right)-\sum_{v\in V}\left(f(v,s)+\sum_{u\in S-\{s\}} f(v,u)\right)$$

$$=\sum_{v\in V}\sum_{u\in S} f(u,v)-\sum_{v\in V}\sum_{u\in S} f(v,u)$$

Lemma 26.4 (2)

- **Proof (cont'd)**
- **because $V = S \cup T$, and $S \cap T = \varnothing$**
  - **split summation over V into summation over S and T**

$$
\begin{aligned}
|f| &= \sum_{v\in S}\sum_{u\in S} f(u,v) + \sum_{v\in T}\sum_{u\in S} f(u,v) - \sum_{v\in S}\sum_{u\in S} f(v,u) - \sum_{v\in T}\sum_{u\in S} f(v,u) \\
&= \sum_{v\in T}\sum_{u\in S} f(u,v) - \sum_{v\in T}\sum_{u\in S} f(v,u) + \underbrace{\left(\sum_{v\in S}\sum_{u\in S} f(u,v) - \sum_{v\in S}\sum_{u\in S} f(v,u)\right)}_{=0}
\end{aligned}
$$

$$
\begin{aligned}
|f| &= \sum_{u\in S}\sum_{v\in T} f(u,v) - \sum_{u\in S}\sum_{v\in T} f(v,u) \\
&= f(S,T)
\end{aligned}
$$

Corollary 26.5

- **The value of any flow $\leq$ capacity of any cut**
  - **Proof**

$$
\begin{aligned}
|f| &= f(S,T) \\
&= \sum_{u\in S}\sum_{v\in T} f(u,v) - \sum_{u\in S}\sum_{v\in T} f(v,u) \\
&\leq \sum_{u\in S}\sum_{v\in T} f(u,v) \\
&\leq \sum_{u\in S}\sum_{v\in T} c(u,v) \\
&= c(S,T)
\end{aligned}
$$

- **Therefore, maximum flow $\leq$ capacity of minimum cut**

Max-flow Min-Cut Theorem (1)

- (Theorem 26.6) The following are equivalent:
  1. f is a maximum flow
  2. G_f has no augmenting path
  3. $|f| = c(S, T)$ for some cut (S, T)

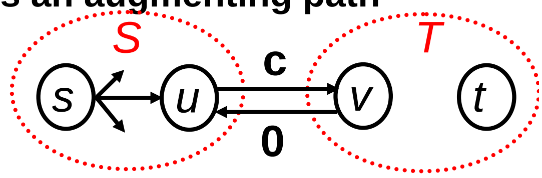
  - Proof: 1➔2 contrapositive
    - assume G_f has an augmenting path, and f is a maximum flow
    - by corollary 26.3, $f \uparrow f_p$ is a flow in G with value $|f|+|f_p| > |f|$
      - so f is not a maximum flow, conflict!
  - Proof: 3➔1
    - (corollary 26.5) $|f| \le c(S, T)$
    - so $|f| = c(S, T)$ means f is a max flow

Max-flow Min-Cut Theorem (2)

- Proof 2➔3
  - Suppose G_f has no augmenting path
  - Let $S = \{v \in V$: there exists a path $s \rightsquigarrow v$ in $G_f\}$, $T = V-S$
    - Must have $t \in T$; otherwise there is an augmenting path
  - Therefore, (S, T) is a cut
  - Consider $u \in S$ and $v \in T$
    - If $(v,u) \in E$, then $c_f(u,v) = f(v,u) = 0$
      - otherwise, $c_f(u,v)=f(v,u)>0$ ➔ $(u,v) \in E_f$ ➔ $v \in S$
    - If $(u,v) \in E$, then $c_f(u,v)=0$ ➔ $f(u,v) = c(u,v)$
      - otherwise, $(u,v) \in E_f$ ➔ $v \in S$
    - If both (u,v) $(v,u) \notin E$, must have $f(u,v) = f(v,u) = 0$
  - therefore $|f| = f(S,T) = \sum_{u\in S}\sum_{v\in T} f(u,v) - \sum_{v\in T}\sum_{u\in S} f(v,u)$

