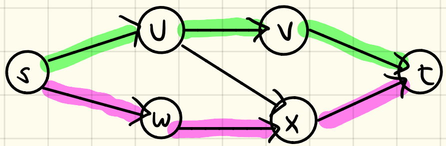
Using flow to count disjoint paths.

Given: A directed graph $G = (V, E)$ and two nodes $s, t \in V$.

Find: The number of edge-disjoint paths from s to t.

Note: Given a collection of paths $P = \{p_1, p_2, \dots, p_k\}$ we say that the paths are edge disjoint if $\forall i \neq j$ p_i and p_j share no edge in common.



here, $(s,u), (u,v), (v,t) = p_1$ and
$(s,w), (w,x), (x,t) = p_2$
are edge-disjoint paths. Any other s-t path in G would share edges with one or both of these.

Like bipartite matching, we will solve this problem by reducing it to an instance of a flow problem.

The transformation:

Given our original directed graph G, we will create a flow network G' in the following way.

Let G' have the same vertex set and edge set as G. Further, for all $e \in E$, let $c_e = 1$.

Now, we make the following claim:

(7.41) If there are k edge-disjoint paths in G from s to t, then the value of the max flow in G' is at least k.

Proof: If there exist k edge disjoint paths in G, then there also exist k edge disjoint paths in G', since the topology is identical. Further, since all capacities in G' are 1, each such path can carry exactly 1 unit of flow. Hence, each of the k paths can carry 1 unit of flow for a total flow of value k. Hence, G' supports a flow of value at least k.

What about the converse?

Claim: If there is a flow of value k in G', then G contains k edge-disjoint paths from s to t.

We show this by the following

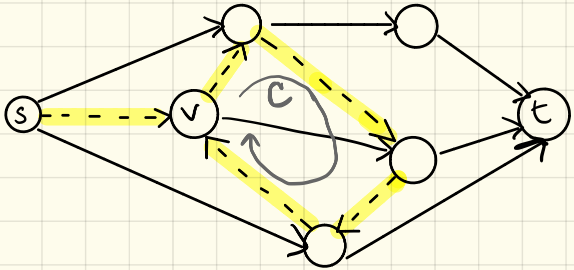
(7.42) If f is a 0-1 flow of value v, then the set of edges in G' with $f(e) = 1$ contains a set of v, edge-disjoint paths.

Proof: Induction on # of edges carrying flow

The case of $v=0$ is trivial. Otherwise, if $v>0$, there must be some edge (s,u) that carries flow from s. However, by conservation, that flow must leave u via some edge (say (u,v)). Likewise, that flow must leave v via some edge (v,w) etc. Continuing this process, there are only 2 possibilities. Either, (a) we eventually reach t or (b) we encounter some node (say v) a second time.

(a) In this case, we've found an s-t path, and it carries exactly 1 unit of flow from s-t. Let f' be the flow we get by decreasing the flow along each edge of this path by 1 unit. This new flow, f', has value $v-1$, and we can apply the same procedure on this flow to extract $v-1$ other (edge-disjoint) paths.

(b) If our path P reaches some node v for a second time, then we have a cycle C and the situation looks like the following:



Consider the cycle C of edges that we traverse between the first and second times we visit vertex v. Consider obtaining a new flow f' from f by decreasing the flow along all edges of C to 0. The new flow f' still has value v, but it has fewer edges carrying flow. Thus we can still apply the induction to f' to recover the remaining v disjoint paths.

So, in both situations (a) and (b) we make progress, and it is always true that if we have a flow of value v, we have v edge-disjoint paths carrying the flow.

Together, 7.41 and 7.42 give us G' has a flow of value k if and only if G has k disjoint s-t paths.

Moreover, because we are dealing with a 0-1 flow, we can make a strong statement about the runtime of an algorithm to solve this problem.

Assume we use Ford-Fulkerson, which has a worst-case bound of $O(mC)$ where $C = \sum_{e \text{ out of } s} c_e$. However, in G', each c_e is 1, and there can be at most $|V-1|$ edges leaving s. Thus, FF will always run on such instances in at most $O(mn)$ time.

Note: The approach we found here was <u>constructive</u>. That is, not only can we count the # of edge-disjoint paths efficiently, we can also extract the actual set of paths in $O(nm)$ time.

