Applications of Network Flow:
Solving Problems by Reduction to Network Flows
Reductions

• We will solve these problems by reducing them to a network flow problem

• We'll focus on the concept of problem reductions
Anatomy of Problem Reductions

At a high level, a problem $X$ reduces to a problem $Y$ if an algorithm for $Y$ can be used to solve $X$

- **Reduction.** Convert an arbitrary instance $x$ of $X$ to a special instance $y$ of $Y$ such that there is a 1-1 correspondence between them.
Anatomy of Problem Reductions

- **Claim.** $x$ satisfies a property iff $y$ satisfies a corresponding property.
- Proving a reduction is correct: prove both directions.
- $x$ has a property (e.g. has matching of size $k$) $\implies$ $y$ has a corresponding property (e.g. has a flow of value $k$).
- $x$ does not have a property (e.g. does not have matching of size $k$) $\implies$ $y$ does not have a corresponding property (e.g. does not have a flow of value $k$).
- Or equivalently (and this is often easier to prove):
  - $y$ has a property (e.g. has flow of value $k$) $\implies$ $x$ has a corresponding property (e.g. has a matching of value $k$).
Remaining Plan

We will explore one application of network flow in detail today

- Matching in bipartite graphs
- Matchings are super practical with many applications
- We have already seen one, can you remember?

Next meeting: another application reducible to network flow (playoff elimination)

- More practice with reductions
- (Reductions will come in handy on our next topic too!)
Bipartite Matching
Definition. Given an undirected graph $G = (V, E)$, a matching $M \subseteq E$ of $G$ is a subset of edges such that no two edges in $M$ are incident on the same vertex.

- Said differently, a node appears as an endpoint in at most one edge in $M$. 
Review: Matching in Graphs

A perfect matching matches all nodes in $G$

- Max matching problem. Find a matching of maximum cardinality for a given graph
  - That is, a matching with maximum number of edges
- Observation: If it exists, a perfect matching is maximum!
A graph is **bipartite** if its vertices can be partitioned into two subsets $X, Y$ such that every edge $e = (u, v)$ connects $u \in X$ and $v \in Y$

- **Bipartite matching problem.** Given a bipartite graph $G = (X \cup Y, E)$ find a maximum matching.
Bipartite Matching Examples

Can be used to model many assignment problems, e.g.:

- $A$ is a set of jobs, $B$ as a set of machines
- Edge $(a_i, b_j)$ indicates where machine $b_j$ is able to process job $a_i$
- Perfect matching: a way to assign each job to a machine that can process it, such that each machine is assigned exactly one job
- Assigning customers to stores, students to dorms, etc.

Note. This is a different problem than the one we studied for Gale-Shapely matching!
Maximum & Perfect Matchings

• One of the oldest problems in combinatorial algorithms:
  • Determine the largest matching in a bipartite graph
• This doesn't seem like a network flow problem, but we will turn it into one!
• Special case: Find a perfect matching in $G$ if it exists
  • What conditions do we need for perfect matching?
    • Certainly need $|A| = |B|$,
  • What are the necessary and sufficient conditions?
  • Will use network flow to determine
Reduction to Max Flow

- Given arbitrary instance $x$ of bipartite matching problem ($X$): $A, B$ and edges $E$ between $A$ and $B$

- **Goal.** Create a special instance $y$ of a max-flow problem ($Y$): flow network: $G(V, E, c)$, source $s$, sink $t \in V$ s.t.

- **1-1 correspondence.** There exists a matching of size $k$ iff there is a flow of value $k$
Reduction to Max Flow

• Create a new directed graph $G' = (A \cup B \cup \{s, t\}, E', c)$
• Add edge $s \to a$ to $E'$ for all nodes $a \in A$
• Add edge $b \to t$ to $E'$ for all nodes $b \in B$
• Direct edge $a \to b$ in $E'$ if $(a, b) \in E$
• Set capacity of all edges in $E'$ to 1
Correctness of Reduction

- **Claim** ($\Rightarrow$).
  If the bipartite graph $(A, B, E)$ has matching $M$ of size $k$ then flow-network $G'$ has an integral flow of value $k$. 

![Graphs](image)
Correctness of Reduction

- **Claim** ($\Rightarrow$).
  If the bipartite graph $(A, B, E)$ has matching $M$ of size $k$ then flow-network $G'$ has an integral flow of value $k$.

- **Proof** (Longer proof in textbook).
  - For every edge $e = (a, b) \in M$, let $f$ be the flow resulting from sending 1 unit of flow along the path $s \to a \to b \to t$
  - $f$ is a feasible flow (satisfies capacity and conservation) and integral
  - $v(f) = k$
Correctness of Reduction

- **Claim (⇐).**
  
  If flow-network $G'$ has an integral flow of value $k$, then the bipartite graph $(A, B, E)$ has matching $M$ of size $k$. 

Correctness of Reduction

- **Claim** (⇐).
  If flow-network $G'$ has an integral flow of value $k$, then the bipartite graph $(A, B, E)$ has matching $M$ of size $k$.

- **Proof**.
  - Let $M = \text{set of edges from } A \text{ to } B \text{ with } f(e) = 1$.
  - No two edges in $M$ share a vertex, why?
  - $|M| = k$
    - $v(f) = f_{\text{out}}(S) - f_{\text{in}}(S)$ for any $(S, V - S)$ cut
  - Let $S = A \cup \{s\}$

Edge capacities are 1
Summary & Running Time

- Proved matching of size $k$ iff flow of value $k$
- Thus, max-flow iff max matching
- Running time of algorithm overall:
  - Running time of reduction + running time of solving the flow problem (dominates)
- What is running time of Ford–Fulkerson algorithm for a flow network with all unit capacities?
  - $O(nm)$
- Overall running time of finding max-cardinality bipartite matching: $O(nm)$
Acknowledgments

- Some of the material in these slides are taken from
  - Jeff Erickson’s Algorithms Book (http://jeffe.cs.illinois.edu/teaching/algorithms/book/Algorithms-JeffE.pdf)
  - Shikha Singh