CS 340 - lecture 26 - Nov 06

final remarks on Dijkstra's algorithm:

(a) for sparse graphs, there are some variants of Dijkstra's algorithm that solve the weighted SP problem more efficiently;

(b) there are variants of Dijkstra's algorithm that (unlike the one discussed in class) can handle negative edge costs (these have a high running time cost though!); in practice, usually non-negative edge costs are sufficient;

(c) If the input graph is known to be acyclic, a more efficient version of Dijkstra's algorithm exists.

4.4. Network Flow Problems

given
- weighted directed graph $G = (V, E)$ with weight function $c: E \rightarrow \mathbb{R}^{\geq 0}$
- source vertex $s \in V$, sink vertex $t \in V$

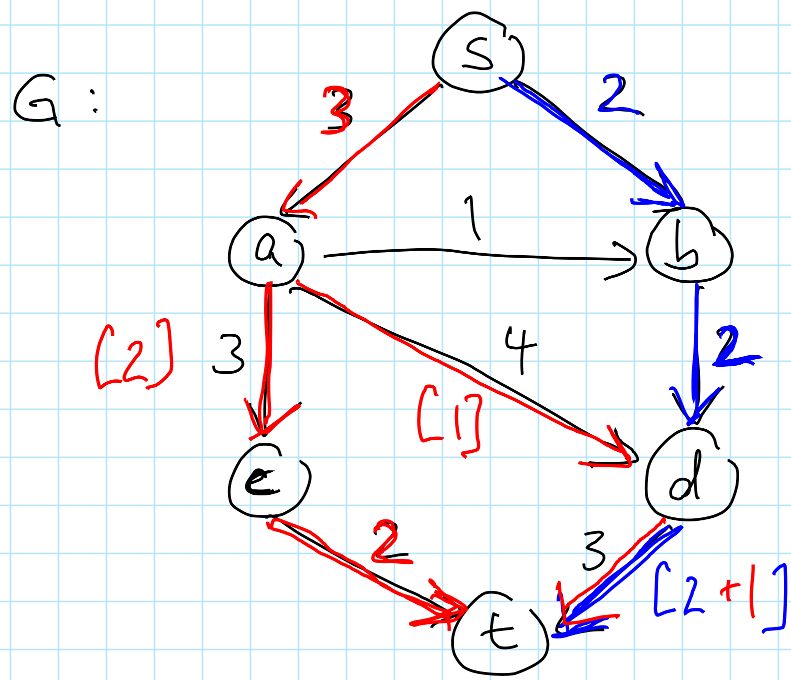
task: determine the max. amount of "flow" that can pass from s to t, where
- for $(v,w) \in E$, $c(v,w)$

of flow that can pass along (v,w)

"capacity of (v,w)" $c \hat{=}$ capacity

- for $v \in V \setminus \{s,t\}$, the total flow coming into v must equal the total flow going out of v

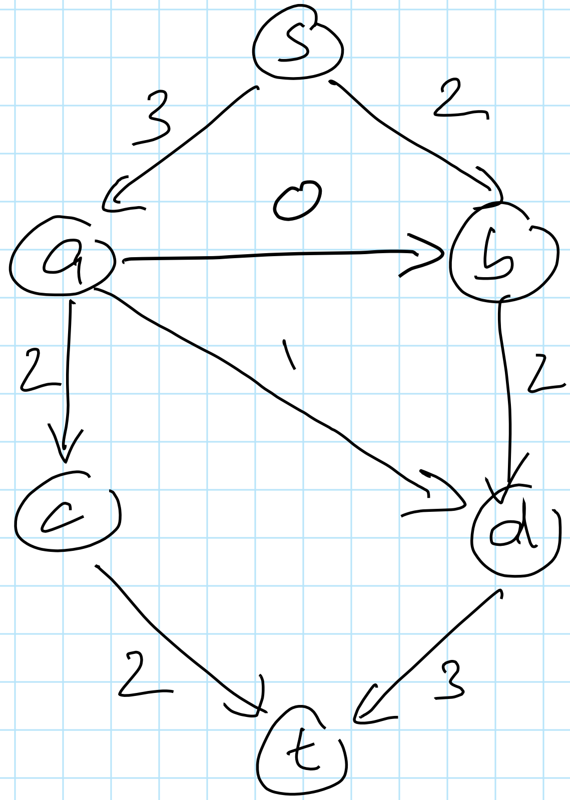
Example 42.



maximum flow is 5

- at most 5, because the sum of the edge capacities of outgoing edges from s is 5.
- at least 5, because the following "flow graph" witnesses a flow amount of 5:

G_f



flow graph

<u>first algorithmic idea</u>:

