19 June

The max-flow, min-cut theorem [P&S §6.1]

Recall the max-flow LP:

			Dual variables
PRIMAL:	$\max\ v$	[value of flow]	
	s.t. $Af + d^T v = 0$	[flow balance]	π_x
	$f \leq b$	[capacities]	γ_{xy}
	$f \geq 0,\ v \geq 0.$		

Yesterday we viewed this LP as the dual (within the primal-dual framework). Today we view it as the primal and investigate its dual.

- Also yesterday we had v unrestricted, while today we have $v \geq 0$. This is OK, because the objective is to maximize v and the all-zero flow is always feasible (as long as $b \geq 0$), so the optimal value of v is nonnegative.

The primal has one flow-balance constraint for each node, so correspondingly the dual will have a variable π_x for each $x \in V$.

The primal has one capacity constraint for each arc, so correspondingly the dual will have a variable γ_{xy} for each $(x,y) \in E$.

The dual of the max-flow LP is

DUAL: $\min \sum_{(x,y)\in E} b_{xy}\gamma_{xy}$

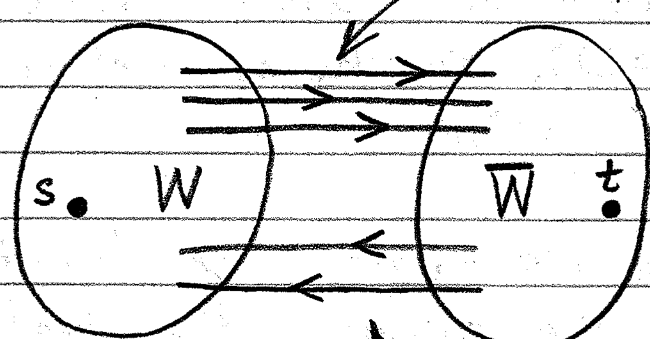
$$\begin{aligned} \text{s.t.}\quad & \pi_x - \pi_y + \gamma_{xy} \geq 0 \quad \text{for all } (x,y)\in E \\ & -\pi_s + \pi_t \geq 1 \\ & \pi \text{ unrestricted}, \quad \gamma \geq 0. \end{aligned}$$

Defn. Given a flow network $N=(s,t,V,E,b)$, an <u>$s-t$ cut</u> is a partition $(W,\overline{W})$ of the node set V into subsets W and $\overline{W}$ such that $s\in W$ and $t\in\overline{W}$. The <u>capacity</u> of the cut is

$$C(W,\overline{W}) = \sum_{\substack{(i,j)\in E \\ \text{such that} \\ i\in W,\, j\in\overline{W}}} b_{ij}.$$

Pictorially:

Arcs that go <u>forward</u> across the cut. The capacity of the cut is the sum of the capacities of these arcs.



Arcs that go <u>backward</u> across the cut. The capacity of the cut does not count these arcs.

19 June

Max-flow, min-cut — 2

<u>Theorem</u> [P&S Thm. 6.1] Every s–t cut $(W, \overline{W})$ determines a feasible solution to the dual of the max-flow LP, having cost $C(W, \overline{W})$, as follows:

$$\pi_x = \begin{cases} 0, & \text{if } x \in W; \\ 1, & \text{if } x \in \overline{W}. \end{cases}$$

$$\gamma_{xy} = \begin{cases} 1, & \text{for arcs } (x,y) \text{ such that } x \in W, y \in \overline{W}; \\ 0, & \text{otherwise.} \end{cases}$$

<u>Proof.</u> First we verify feasibility. Clearly these values of π_x and γ_{xy} satisfy the domains. To verify that the constraints are satisfied:

<u>Case 1</u>: $x \in W, y \in W$. Then $\pi_x - \pi_y + \gamma_{xy} = 0 - 0 + 0 \geq 0$. ✓
<u>Case 2</u>: $x \in W, y \in \overline{W}$. Then $\pi_x - \pi_y + \gamma_{xy} = 0 - 1 + 1 \geq 0$. ✓
<u>Case 3</u>: $x \in \overline{W}, y \in W$. Then $\pi_x - \pi_y + \gamma_{xy} = 1 - 0 + 0 \geq 0$. ✓
<u>Case 4</u>: $x \in \overline{W}, y \in \overline{W}$. Then $\pi_x - \pi_y + \gamma_{xy} = 1 - 1 + 0 \geq 0$. ✓

For the last constraint: Since $s \in W$ and $t \in \overline{W}$, we have $-\pi_s + \pi_t = -0 + 1 \geq 1$. ✓
So the stated solution is feasible. Its objective value is

$$\sum_{(x,y)\in E} b_{xy}\gamma_{xy} \underbrace{=}_{\text{by construction of } \gamma_{xy}} \sum_{\substack{(x,y)\in E \\ \text{such that} \\ x \in W,\, y \in \overline{W}}} b_{xy} = C(W, \overline{W}).$$

□

Max-flow, min-cut — 3

19 June

<u>Theorem</u> (Max-flow, min-cut — P&S Thm 6.2)
The value of any s-t flow is no greater than the capacity $C(W, \overline{W})$ of any s-t cut. Furthermore, the value of the maximum flow equals the capacity of the minimum cut, and a flow f and cut $(W, \overline{W})$ are jointly optimal if and only if

backward edges across cut → $f_{xy} = 0$ for all $(x,y) \in E$ such that $x \in \overline{W}$ and $y \in W$;

forward edges across cut → $f_{xy} = b_{xy}$ for all $(x,y) \in E$ such that $x \in W$ and $y \in \overline{W}$.

<u>Proof.</u> Almost directly from LP duality.

Previous theorem (with weak duality thm) proves first sentence.

Optimality conditions are from complementary slackness.

The only catch is whether an optimal solution to the dual of max-flow LP actually corresponds to a cut (because a cut must be an integral solution). The Ford-Fulkerson algorithm (next) shows that a given maximal flow of value v can always be used to construct a cut with value $C = v$. $\square$

Ford-Fulkerson algorithm [P&S §6.2]

Defn. Given a flow network $N=(s,t,V,E,b)$ and a feasible s-t flow f, an augmenting path is a path P from s to t in the undirected graph resulting from G by ignoring arc directions, with the following properties:

(a) For every arc $(i,j) \in E$ that is traversed by P in the forward direction (called a forward arc), we have $f_{ij} < b_{ij}$.

(b) For every arc $(j,i) \in E$ that is traversed by P in the reverse direction (called a backward arc), we have $f_{ji} > 0$.

In other words:

- Forward arcs of P are unsaturated.
- Backward arcs of P have nonzero flow.

Observe: If we have an augmenting path P for a flow f, we can increase the value of the flow* by increasing the flow on every forward arc and decreasing it on every backward arc.

* while preserving flow conservation

19 June

Ford–Fulkerson — 2

Maximum amount of flow augmentation possible along P:

$$\delta = \min_{\text{arcs of } P} \left\{ \begin{array}{ll} b_{ij} - f_{ij} & \text{along forward arc} \\ f_{ji} & \text{along backward arc} \end{array} \right\}$$

<u>Q</u>: How do we systematically find an augmenting path?

<u>A</u>: We can <u>label</u> the nodes.

The label for a node x will have two parts: (<u>from</u>, <u>how-much</u>)

- <u>from</u> indicates from where x was labeled.
- <u>how-much</u> indicates the amount of extra flow that can be brought to x from s.

Labeling nodes outward from x is called <u>scanning</u> x.

We maintain a LIST of labeled but unscanned nodes.

How to scan a node x: Identify all *unlabeled* nodes y such that either (x,y) or (y,x) is an arc.