$$= \sum_{u\in S}\sum_{v\in T} c(u,v) - \sum_{v\in T}\sum_{u\in S} 0$$

$$= c(S,T)$$



Basic Ford Fulkerson

- Keep augmenting flow along an augmenting path
  - until there is no augmenting path

```
FORD-FULKERSON(G,s,t)
for each edge (u,v) ∈ G.E    // initialize
        (u,v).f = 0
while there exists a path p from s to t in the residual network G_f
        c_f (p) = min {c_f (u,v) : (u,v) is in p}
        for  each edge (u,v) in p
                if (u,v) ∈ E
                        (u,v).f = (u,v).f + c_f (p)    // increase
                else  (v,u).f = (v,u).f − c_f (p)    // cancellation
```

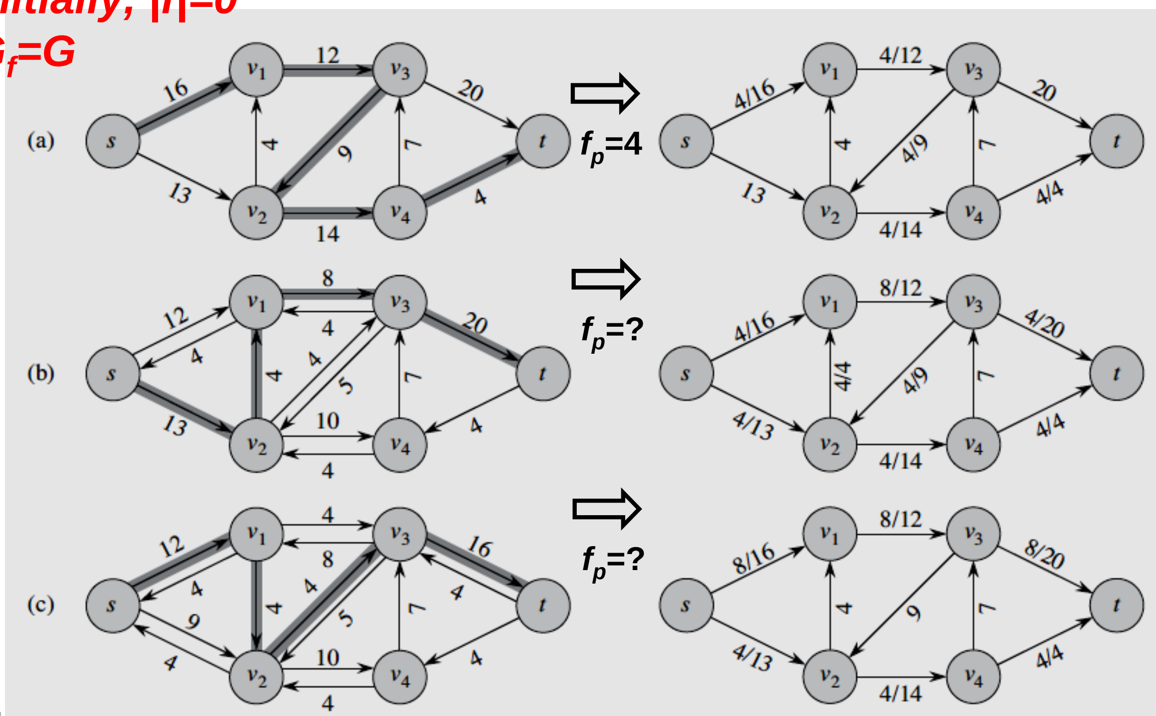

Example

- Fig 26.6

G_f G

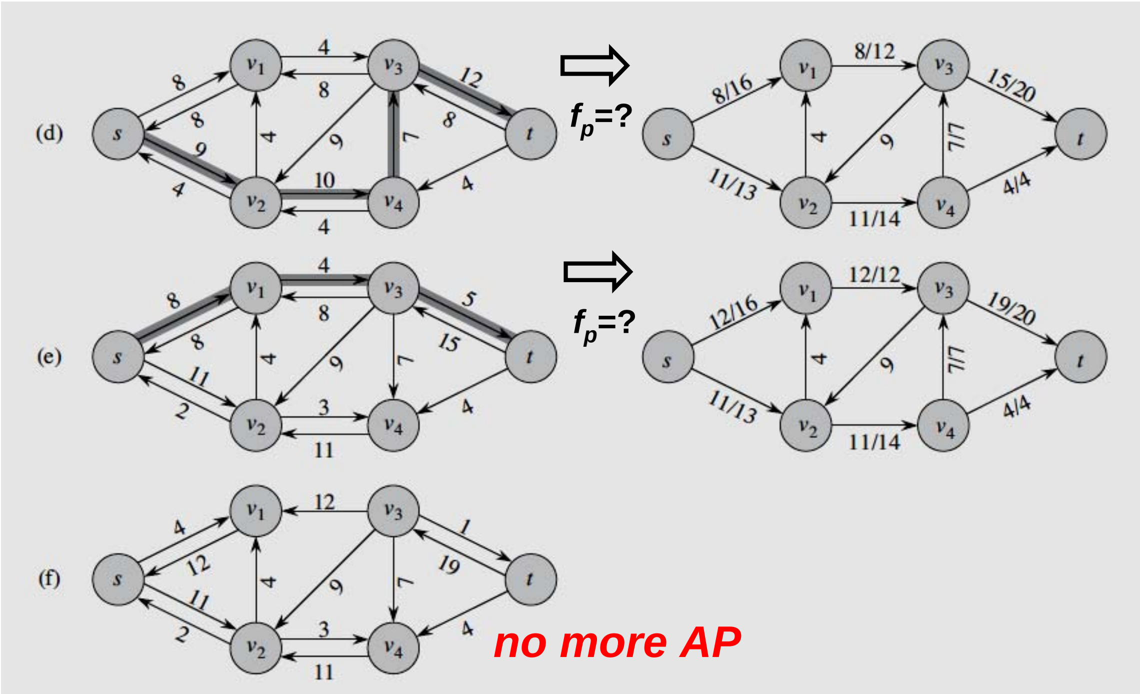
initially, |f|=0

G_f=G



Example (cont'd)

- **Fig 26.6 (cont'd) Q1: maximum flow =?**
- **Q2: can you find the min CUT?**



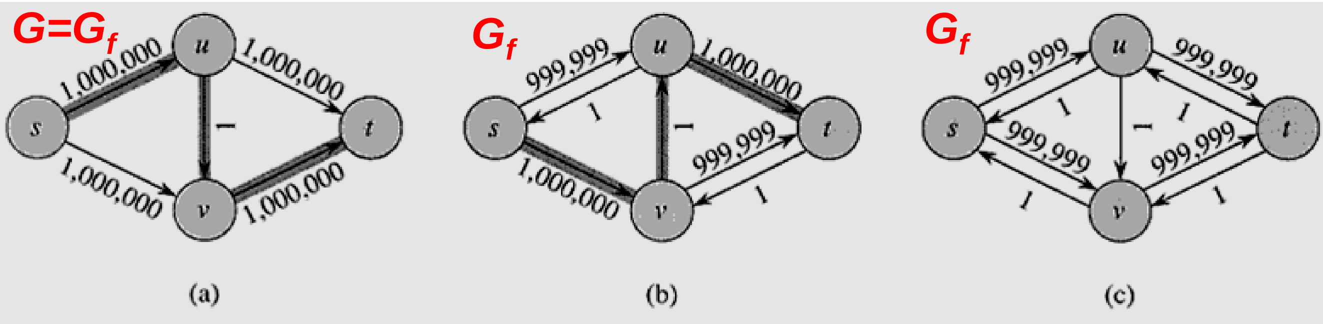
Time Complexity

```
FORD-FULKERSON(G,s,t)
1  for each edge (u,v) ∈ G.E
2        (u,v).f = 0
3  while there exists a path p from s to t in the residual network G_f
4        c_f (p) = min {c_f (u,v) : (u,v) is in p}
5        for each edge (u,v) in p
6              if (u,v) ∈ E
7   O(E)             (u,v).f = (u,v).f + c_f (p)
8              else (v,u).f = (v,u).f − c_f (p)
```

- **line 3: finding G_f using BFS or DFS**
  - **O(V+E') = O(E)**
  - **E' = {(u,v): (u,v)∈E or (v,u)∈E}**
- **line 3~8: while loop**
  - **Assume capacities are integers. Assume max flow is f***
  - **each iteration increase flow by at least 1**
    - **needs |f*| iterations**
- **Time complexity = O(E |f*|)**

Disadvantage

- FF running time is NOT polynomial in input size.
  - It depends on |*f**|, which is not a function of *V* and *E*
- worst case example: Fig 26.7
  - need 2,000,000 times augmentations!



