Today: Matching
- bipartite matching as flow
- augmenting paths
- Edmonds' algorithm & improvements
- weighted matching

Matching = set $M$ of edges sharing no endpoints
- cardinality $|M| = \# \text{edges in } M$
- goal: given undirected graph, find max. cardinality matching
- perfect if $|M| = |V|/2$

Bipartite matching: matching in bipartite graph $G = (V = A \cup B, E), E \subseteq A \times B$
- can be reduced to network flow:

- add edges $(s, A) \& (B, t)$, all capacities 1
- choose $\leq 1$ edge per vertex... in total
- Ford-Fulkerson uses integer flows if integer capacit.
- no splitting of unit flow e.g.,
- max flow = max cardinality matching
Augmenting paths:
- what does an augmenting path in the flow network look like in the matching?
- 1-flow edge \((u,v)\) isn’t in the residual graph... but its reverse \((v,u)\) is
- starts with 0-flow edge \((s,a_1)\), \(a_1 \in A\) = unmatched vertex \(a_1\) in \(A\)
- ends with 0-flow edge \((b_k,t)\), \(b_k \in B\) = unmatched vertex \(b_k\) in \(B\)
- in between, follow a path in \(G\):
  \[ a_1 \rightarrow b_1 \rightarrow a_2 \rightarrow b_2 \rightarrow \ldots \rightarrow a_k \rightarrow b_k \]
  - each \(a_i \rightarrow b_i\) must be 0-flow
    \[ \{a_i, b_i\} \text{ is not in matching} \]
  - each \(b_i \rightarrow a_{i+1}\) must be 1-flow
    \[ \{a_{i+1}, b_i\} \text{ is in matching} \]

\Rightarrow \text{augmenting path looks like: (without } s \text{ & } t) \]

i.e. an (odd-length) alternating path starting & ending with unmatched vertices

- what does augmentation do?
- flips 0-flows \(\Leftrightarrow\) 1-flows
- increases flow value by 1
Matching in general graph $G$:

- **Alternating path** = path in $G$ where every second edge is matched
- **Augmenting path** = alternating path where first & last vertices unmatched
  - can flip edges matched/unmatched along path
  - get one more edge in matching
  - $\Rightarrow$ wasn’t maximum cardinality

**Edmonds’ algorithm:** (high level) [1965]
- find an augmenting path $\Rightarrow$ how?
- flip it
- repeat until no augmenting paths $\Rightarrow$ enough?

**Augmentation is enough:** [Berge 1957]
- if matching has no aug. paths then max. cardinality

**Proof:** say $M$ has no augmenting paths & $M^*$ has maximum cardinality
- look at $M \oplus M^* = \text{XOR/symmetric difference}$
- $M \& M^*$ max. degree 1 $\Rightarrow M \oplus M^*$ max. degree 2
  $\Rightarrow$ paths
  and/or
  cycles:
- if $|M^*| > |M|$ then this type must exist $\Rightarrow$ that's an augmenting path. □
Finding an augmenting path:
- in bipartite graphs, this is easy (BFS/DFS):
  always unmatched edges A→B & matched edges B→A
  so guaranteed alternating
- general graphs have odd cycles:
  - need to try traversing in both directions
  (otherwise may miss opportunities)

Edmonds' blossoms: [1965] "Paths, Trees, & Flowers"
- do BFS/DFS/any locally advancing search
  - forced to follow matched edges half the time
- if encounter an odd cycle, contract it to form smaller graph G' & smaller matching M':
- can extend aug. path in G' to one in G:

(traverse clockwise or counterclockwise according to parity of unmatched edge used)
- so we've reduced finding an augmenting path to a smaller problem
- can just recurse
**Simple implementation:**
- for each unmatched vertex $s$:
  - DFS or BFS from $s$
    - at even depths (including $s$)
      - try all available edges not already used in that direction
    - at odd depths, forced to follow matching
  - if ever encounter another unmatched vertex: done, return augmenting path
  - if ever discover a cycle:
    - ignore if even
    - if odd: contract blossom
      - recurse
      - expand blossom

**Time:** $O(V)$ blossom-induced recursions (each decreases $|V|$)
- $O(V)$ unmatched vertices $s$
- $O(E)$ time for DFS/BFS (assume connected)
- $O(V^2E)$ per augmentation
- $O(N)$ augmentations (each increases $|M|$)
- $O(V^3E)$ total
**Improvements:**

- re-use "edge visited in this direction?" between BFS/DFS calls ⇒ avoid repeating >2x
  ⇒ \( O(V^3E) \) time
- don't actually contract blossoms, just carefully traverse them both ways
  ⇒ DFS with stack of blossoms for revisiting
  ⇒ \( O(VE) \) time [Kameda & Munro 1974]
  [Micali & Vazirani 1980]
  [Peterson & Loui 1988]
- best algorithm to date: \( O(V^3E) \) time
  [Mucha & Sankowski 2004]
- idea: re-use structure from one
  augmenting path search to the next
- for dense graphs: \( O(V^{2.376}) \) via fast matrix multiplication
Weighted matching: given graph $G=(V,E)$ & edge weights $w:E \rightarrow \mathbb{R}_+$

- find matching of maximum total weight
  - can drop edges of negative weight
  - can add edges of zero weight (complete graph)

$\Rightarrow$ find perfect matching of maximum weight

- algorithms use blossoming + more
  - first: $O(V^4)$ [Edmonds 1965]
  - best: $O(VE \log V)$ [Galil, Micali, Gabow 1982]
    & [Ball & Derigs 1983]

Bipartite case: "assignment problem" highly motivated

- suffices to repeatedly find augmenting path of maximum weight, where matched edges get negative weight & unmatched get positive.

  - invariant: max-weight matching of $t$ edges
  - proof: by induction on $t$
    - $M_{t-1} \lor M_t^* = \text{alt. paths} \& \text{cycles}$
      - even weight 0
      - weight 0

  - look add odd alt. paths
    - some have one more edge in $M_{t-1}$ (kind 1)
    - while others \(\text{---} \text{---} \text{---} \text{---} M_t^*\) (kind 2)

  - claim: exists one more alt. path of second kind
    - (because $M_t^*$ has 1 more edge)

  - pair up kind 1 & kind 2 paths so that one kind 2 path $p$ is left unpaired
• **Claim:** All paired up paths have total wgt 0.
  (otherwise we could increase the weight of either \( M_{t-1} \) or \( M^*_t \) by flipping matched/unmatched edges)

• Hence:
  \[
  \text{weight}(M^*_t) = \text{weight}(M_{t-1}) + \text{weight}(P).
  \]

• Note that \( P \) is an alternating path w.r.t. matching \( M_{t-1} \).

• Since \( M_t \) is derived from \( M_{t-1} \) by choosing the max-weight alternating path
  \[
  \text{wgt}(M_t) \geq \text{wgt}(M_{t-1}) + \text{wgt}(P) \geq \text{wgt}(M^*_t). \]

- **Note:** no negative weight cycles, as otherwise we could flip matched/unmatched edges on cycle violating our invariant.

- Direct matched edges \( A \rightarrow B \) & unmatched \( B \rightarrow A \)
  \Rightarrow shortest path problem (if we reverse sign of wts on edges)

- \( |V| \times \text{Bellman-Ford} \Rightarrow O(V^2E) \) time

- Johnson trick \Rightarrow O(VE + V^2 \log V) time