network flow
problem: send flow from s to t

flow into each node must equal the flow out, except for s and t
problem formulation

• network is a directed graph G=(V,E)
• each edge e in E has a capacity c(e)
• goal is to assign a flow function f
• for each edge e, 0≤f(e)≤c(e)
• for each node v (not s or t), the flow into v equals flow out of v
• total flow out of s equals flow into t
• MAXIMIZE total flow from s to t
example flow assignment

value = 8 + 10 + 10 = 28

draw edge labels are flow/capacity

maximum flow problem:

A function \( \phi \) that satisfies:

- for every edge \( e \in E \):
  \[ \phi(e) \leq c(e) \]

- for every node \( v \in V \setminus \{s, t\} \):
  \[ \sum_{e \in \text{in}(v)} \phi(e) = \sum_{e \in \text{out}(v)} \phi(e) \]

The value of \( \phi \) is:

\[ \phi(v) = \sum_{e \in \text{in}(v)} \phi(e) - \sum_{e \in \text{out}(v)} \phi(e) \]
Ford-Fulkerson Method

input: graph G, source s, terminus t

1. initialize flow f of each edge to 0
2. while there exists an augmenting path p in the residual network $G_f$
3. augment flow along p
4. return f
residual graph $G_f$

original graph $G$

residual graph $G_f$

where $\text{flow on } G\text{ reverse edge negates flow on } G\text{ forward edge}$
how to update flow

• let p be a path in the residual graph $G_f$
• the bottleneck capacity $b$ is the smallest label on the path $p$
• to update the flow $f(e)$ of each edge $e$
  • ... if the path followed $e$ in the forward direction, set $f(e) = f(e)+b$
  • ... if $p$ followed $e$ as a back edge, set $f(e)=f(e)-b$
• (that’s what augmenting is, where flow might get diverted from the head of edge $e$)
example graph

network G

start with this – work done on board
max-flow min-cut theorem

The size of the maximum flow from s to t is the capacity of the minimum (s,t)-cut

*note*

- An (s,t)-cut is a partition of the nodes S and V-S with s in S and t in V-S
- the capacity of that cut is the sum of the flow leaving S
- algorithm correctness follows (with a lot of work)
time for algorithm

- flow increases by at least 1 with each iteration
- total time is $O(V \cdot E \cdot C)$ where $C$ is network capacity
- could be slow if $C$ is very large

- many improvements
- Edmonds-Karp is $O(V \cdot E^2)$
- idea is to use BFS to find $s$-$t$ path in $G_f$
bipartite matching

- bipartite graph has edges only between L side and R side
- problem is to choose the max number of edges that match elements

Matching: 1-2', 3-1', 4-5'
larger matching
rephrase as max flow problem

\[ G' = (L \cup R \cup \{s, t\}, E') \]

rephrase as max flow problem

Let \( G' \) be a bipartite graph with bipartition \( (L, R) \) and \( s \) and \( t \) are odd source and sink, respectively.

max s-t flow is max matching
airport problem

An airline company offers flights out of $n$ airports, conveniently labeled from 1 to $n$. The flight time $t_{ij}$ from airport $i$ to airport $j$ is known for every $i$ and $j$. It may be the case that $t_{ij} \neq t_{ji}$, due to things like wind or geography. Upon landing at a given airport, a plane must be inspected before it can be flown again. This inspection time $p_i$ is dependent only on the airport at which the inspection is taking place and not where the previous flight may have originated.

Given a set of $m$ flights that the airline company must provide, determine the minimum number of planes that the company needs to purchase. The airline may add unscheduled flights to move the airplanes around if that would reduce the total number of planes needed.

from 2015 Pac NW Regional ACM Programming Contest
airport problem (cont’d)

Input
The first line of input contains two space-separated integers $n$ and $m$ ($1 \leq n, m \leq 500$). The next line contains $n$ space-separated integers $p_1, \ldots, p_n$ ($0 \leq p_i \leq 10^6$).

Each of the next $n$ lines contains $n$ space-separated integers. The $j$th integer in line $i + 2$ is $t_{ij}$ ($0 \leq t_{ij} \leq 10^6$). It is guaranteed that $t_{ii} = 0$ for all $i$. However, it may be the case that $t_{ij} \neq t_{ji}$ when $i \neq j$.

Each of the next $m$ lines contains three space-separated integers, $s_i$, $f_i$, and $t_i$ ($1 \leq s_i, f_i \leq n$, $s_i \neq f_i$, $1 \leq t_i \leq 10^6$), indicating that the airline company must provide a flight that flies out from airport $s_i$ at exactly time $t_i$, heading directly to airport $f_i$.

Output
Print, on a single line, a single integer indicating the minimum number of planes the airline company must purchase in order to provide the $m$ requested flights.

rephrase as matching problem (how?)
circulation with demands

Each node $v$ has demand $d(v)$. If $d(v)<0$, then $v$ is a supply node. $d(v)>0$ means $v$ is a demand node. **question**: Is there a valid circulation (a flow), with supply and demand matching up within the edge capacities?

network $G$

**reduce to a regular flow problem (how?)**