- can we do better?

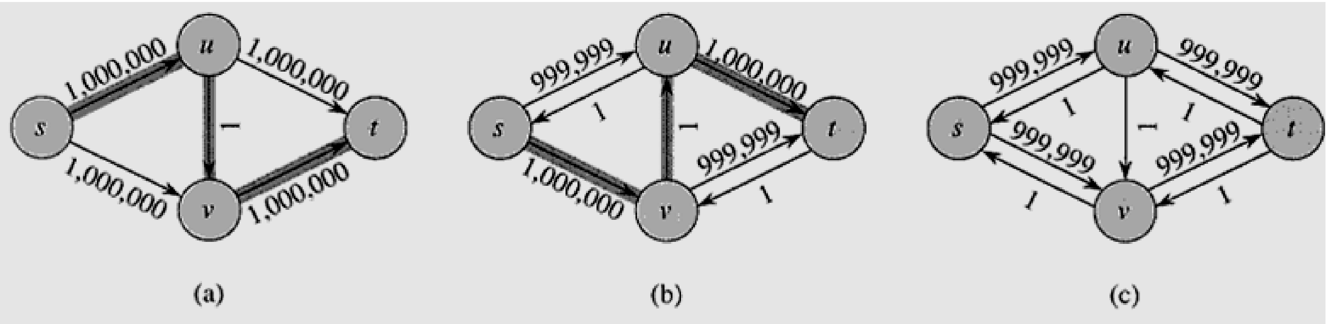
Outline

- Elementary Graph Algorithms, CH22
- Minimum Spanning Trees, CH23
- Single Source Shortest Paths, CH24
- All-pairs Shortest Paths, CH25
- <u>Maximum Flow, CH26</u>
  - Flow Networks, 26.1
  - Ford-Fulkerson Method 26.2
  - <u>Edmond-Karp Algorithm 26.2</u>
  - Maximum Bipartite Matching 26.3

Edmonds-Karp Algorithm

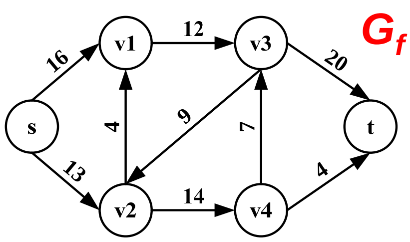
- Do FORD-FULKERSON, but compute augmenting paths by ***BFS***
  - AP are shortest paths ***s* ~> *t*** in G_f, with ***unit edge weights***
  - time complexity $O(VE^2)$ (Theorem 26.8)
- push-relabel algorithm is even better $O(V^3)$
  - 26.4 – 26.5 *not in exam

- Exercise 1
  - show Edmonds-Karp is better than Ford-Fulkerson



Exercise 2

- Use Edmonds-Karp to find max flow
  - Q1: how many iterations do we need? Q2: what is the max flow?



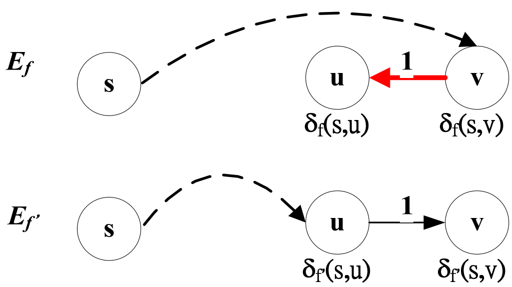
Lemma 26.7 (1)

