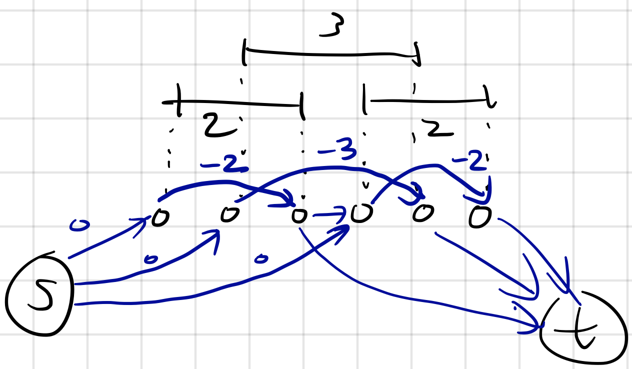
4 Mar 2019 — Network Flow (Chapter 7)

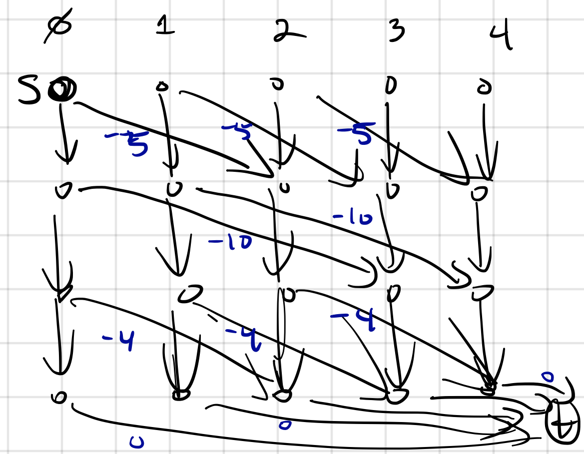
Digression: reductions!

Paths in directed graphs are a surprisingly expressive "gadget" for describing other computational problems.

Ex. Weighted interval scheduling



Knapsack: $v_1 = 5$ $w_1 = 2$
$v_2 = 10$ $w_2 = 3$
$v_3 = 4$ $w_3 = 2$
$W = 4$



Cost 0 on vertical edges
$-v_i$ on edges from row $i-1$ to row i

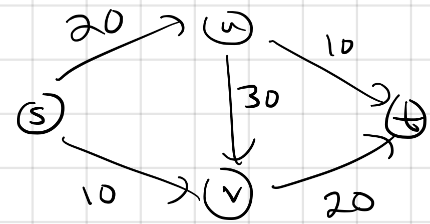
Network flow problems involve packing paths into graphs with edge capacities.

"Pick as many paths as possible from s to t, given that paths may not 'conflict' with each other by using more than the allowed capacity on any edge."

Def. A <u>flow</u> <u>network</u> is specified by

- a directed graph G.
- a capacity $c(e) \geq 0$ for each edge e of G
- a source node, s, and sink node, t.

<u>E.g.</u>



Conventions:

[1] s has no incoming edges, t has no outgoing edges.
[2] every node has at least one edge coming in or going out $\therefore\ n = O(m)$.
\# verts $\leq 2 \cdot$ (\# edges)

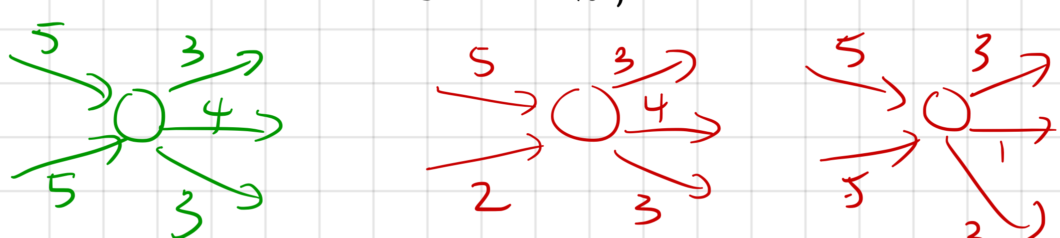
[3] All edge capacities are integers.

A <u>flow</u> in (G, c, s, t) is a function assigning a number $f(e)$ to each edge e, s.t.

- Capacity constraints $\quad 0 \leq f(e) \leq c(e) \quad \forall e$
- Conservation constraints $\quad f^{out}(v) = f^{in}(v) \quad \forall v \neq s, t$.

$$f^{out}(v) \triangleq \sum_{e:\, v = tail(e)} f(e)$$

$$f^{in}(v) \triangleq \sum_{e:\, v = head(e)} f(e)$$



YES NO No

Max Flow problem: find a flow f that maximizes $f^{out}(s)$.