Graphs (cont.)

Reading: Sections 3.3, 3.5 and 3.6
Topological Sorting Algorithm

TPsort(G)

while there are nodes remaining do
    Find a node v with no incoming edges
    Place v next in the order
    Delete v and all of its outgoing edges from G
end while
Why Does It Work?

**Theorem:** Graph \( G \) is a DAG if and only if \( G \) has a topological ordering.

**Claim** \(_1\): In every DAG \( G \), there is a node \( v \) with no incoming edges.

**Claim** \(_2\): If \( G \) has a topological ordering, then \( G \) is a DAG.
Running Time

- Maintain the following information:
  - For each node, the current number of incoming edges - array $in[v]$
  - Set of remaining nodes with no incoming edges - $S$
Running Time

- Maintain the following information:
  - For each node, the current number of incoming edges - array $in[v]$
  - Set of remaining nodes with no incoming edges - $S$

- Initialization: ??

- Update: to delete $v$
  - remove $v$ from $S$
  - decrement $in[u]$ for all edges from $v$ to $u$;
  - Add $u$ to $S$ if $in[u]$ hits 0
  - $O(1)$ per edge
Running Time

- Maintain the following information:
  - For each node, the current number of incoming edges - array $in[v]$
  - Set of remaining nodes with no incoming edges - $S$

- Initialization: ??

- Update: to delete $v$
  - remove $v$ from $S$
  - decrement $in[u]$ for all edges from $v$ to $u$;
  - Add $u$ to $S$ if $in[u]$ hits 0
  - $O(1)$ per edge

- Running time: ??
Greedy Algorithms
Coin Change

Problem: You work as a cashier and need to give back change once in a while. Your goal is to do that using the fewest number of coins possible. Assume that you have as much coins as you need from a penny to a dollar.

● What is the solution if change is $0.78? How about $2.56?

● What algorithm did you use to find the solution?

● Would your algorithm work if the values of the coins were different?
Greedy Approach

- It builds up a solution in small steps.

- It chooses a decision at each step to optimize some underlying criterion.

- There can be different greedy algorithms for the same problem, each one locally, incrementally optimizing some different measure on its way to a solution.

- When a greedy algorithm succeeds in solving a nontrivial problem optimally, it typically implies something interesting and useful about the structure of the problem.
Review
• Stable Matching
  ○ What is a stable matching?
  ○ Different notions of instability
  ○ Gale-Shapley algorithm
    ■ Termination/Running time
    ■ Correctness
• Asymptotic Notation
  ○ Definition of bigO, bigOmega, bigTheta
  ○ How is it used to compare algorithms?
• Graphs
  ○ Definitions, Representation (undirected and directed)
  ○ Search
  ○ Connectivity
  ○ Topological sorting