- Let $\delta_f(u, v)$= shortest path distance from *u* to *v* in G_f
  - assume unit edge weights
- For all $v \in V$- {*s*, *t*}, $\delta_f(u, v)$ *increases monotonically* with each flow augmentation.
  - Proof by contradiction
  - Suppose there exists $v \in V$- {*s*, *t*} such that some flow augmentation causes shortest path distance *s* ~>*v* to decrease
  - Let *f* = flow before the *first augmentation* that causes a shortest-path distance to decrease. Let *f'* = the flow afterward
  - Let *v* be a vertex with minimum $\delta_{f'}(s, v)$ whose distance was decreased by the augmentation $\delta_{f'}(s,v) < \delta_f(s,v)$
  - Let a shortest path *s* to *v* in $G_{f'}$ be *s*~>*u* →*v*
    - so $(u, v) \in E_{f'}$, $\delta_{f'}(s,u) = \delta_{f'}(s,v) - 1 \quad (26.12)$
  - Because of how we chose *v*, we know the distance from *s* to *u* does not decrease $\delta_{f'}(s,u) \ge \delta_f(s,u) \quad (26.13)$

Lemma 26.7 (2)

- Claim $(u, v) \notin E_f$ why? if $(u, v) \in E_f$ then

$$\delta_f(s,v) \le \delta_f(s,u)+1 \quad \text{(by Lemma 24.10, the triangle inequality)}$$
$$\le \delta_{f'}(s,u)+1 \quad \text{(by inequality (26.13))}$$
$$= \delta_{f'}(s,v) \quad \text{(by equation (26.12))}$$



  - contradict assumption
    - $\delta_{f'}(s, v) < \delta_f(s, v)$
- How can $(u, v) \notin E_f$ but $(u, v) \in E_{f'}$?
  - The augmentation must have increased flow *v* to *u*
  - Since Edmonds-Karp augments along shortest paths, the shortest path *s* to *u* in G_f has (*v*, *u*) as its last edge

$$\delta_f(s,v) = \delta_f(s,u) - 1 \quad (v,u) \text{ is last edge on shortest path in } G_f$$
$$\le \delta_{f'}(s,u) - 1 \quad \text{(by inequality (26.13))}$$
$$= \delta_{f'}(s,v) - 2 \quad \text{(by equation (26.12))}$$

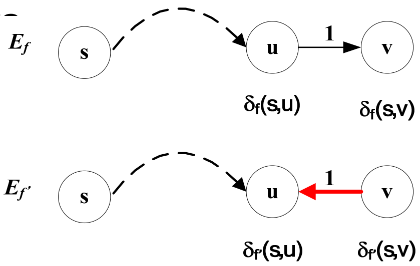
  - contradict assumption $\delta_{f'}(s, v) < \delta_f(s, v)$
- Therefore no *v* exist such that $\delta_f(s, v)$ decreases

Theorem 26.8 (1)

- Edmonds-Karp performs O(*VE*) augmentations
  - Proof:
  - Let *p* be an augmenting path and (*u, v*) is critical
    - it disappears from residual network after augmenting along *p*
  - claim: each of the |E| edges can become critical ≤ |V|/2 times
  - Consider *u,v*∈*V* such that either (*u, v*)∈*E* or (*v, u*)∈*E* or both
  - when (*u, v*) becomes critical first time

    $$\delta_f(s,v) = \delta_f(s,u)+1$$
  - After augmentation, (*u, v*) disappears from residual network
  - (*u, v*) won't reappear in G_f until flow from *u* to v decreases, which happens only if (*v, u*) is on an augmenting path in $G_{f'}$
    - Let *f'* be the flow when this occur

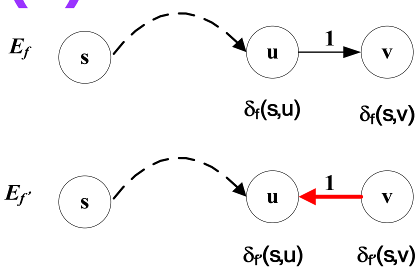
      $$\delta_{f'}(s,u) = \delta_{f'}(s,v)+1$$



Theorem 26.8 (2)