- construct a flow graph G_f in stages
  G_f contains same vertices and edges as G.
  Initially all edge weights are 0.
- maintain a residual graph G_r (representing flow not used yet).
  Initially, $G_r = G$. At every stage, edge weights indicate residual (= not yet used) flow capacity.
  If an edge weight becomes 0, remove edge.
- at

find a path from s to t in G_r. The min. edge cost along that path is added to every edge on the same path in G_f

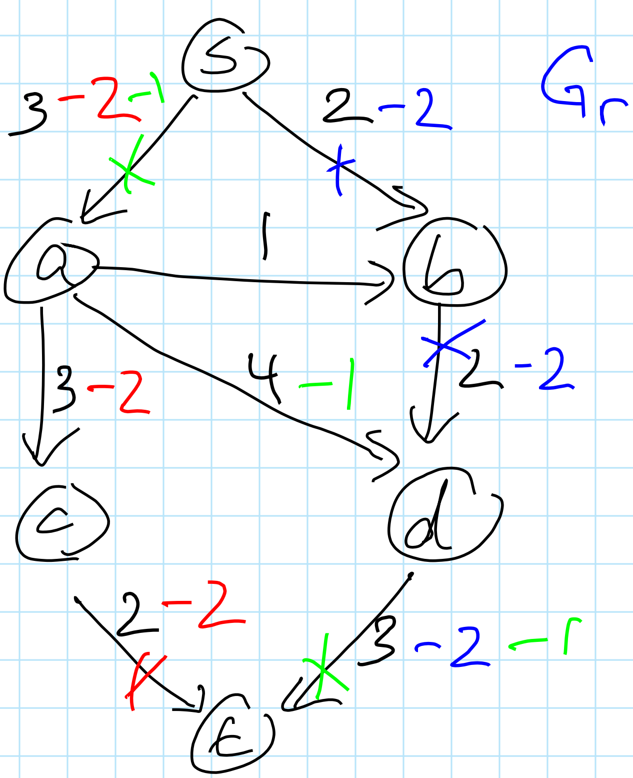
Update G_r.

If no path from s to t in G_r exists, stop.

How to pick a path when more than one available?

Does it matter? YES!

<u>Example 42 ctd.</u>



four paths from s to t

[s, a, c, t]

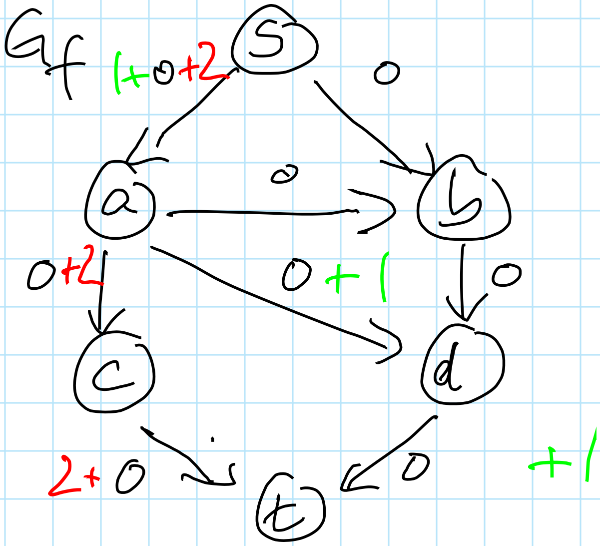
[s, b, d, t]

[s, a, b, d, t]

[s, a, d, t]

If we pick first [s, b, d, t]

G_f



[s, a, d, t]

[s, a, c, t]

Suppose next we choose [s, a, c, t]

Next we choose [s, a, d, t]

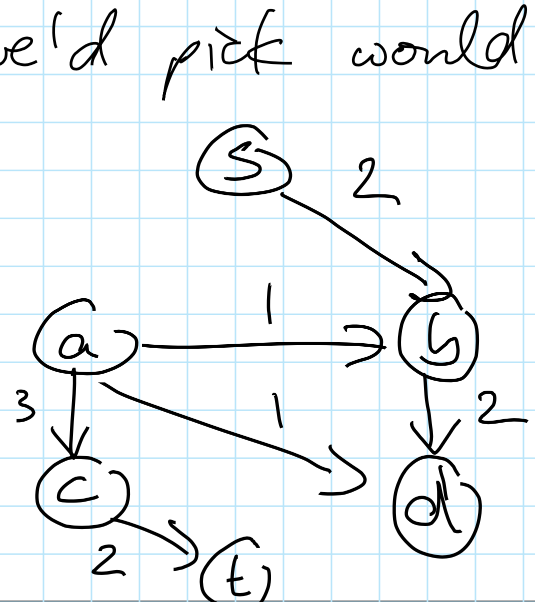
$\Rightarrow$ flow of 5

HOWEVER, what if we had chosen differently at the start?