Case 1: (x,y) is an arc. $x \longrightarrow y$

If $f_{xy} < b_{xy}$ (arc is *unsaturated*), then label y with (*from*[y], *how-much*[y]), where

- *from*[y] = x
- *how-much*[y] = $\min\{$*how-much*[x], $b_{xy} - f_{xy}\}$

Case 2: (y,x) is an arc. $x \longleftarrow y$

If $f_{yx} > 0$, then label y with (*from*[y], *how-much*[y]), where

- *from*[y] = $-x$ [minus sign indicates backward arc]
- *how-much*[y] = $\min\{$*how-much*[x], $f_{yx}\}$

19 June

Outline of Ford-Fulkerson algorithm [P&S Fig. 6-6]

1. Initialize: $f := 0$.

2. Erase all node labels. Set LIST $:= \{s\}$ and how-much[s] $:= \infty$.

3. Choose any node x from LIST and remove it from LIST.

4. Scan x:
   - For every unlabeled node y such that $(x,y) \in E$ and $f_{xy} < b_{xy}$, set from[y] $:= x$ and how-much[y] $:= \min\{$how-much[x], $b_{xy} - f_{xy}\}$ and add y to LIST.
   - For every unlabeled node y such that $(y,x) \in E$ and $f_{yx} > 0$, set from[y] $:= -x$ and how-much[y] $:= \min\{$how-much[x], $f_{xy}\}$ and add y to LIST.

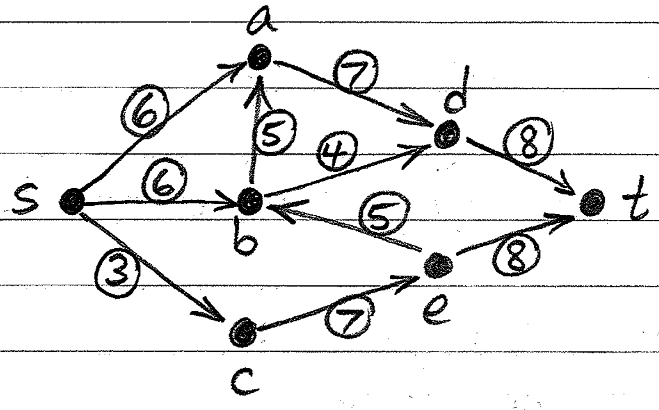
5. If t is labeled, then augment flow f along the augmenting path by how-much[t] (using from values backward from t to identify the augmenting path) and go back to step 2.

6. If LIST is nonempty, go back to step 3.

7. Done. Flow f is optimal.

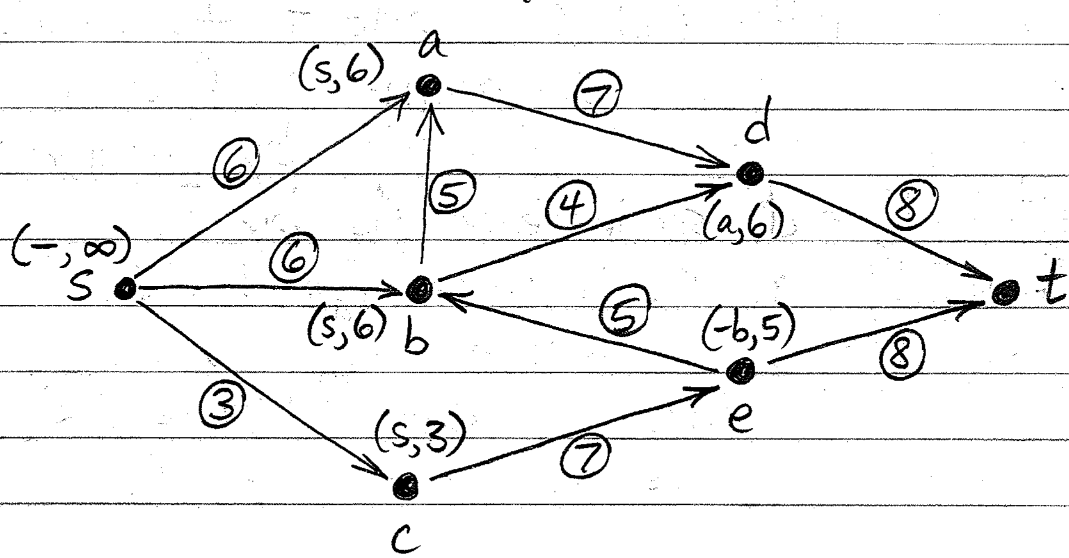
19 June

Ford-Fulkerson: Example.