- by Lemma 26.7

$$\begin{aligned}\delta_{f'}(s,u) &= \delta_{f'}(s,v)+1 \\ &\geq \delta_f(s,v)+1 \\ &= \delta_f(s,u)+2\end{aligned}$$



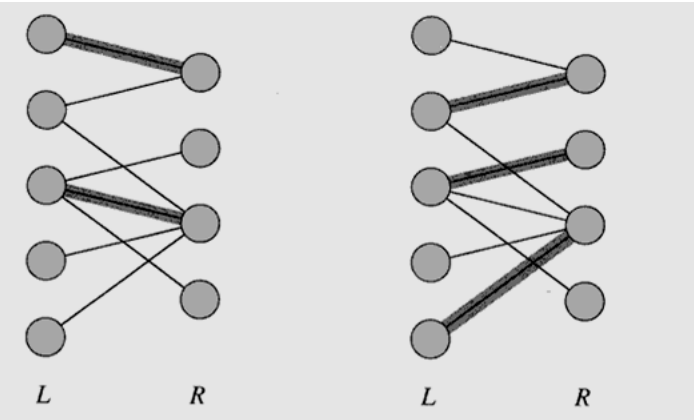
- Each time, δ_f(*s, u*) increases by at least 2
- Initially, δ_f(*s, u*) = 0,
- eventually, δ_f(*s, u*) ≤|*V*|-2
  - augmenting path can't have *s*, and *t* as intermediate vertices
  - *u* can become critical less than (|*V*|-2) /2 =|*V*|/2 - 1 times
  - totally, *u* can become critical less than |*V*|/2 times
- Since O(*E*) pairs of vertices have an edge between them in G_f
  - Each AP has at least 1 critical edge
  - total O(*VE*) augmentations
- Use BFS to find each AP in O(*E*) time
  - Edmons-Karp is O(VE^2)

Outline

- Elementary Graph Algorithms, CH22
- Minimum Spanning Trees, CH23
- Single Source Shortest Paths, CH24
- All-pairs Shortest Paths, CH25
- Maximum Flow, CH26
  - Flow Networks, 26.1
  - Ford-Fulkerson Method 26.2
  - Edmond-Karp Algorithm
  - Maximum Bipartite Matching 26.3

Bipartite Matching

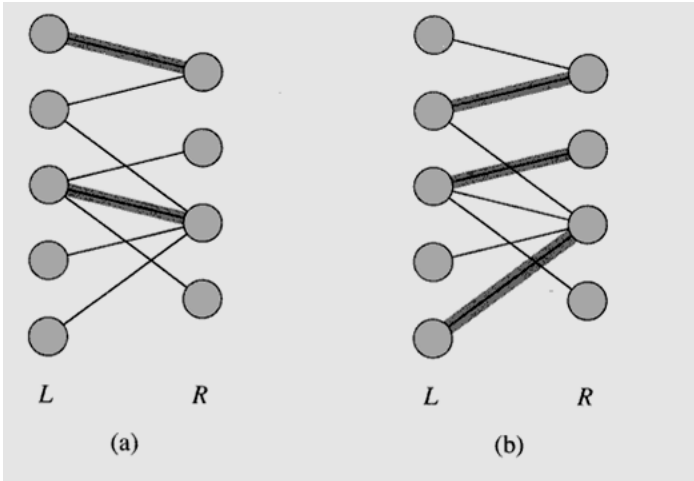
- Undirected $G = (V, E)$ is ***bipartite*** if we can
  - partition $V = L \cup R$ such that all edges go between L and R
- A ***matching*** is a subset of edges $M \subseteq E$ such that
  - for all $v \in V$, one edge of M is incident on v
  - ***cardinality*** = size of M = $|M|$
- Vertex v is *matched* if an edge of M is incident on it
  - otherwise unmatched
- Example:
  - Fig 26.8
  - (a) cardinality =2
  - (b) cardinality =3



(a) (b)

