Elementary Graph Algorithms

Chapter 22

Topological Sort

- A topological sort is performed on a directed acyclic graph
- A topological sort is a linear ordering of all vertices of a graph such that if G contains an edge (u, v), then u appears before v in the ordering

Topological Sort

- A topological sort of a graph can be viewed as an ordering of its vertices along a horizontal line so that all directed edges go from left to right
- Directed Acyclic Graphs (DAG) are used in many applications to indicate precedences among events
- What is a DAG?

Topological Sort

- Good for modeling processes and structures that have a partial order:
  - a > b and b > c implies that a > c
  - But may have a and b such that neither a > b nor b > c

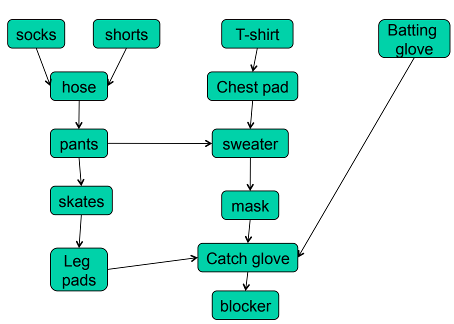
4/4/11 CS380 Algorithm Design and Analysis 4

TOPOLOGICAL-SORT(G)

- Call DFS(G) to compute finishing times f[v] for each vertex v
- As each vertex is finished, insert it onto the front of a linked list
- Return the linked list of vertices

4/4/11 CS380 Algorithm Design and Analysis 5

Example



4/4/11 CS380 Algorithm Design and Analysis 6

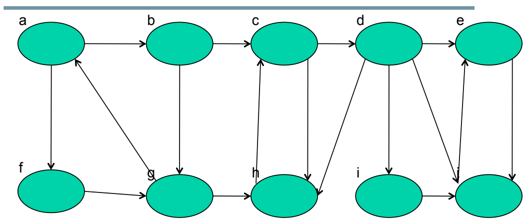
Topological Sort

- Running time for topological sort is:

Strongly Connected Components

- Given a directed graph G = (V, E)
- A strongly connected component (SCC) of G is a maximal set of vertices $C \subseteq V$
- Such that for all $u, v \in C$ both u -> v and v->u

Example



- Identify the strongly connected components

Transpose

- Algorithm uses G^T = transpose of G
  - G^T
- How long does it take to create G^T if using adjacency lists?
- Observation: G and G^T have the same SCC's.

SCC(G)

- Call DFS(G) to compute finishing times f[u] for all u
- Compute G^T
- Call DFS(G^T), but in the main loop, consider vertices in order of decreasing f[u] (as computed in first DFS)
- Output the vertices in each tree of the depth-first forest formed in second DFS as a separate SCC

Example