Arc capacities are circled.

Initialize: All flows f_{ij} along arcs are set to zero, how-much[s] is set to ∞, LIST is set to $\{s\}$.



[Node labels computed below.]

LIST = $\{s\}$.

Choose s, remove it from LIST.

Scan s:

Label a with (s, 6), add a to LIST.

Label b with (s, 6), add b to LIST.

Label c with (s, 3), add c to LIST.

LIST = $\{a,b,c\}$.

→

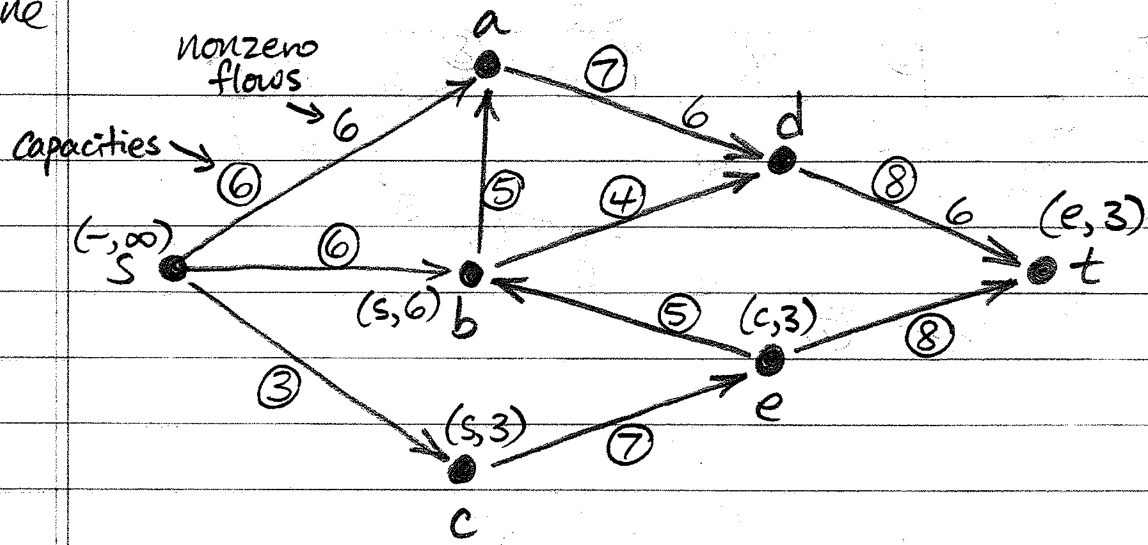
Choose a, remove it from LIST.
Scan a:
Label d with (a, 6), add d to LIST.
LIST = {b, c, d}.
Choose b, remove it from LIST.
Scan b:
Label e with (−b, 5), add e to LIST.
LIST = {c, d, e}.
Choose c, remove it from LIST.
Scan c: [nothing to be done]
LIST = {d, e}.
Choose d, remove it from LIST.
Scan d:
Label t with (d, 6), add t to LIST.
Now t is labeled, so augment flow f along the augmenting path by <u>how-much</u>[t] = 6:

- <u>from</u>[t] = d, so increase f_{dt} by 6.
- <u>from</u>[d] = a, so increase f_{ad} by 6.
- <u>from</u>[a] = s, so increase f_{sd} by 6.

Erase all node labels, set LIST := {s}, set <u>how-much</u>[s] := ∞, and repeat.

