Networks and flows

- Flows are directed graphs where each edge has a capacity $c(u \to v > 0)$, and each edge is labeled by $f(u \to v)$ such that

(1) $$0 \le f(u \to v) \le c(u \to v)$$ ← obvious by def. of capacity

(2) $$\sum_{u : u \to v} f(u \to v) = \sum_{w : v \to w} f(v \to w) \quad \text{for all } v \in V \setminus \{s, t\}$$

source and sink vertices

- (2) is flow conservation (like Kirchoff's current law)
- The value of a flow is the net outflow from s.
- A cut is a partition of V: $V = S \cup \bar{S}$, $s \in S$, $t \in \bar{S}$
  ↳ the capacity of a cut is defined by $\text{capacity}(S, \bar{S}) = \sum_{u \in S, v \in \bar{S} : u \to v} c(u \to v)$
- The max-flow min-cut theorem:

  maximum possible flow value = minimum cut capacity
- We can demonstrate $\text{value}(f) \le \text{capacity}(S, \bar{S})$ as follows

$$\text{value}(f) = \sum_v f(s \to v) - \sum_v f(v \to s) \quad \text{by def.}$$

$$= \sum_{v \in S} \left(\sum_u f(v \to u) - \sum_u f(u \to v) \right) \quad \text{using flow cons}$$

$$= \sum_{v \in S} \sum_{u \in S} f(v \to u) + \sum_{v \in S} \sum_{u \in \bar{S}} f(v \to u) - \sum_{v \in S} \sum_{u \in S} f(u \to v) - \sum_{v \in S} \sum_{u \in \bar{S}} f(u \to v)$$

telescopes to zero.

← split into $u \in S$ and $u \in \bar{S}$ since $S, \bar{S}$ partition $\{V\}$.

$$= \sum_{v \in S} \sum_{u \in \bar{S}} f(v \to u) - \sum_{v \in S} \sum_{u \in \bar{S}} f(u \to v)$$

since $0 \le f \le c$ ↓

$$\le \sum_{v \in S} \sum_{u \in \bar{S}} f(v \to u) \le \sum_{v \in S} \sum_{u \in \bar{S}} c(v \to u) = \text{capacity}(S, \bar{S})$$

- By solving the dual problem, and knowing that the primal and dual must meet, we can prove the theorem.

Ford-Fulkerson method

- To find the max-flow, we start with all flows 0.
- Then we iteratively find augmenting paths along which we can add flow. These augmenting paths will be on the residual graph.
  - ↳ residual graph also contains backward edges: to increase flow on one edge, we may need to decrease it on another.
  - ↳ for each augmenting path, we find the bottleneck then augment the original graph.
- FF doesn't specify exactly how the path should be chosen. ~~Once the~~
  - ↳ for each pair of vertices (v, w) in graph:

    if cap can be added

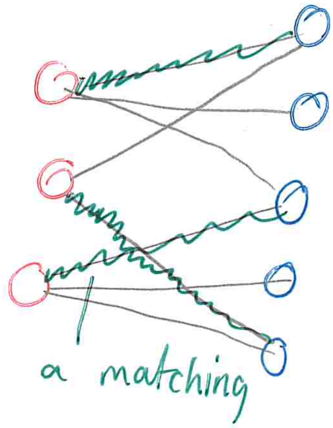
    if $f(v \to w) < c(v \to w)$: give h edge $v \to w$ labelled "forward"

    if $f(w \to v) > 0$: give h an edge $v \to w$ labelled "backwards"
  - ↳ thus the residual graph h is 'normal' and we can use BFS to find a path.
- FF terminates for ~~real~~ integer (and ∴ rational) capacities:
  - if an aug. path is found, its bottleneck $\delta > 0$
  - thus augmenting strictly increases flow
  - because flow is bounded (e.g by $\sum$ cap), it must terminate.
- The loop runs at most f^* times if f^* is the max flow (e.g $\delta = 1$ every time)
- Find aug path is $O(V+E)$ ∴ runtime $O(Ef^*)$
- Proving correctness:
  - once there are no more aug paths, with f^* the terminating flow, let S^* be the cut associated with f^*.
  - for all $v \in S^*$, $w \notin S^*$: $f^*(w \to v) = 0$ and $f^*(v \to w) = c(v \to w)$

    otherwise there is an aug path.
  - then by the max-flow min-cut theorem, f^* is a maximum flow.

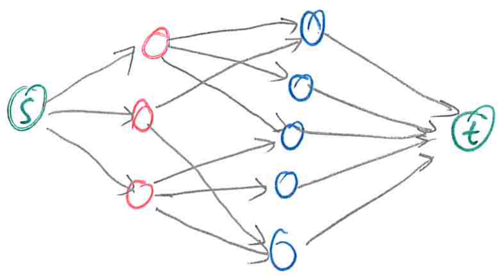
Matchings

- A bipartite graph has vertices split into two sets, with edges going from one set to the other.
- A matching on a bipartite graph is a selection of edges such that no vertex is connected to more than one edge
  - ↳ the size of a matching is the number of edges it contains.
  - ↳ the maximum matching has the largest size



- It can be translated into a flow problem

1. Add a source and sink
2. Make original edges into directed edges with capacity 1
3. Run Ford-Fulkerson



Proof of correctness:

- Because capacities are integer, algo terminates.
- The capacity constraint means that each vertex apart from s and t have at most one input & one output ⟹ valid matching.
- Because f^* is a max flow and $size(m) = value(f)$, m^* must be a maximum matching.