Important extensions to consider:

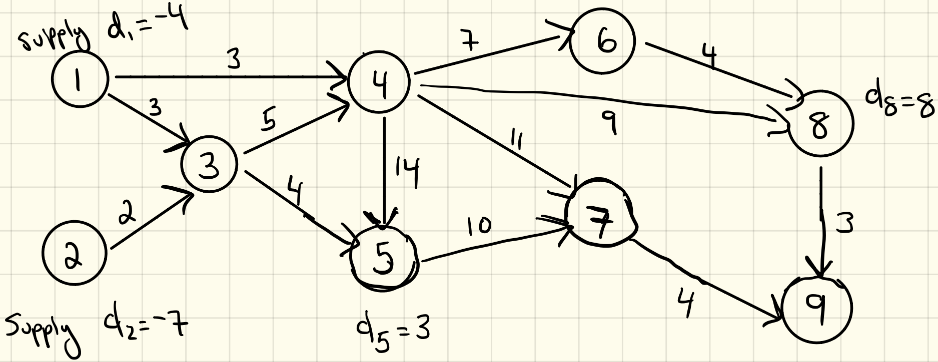
- What if G was undirected? (pg 377-378 of K+T)
- What if we wanted node-disjoint paths... how to reduce node disjoint to edge disjoint?

Extensions to flow problems:

Circulation with Demands:

- Suppose there are multiple sources and multiple sinks.
- Each sink wants a certain amount of flow (called the demand of the sink)
- Each source produces a certain amount of flow (called the supply)
- We can represent supply as negative demand.

E.g.



In this problem, constraints change somewhat

Goal: Find a flow that satisfies

1) Capacity constraints: For each $e \in E$, $0 \leq f(e) \leq c_e$

2) Demand constraints: For each $v \in V$, $f^{in}(v) - f^{out}(v) = d_v$

- The demand is the excess flow that should come into the node.

Let S be the set of sources with negative demands (supply)
Let T be the set of sinks with positive demands

In order for there to be a <u>feasible</u> flow, we must have

$$\sum_{s \in S} -d_s = \sum_{t \in T} d_t$$

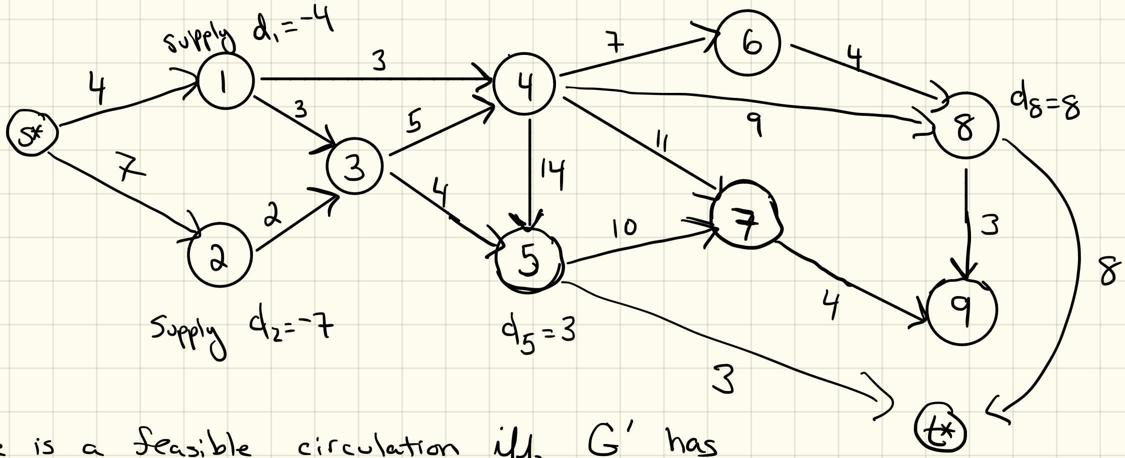
Let $D = \sum_{t \in T} d_t$ be the total demand.

So, there appear to be some substantial differences between circulation with demands and Max flow. However, they are equivalent!

Reduction: $G \to G'$

1) Add a new source node s^* and an edge (s^*, s) for all $s \in S$.
2) Add a new sink node t^* and an edge (t, t^*) for all $t \in T$.

