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

\( G = (V, E) \) undirected graph

Matching = set \( M \) of edges sharing no endpoints
- cardinality \( |M| \) = # edges in \( M \)
- goal: given undirected graph, find max. cardinality matching
- perfect if \( |M| = |V|/2 \) (note: maximum vs. maximal)

(note: weighted version) \( \rightarrow \) boys vs. girls, doctors vs. hospitals, etc.

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

\[ \text{add edges } (s, A) \text{ & } (B, t), \text{ all capacities 1} \]
\[ \Rightarrow \text{choose } \leq 1 \text{ edge per vertex... in total} \]
\[ \Rightarrow \text{Ford-Fulkerson uses integer flows if integer capac.} \]
\[ \Rightarrow \text{no splitting of unit flow e.g.} \rightarrow \]
\[ \text{max flow = max cardinality matching} \]
Augmenting paths:
- what does an augmenting path in the flow network look like in the matching?
- 1-flow edge \((uv)\) 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
    \(\{a_i, b_i\}\) is not in matching
  - each \(b_i \rightarrow a_{i+1}\) must be 1-flow
    \(\{a_i, b_i\}\) is in matching
\(\Rightarrow\) augmenting path looks like: (without \(s\) & \(t\))

\[
\begin{align*}
\text{EA} & \rightarrow \text{EB} & \text{EA} & \rightarrow \text{EB} & \text{EA} & \rightarrow \text{EB} & \text{EA} & \rightarrow \text{EB} & \text{EA} & \rightarrow \text{EB} & \text{EA} & \rightarrow \text{EB}
\end{align*}
\]
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
Noirey alg: time $\Theta(V \cdot E)$ (Edmonds)

Hopcroft & Karp: unweighted bipartite matching in time $O(V^{1.5} \cdot E)$

(idea: find many augmenting paths at once.)
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) \[^{[1965]}\] [how?]
- find an augmenting path
- flip it
- repeat until no augmenting paths \[^{\rightarrow \text{enough?}}\]

**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\[^{\text{and/or}}\]
cycles\[^{\text{if } |M^*| > |M| \text{ then this type}}\]
must exist $\Rightarrow$ that's an augmenting path $\square$
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...

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
Note: no aux path here

found odd cycle

recurr on $G'$
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:
\[
O(V) \text{ blossom-induced recursions (each decreases } |V|) \\
O(V) \text{ unmatched vertices } s \\
O(E) \text{ time for DFS/BFS (assume connected) } \\
\Rightarrow O(V^2E) \text{ per augmentation} \\
O(V) \text{ augmentations (each increases } |M|) \\
O(V^3E) \text{ total}
\]
Improvements:
- re-use “edge visited in this direction?” between BFS/DFS calls ⇒ avoid repeating >2x
  ⇒ O(V^2E) time
- don’t actually contract blossoms, just carefully traverse them both ways
  ⇒ O(V^2E) time  ⇒ O(VE) time  [Kameda & Munro 1979]
  DFS with stack of blossoms for revisiting  [Micali & Vazirani 1980]
- best algorithm to date: [Peterson & Loui 1988]
  O(V^1.376) time
- idea: re-use structure from one augmenting path search to the next
- for dense graphs: [Mucha & Sankowski 2004]
  O(V^2.376) via fast matrix multiplication
Weighted matching: given graph $G = (V, E)$ & edge weights $w: E \rightarrow \mathbb{R}_+$
- can drop edges of negative weight
- can add edges of zero weight to complete graph
$\Rightarrow$ find perfect matching of maximum weight
- algorithms use blossoming + more
  - first: $O(V^4)$ \cite{Edmonds 1965}
  - best: $O(V E \log V)$ \cite{Galil, Micali, Gabow 1982} & \cite{Ball & Derigs 1983}

Bipartite case: "assignment problem" ~ highly motivated
- suffices to repeatedly find augmenting paths of minimum weight, where matched edges get positive weight & unmatched get negative
- invariant: max-weight matching of $t$ edges
  - proof: by induction on $t$
  - $M_{t-1}$ xor $M_t^*$ = all paths & cycles
  - our solution = OPT  \quad OPT  \quad\text{even weight} 0  \quad\text{weight} 0
  - can join odd paths together to aug path...

\[
M_t \geq \omega(M_t^*) \Rightarrow \square
\]
- direct matched edges $A \rightarrow B$ & unmatched $B \rightarrow A$
- shortest path problem (if we reverse sign of weight on matched edges)
- $|V| \times$ Bellman-Ford $\Rightarrow O(V^2 E)$ time
- Johnson trick $\Rightarrow O(V E + V^2 \log V)$ time