Suppose we pick <u>greedily</u> path with highest flow initially?

Then the <u>first</u> path we'd pick would be [s, a, d, t].

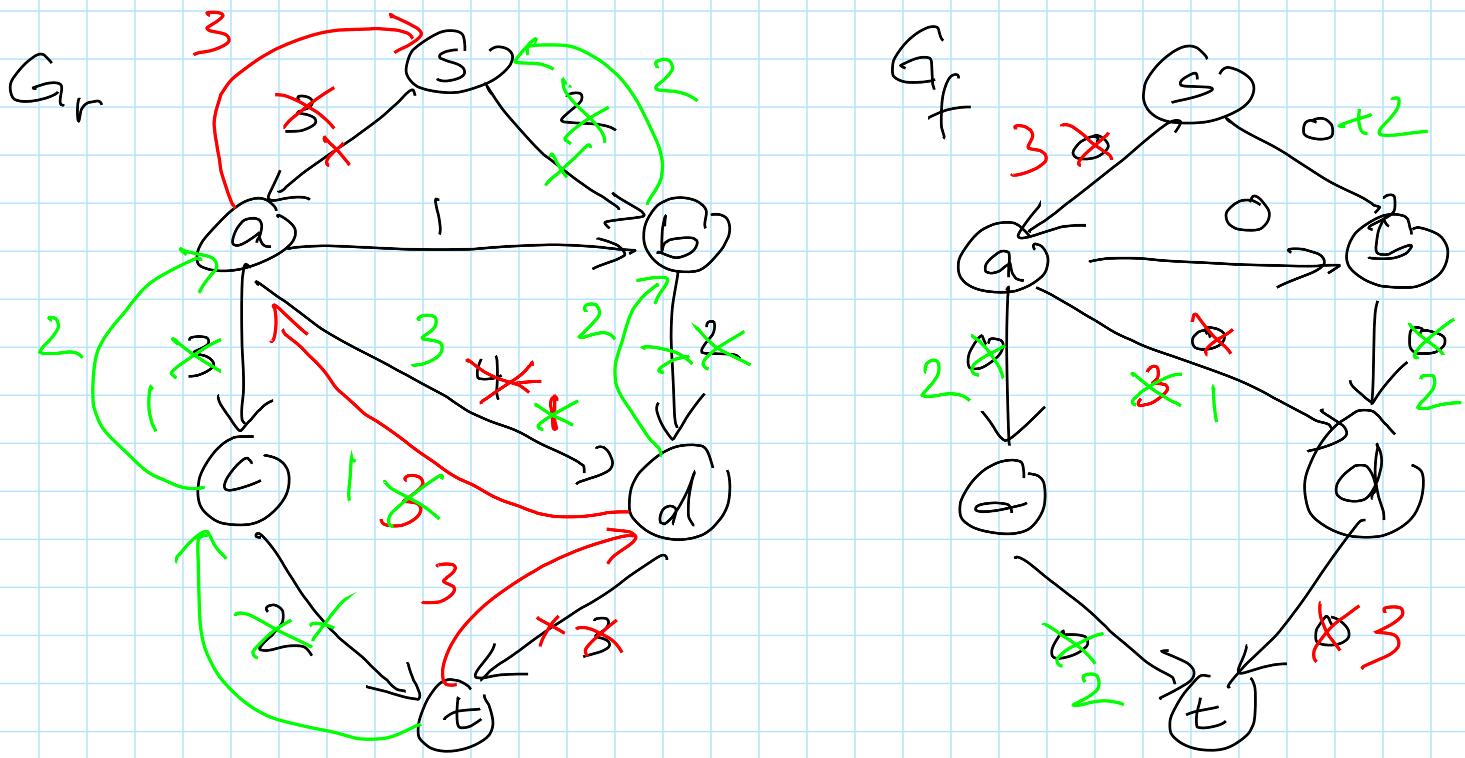
(for a flow of 3) $\leadsto$ G_r



$\rightsquigarrow$ we only get flow of 3 in total
$\longrightarrow$ suboptimal

<u>idea</u>: use greedy alg., but add reverse edge in G_r so that flow can be reversed when alg. "changes its mind"

<u>Example 42 ctd.</u>



<u>greedy choice</u>: $[\underline{s, a, d, t}]$ 3

<u>next greedy choice</u>: $[\underbrace{s, b, d, a, c, t}_{2}]$

then no more path left from s to t

$\rightarrow$ done ✓

proof of correctness omitted.

$$T_{worst} = O\left(|E|^2 \cdot \log \max_{(v,w)\in E} c(v,w) \cdot \log|V|\right)$$

(no proof given)