Ford-Fulkerson example — 2

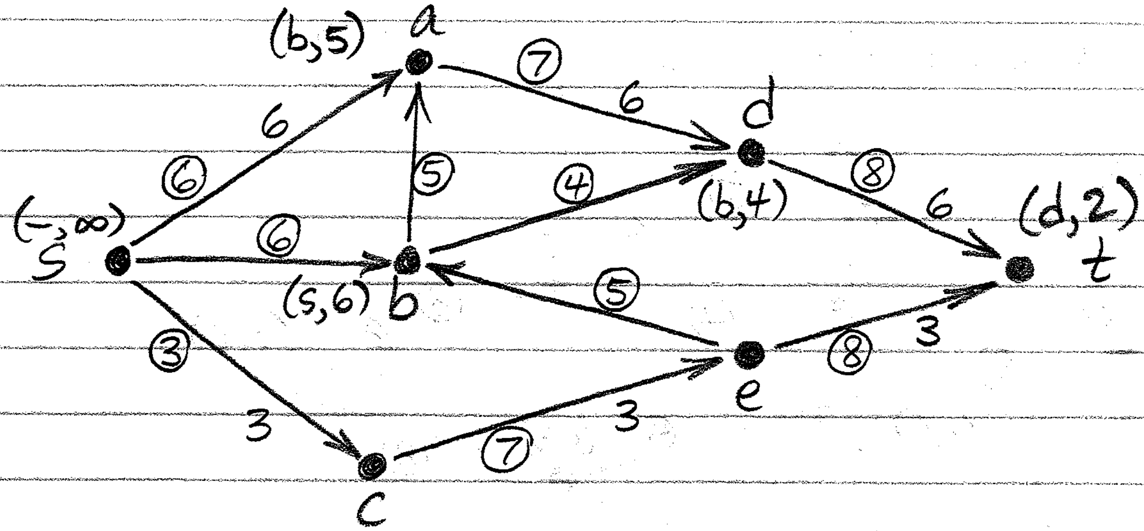
19 June



LIST = {s}.
Choose s, remove it from LIST.
Scan s:
  Label b with (s, 6), add b to LIST.
  Label c with (s, 3), add c to LIST
LIST = {b, c}.
Choose c (why not?), remove it from LIST.
Scan c:
  Label e with (c, 3), add e to LIST.
LIST = {b, e}.
Choose e, remove it from LIST.
Scan e:
  Label t with (e, 3), add t to LIST.
Now t is labeled, so augment flow f along the augmenting path by <u>how-much</u> [t] = 3:

- <u>from</u> [t] = e, so increase f_{et} by 3.
- <u>from</u> [e] = c, so increase f_{ce} by 3.
- <u>from</u> [c] = s, so increase f_{sc} by 3.

Erase all node labels, set LIST := {s}, set <u>how-much</u> [s] := ∞, and repeat.



LIST = {s}.

Choose s, remove it from LIST.

Scan s:

Label b with (s, 6), add b to LIST.

LIST = {b}.

Choose b, remove it from LIST.

Scan b:

Label a with (b, 5), add a to LIST.

Label d with (b, 4), add d to LIST.

LIST = {a, d}.

Choose a, remove it from LIST.

Scan a: [nothing to be done]

LIST = {d}.

Choose d, remove it from LIST.

Scan d:

Label t with (d, 2), add t to LIST.

