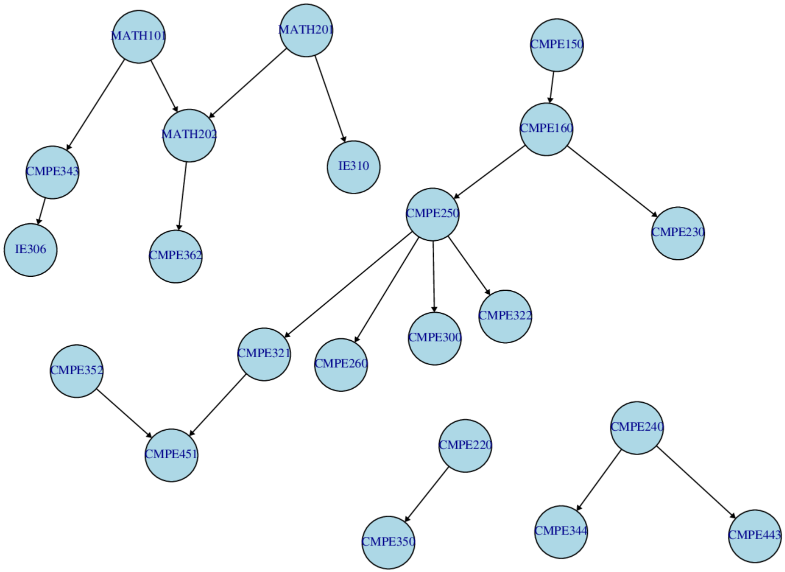
Graphs-Topological Sort

November 9, 2016

Introduction

- There are many problems involving a set of tasks in which some of the tasks must be done before others.
- For example, consider the problem of taking a course only after taking its prerequisites.
- Is there any systematic way of linearly arranging the courses in the order that they should be taken?

CMPE Course Prerequisites



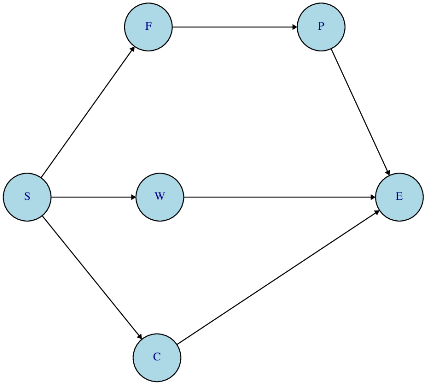
Problem

- **P**: Assemble pingpong table
- **F**: Carpet the floor
- **W**: Panel the walls
- **C**: Install ceiling
- How would you order these activities?

Problem Graph

S: Start **E**: End
P: Assemble pingpong table **F**: Carpet the floor
W: Panel the walls **C**: Install ceiling

Some possible orderings:
C W F P
W C F P
F P W C
C F P W



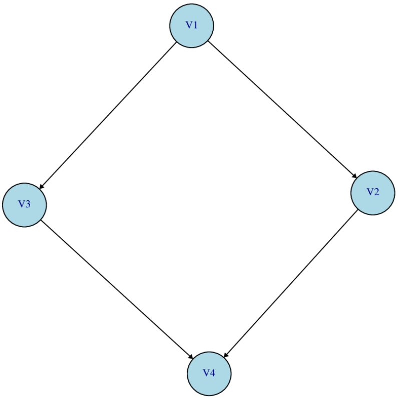
Topological Sort

- RULE:
  - If there is a path from u to v, then v appears after u in the ordering.

Graphs' Characteristics

- Directed
  - otherwise (u, v) means a path from u to v and from v to u , cannot be ordered.
- Acyclic
  - otherwise u and v are on a cycle : u would precede v , v would precede u.

The ordering may not be unique



Legal orderings

V1, V2, V3, V4

V1, V3, V2, V4

Basic Algorithm

1. Compute the indegrees of all vertices
2. Find a vertex U with **indegree 0** and print it (store it in the ordering)
   If there is **no** such vertex then **there is a cycle** and the vertices cannot be ordered. Stop.
3. Remove U and all its edges (U, V) from the graph.
4. **Update** the indegrees of the remaining vertices.

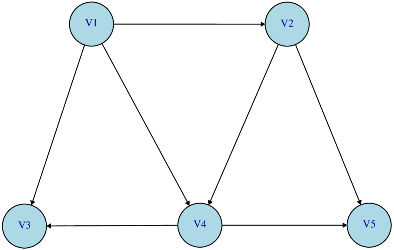
Repeat steps 2 through 4 while there are vertices to be processed.

Example

Indegrees
V1 0
V2 1
V3 2
V4 2
V5 2

First to be sorted: V1

New indegrees:
V2 0
V3 1
V4 1
V5 2



Possible sorts: V1, V2, V4, V3, V5; V1, V2, V4, V5, V3

Complexity

- $O(|V|^2)$, $|V|$ - the number of vertices.
- To find a vertex of indegree 0 we scan all the vertices - $|V|$ operations.
- We do this for all vertices: $|V|^2$ operations

Improved Algorithm

- Find a vertex of degree 0,
- Scan **only** those vertices whose indegrees have been **updated to 0**.

Improved Algorithm

1. Compute the indegrees of all vertices
2. Store all vertices with indegree 0 in a queue.
3. Get a vertex U.
4. For all edges (U, V) update the indegree of V, and put V in the queue if the updated indegree is 0.

Repeat steps 3 and 4 while the queue is not empty.

Complexity

- $|V|$ - number of vertices,
- $|E|$ - number of edges.
- Operations needed to compute the indegrees:
  - Adjacency lists: $O(|E|)$
  - Adjacency Matrix representation: $O(|V|^2)$
- Complexity of the improved algorithm
  - Adjacency lists: $O(|E| + |V|)$,
  - Adjacency Matrix representation: $O(|V|^2)$
- Note that if the graph is complete $|E| = O(|V|^2)$