The capacity of $(s^*, s) = -d_s$ (since $d_s < 0$, this is positive)
The capacity of $(t, t^*) = d_t$



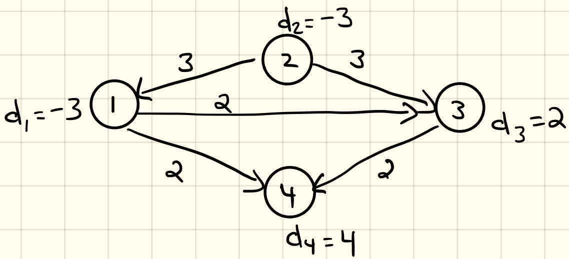
There is a feasible circulation iff G' has a flow f^* with $v(f^*) = D$.

- Capacity of (s^*, s) edges limits the supply of source nodes
- Capacity of (t, t^*) edges allow d_t flow to reach t^* from each t.

• We can use "normal" max flow to find these circulations.

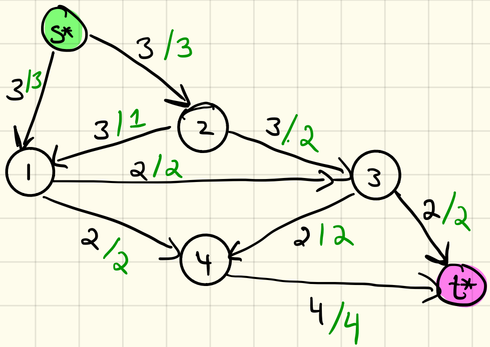
Here is an example that does work

G :



reduction to max flow

G' :



Consider a related problem:

X - What if there are multiple "commodities" i.e. each sink t_i only accepts (demands) flow from source s_i. It turns out this modification makes the problem (with integer flows) NP-complete.

✓ - What if we also want a lower bound on the amount of flow going through some edges?

This is a way to require that certain edges are used at some capacity.

Goal: Find a flow f that satisfies

1) Capacity constraints: for each $e \in E$: $l_e \leq f(e) \leq c_e$
2) Demand constraints: for each $v \in V$: $f^{in}(v) - f^{out}(v) = d_v$

Consider an initial flow that sets the flow along each edge equal to the lower bound i.e.: $f_0(e) = l_e$

- f_0 satisfies the capacity constraints, but not necessarily the demand constraints

Def $L_v = f_0^{in}(v) - f_0^{out}(v)$

L_v is the amount of demand satisfied at each v by f_0.

Consider the demand constraints:

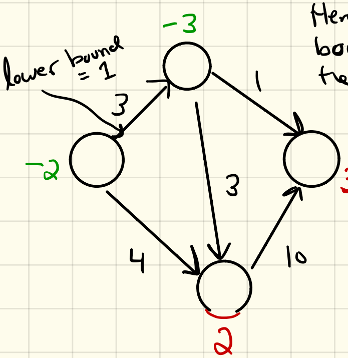
$$f^{in}(v) - f^{out}(v) = d_v - L_v$$

and capacity constraints

$$0 \le f(e) \le c_e - \ell_v$$

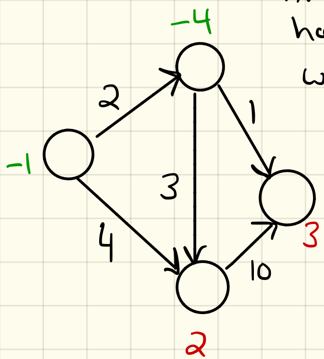
These constraints yield a <u>standard instance</u> of the circulation with demands problem.

E.g.



Here, the lower bound prevents the obvious circulation

After our transformation, we have an equivalent instance with no lower bound.



Reduction : Given an instance G of circulation with demands and lower bounds

1) Subtract l_e from the capacity of each edge e
2) Subtract L_v from the demand of each node v

note: this may create new sources or sinks

Then: Solve the "standard" circulation with demands on this new instance G' to get a flow f'!

To find a flow satisfying the original constraints, we add l_e to every $f'(e)$.

This works because reductions can be "chained"

Circulation with demands + lower bounds

Circulation with demands

Max flow

$\rightarrow$ = can be reduced to.

$\rightarrow$ = Solution can be transformed to.