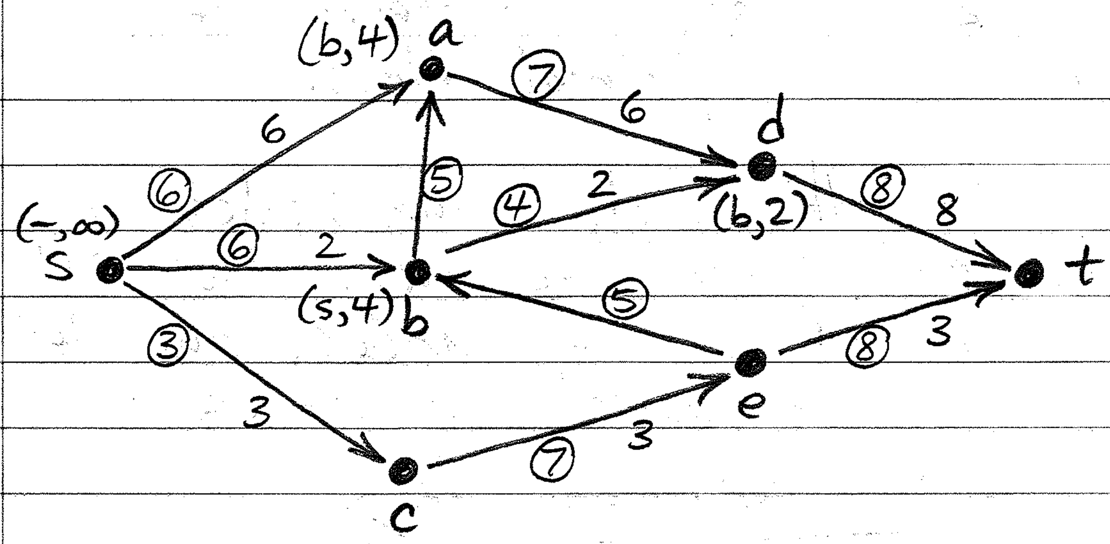
Now t is labeled, <u>how-much</u>[t] = 2:

- <u>from</u>[t] = d, so increase f_{dt} by 2.
- <u>from</u>[d] = b, so increase f_{bd} by 2.
- <u>from</u>[b] = s, so increase f_{sb} by 2.

Erase all node labels, set LIST := {s}, set <u>how-much</u>[s] := ∞, repeat.

Ford-Fulkerson example – 3

19 June

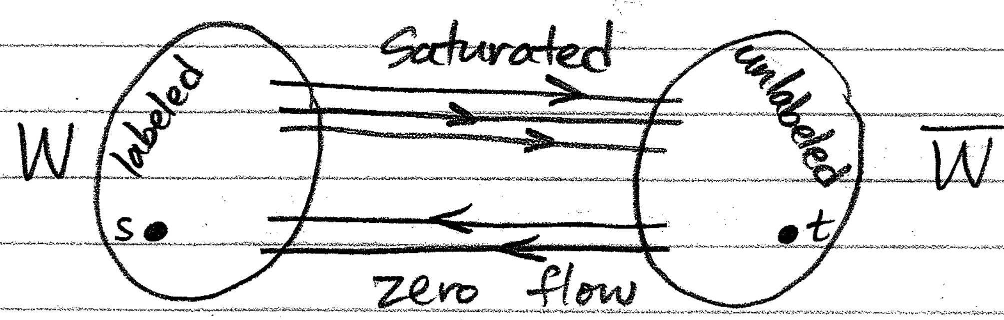


LIST = {s}.
Choose s, remove it from LIST.
Scan s:
  Label b with (s,4), add b to LIST.
LIST = {b}.
Choose b, remove it from LIST.
Scan b:
  Label a with (b,4), add a to LIST.
  Label d with (b,2), add d to LIST.
LIST = {a, d}.
Choose a, remove it from LIST.
Scan a: [nothing to be done]
LIST = {d}.
Choose d, remove it from LIST.
Scan d: [nothing to be done]
LIST = ∅.

Done. Current flow is optimal.

Theorem [P&S Thm 6.3] When the Ford–Fulkerson algorithm terminates, it does so at optimal flow.

Proof. At the end, some nodes are labeled and some are not. Let W be the set of labeled nodes, and let $\overline{W}$ be the set of unlabeled nodes.



All arcs (x,y) with $x \in W$, $y \in \overline{W}$ must be saturated; otherwise y would have been labeled when x was scanned. Likewise, all arcs (y,x) with $x \in W$, $y \in \overline{W}$ must have zero flow. So $(W, \overline{W})$ is a cut whose value equals the value of the flow, so the flow and the cut must be jointly optimal by max-flow min-cut theorem.

$\square$