EECS 336: Introduction to Algorithms **Lecture X**
Final Review

Reading: All chapters except 9, 10, 12, and 13.

Problems and Algorithms

- Maximum Network Flow (Ford-Fulkerson)
  - Augmenting Paths in Residual Graphs
  - $O(nC)$ (or $O(m^2 \log C)$ if clever)
- Bipartite Matching
  - reduction to network flow
  - $O(n^2)$
- NP
  - input: verifier program V
  - does certificate exist that program verifies.
  - NP is NP-complete.
- CIRCUIT-SAT
  - exists input to boolean circuit to make it output true?
  - NP $\leq_P$ CIRCUIT-SAT
- 3-SAT
  - exists assignment to veriables to satisfy 3-SAT formula.
  - CIRCUIT-SAT $\leq_P$ 3-SAT
- INDEP-SET
  - subset of non-adjacent vertices with largest cardinality.
  - 3-SAT $\leq_P$ INDEP-SET
- Vertex Cover
  - subset of vertices that cover all edges.
  - INDEP-SET $=_P$ Vertex Cover
- Hamiltonian Cycle
  - exists tour in graph = cycle that visits each vertex exactly once.
  - 3-SAT $\leq_P$ HAMILTONIAN-CYCLE.
- TSP
  - find cheapest cost tour
  - HAMILTONIAN-CYCLE $\leq_P$ TSP.
  - not approximable to any factor.
- METRIC-TSP
  - costs obey triangle inequality.
  - 2-approximation via MST.
- Knapsack with Integer Sizes (or Values)
  - DP.

    - $O(nC)$ or $O(n^2 v_{\max})$.
    - pseudo-polynomial time.
- Knapsack (general)
    - NP-complete
    - 2-approximation via "Greedy or Max" ($O(nlogn)$)
    - PTAS via Integer-Size Knapsack DP ($O(n^3/\epsilon)$)

Techniques

- greedy
- divide and conquer
- dynamic programming
    - always write subproblem in english.
- reduction (X to Y)
    - turn instance of X into instance of Y
    - iff.
- NP-completeness
    - in NP.
    - NP-hardness reduction:
        - construction
        - runtime of construction
        - correctness of construction (iff)
    - algorithms in reductions:

```
          3-SAT          INDEP-SET
input:     f       =>     G,D
output:    z      <=>      S
```

- approximation (maximization)
    - upper bound on OPT
    - lower bound on Alg

Algorithm Design Flow Chart