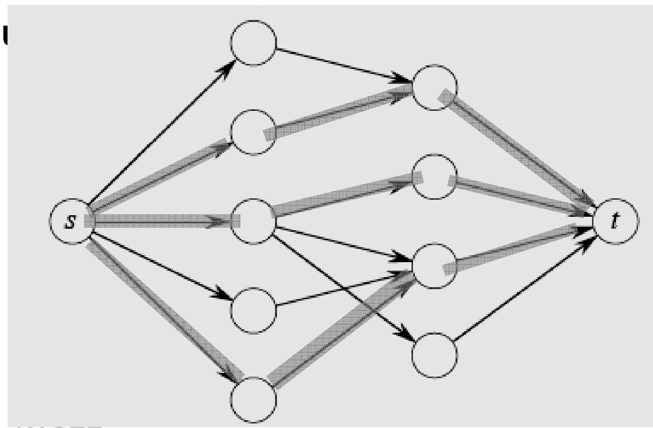
Maximum Bipartite Matching

- ***Maximum bipartite matching***: a matching of *maximum cardinality*
  - M is a maximum matching if $|M| \geq |M'|$ for all matching M'
- **Applications: machine-task matching**
  - L = machines, R = tasks
  - edge (u, v) means machine u is capable of performing a task v
  - MBM find maximum number of tasks
- **Example**
  - (b) is MBM



Corresponding Flow Network

- ***Corresponding flow network G'=(V', E')***
  - $V' = V \cup \{s, t\}$ $\quad E' = \underbrace{\{(s,u) : u \in L\}}_{\text{edges from } s} \cup \underbrace{\{(u,v) : (u,v) \in E\}}_{\text{original edges}} \cup \underbrace{\{(v,t) : v \in R\}}_{\text{edges to } t}$
  - $c(u,v) = 1$ for all $(u, v) \in E'$
  - $|E'| = |E| + |V|$
- **Each vertex in V has at least one incident edge, $|E| \geq |V|/2$**
  - $|E'| = |E| + |V| \leq 3|E|$. therefore, $|E'| = \Theta|E|$
- **Idea: a flow in G' correspond to a matching in G**
  - solve MBM using Ford-Fu
- **Example Fig 26.8c**
  - max flow = 3
  - MBM cardinality =3



Lemma 26.9

- Assume *integer-valued flow*: *f(u,v)* is integer are for all edges (*u, v*)
- If *M* is a matching in *G*, then there is an *f* in *G'* with value |*f*| = |*M*|
- Conversely, if *f* is a flow in *G'*, then there is a matching with |*M*| = |*f*|
  - Proof 1: *M* corresponds to *f*
    - if *f(u,v)*∈ *M*, then *f(s,u)* = *f(u,v)* = *f(v,t)* = 1
      - other edges, *f(u,v)* = 0
    - *(u,v)*∈ *M* corresponds to one unit of flow in *G'* *s*→*u*→*v*→*t*
    - net flow across cut ($L \cup \{S\}$, $R \cup \{T\}$) = |*M*|
    - by Lemma 26.4, the value of the flow is |*f*|=|*M*|
  - Proof 2: *f* corresponds to *M*
    - Let $M = \{(u,v) : u \in L, v \in R, \text{and } f(u,v) > 0\}$
    - for each *u*∈*L,* one unit flow enters *u* if and only if one vertex *v*∈*R* such that *f(u,v)* = 1
    - for every matched vertex *u*∈*L*, we have *f(u,v)* = 1
    - net flow across cut ($L \cup \{S\}$, $R \cup \{T\}$) = |*M*|
    - by Lemma 26.4, the value of the flow is |*M*|=|*f*|

- (Theorem 26.10) If the capacity function *c* takes on only integral values, then maximum flow f produced by FF method |*f*| is integer. Moreover, for all vertices *u* and *v*, *f(u,v)* is integer
  - exercise: 26.3-2
- (corollary 26.11) The cardinality of maximum matching *M* in a bipartite graph *G* equals the value of a maximum flow *f* in its corresponding flow network *G'*

Conclusion

- How to solve MBM?
  - create corresponding flow network *G'*
  - run FF method, O(|*f**| *E'*) time
  - obtain the MBM
- MBM Time complexity
  - $|f^*| = O(V)$
  - $|E'| = \Theta(E)$
  - totally, $O(VE)$

Reading

- CH 26.1-26.3