CS 312: Algorithms

Dan Sheldon

Mount Holyoke College

Last Compiled: September 26, 2018

Today

- Bipartite testing
- Directed graphs

Graph Traversal

- BFS/DFS: $\Theta(m+n)$ (linear time) graph primitives for:
  - Path from s to t?
  - Connected components
  - Subroutine in other algorithms
- Can be modified to solve related problems
  - Sometimes properties of BFS/DFS trees are useful
- Example: Bipartite Testing

Bipartite Graphs

Definition Graph $G = (V, E)$ is bipartite if V can be partitioned into sets X, Y such that every edge has one end in X and one in Y.

Can color nodes red and blue s.t. no edges between nodes of same color.

Examples (board work)

- Bipartite: student-college graph in stable matching
- Bipartite: client-server connections
- Not bipartite: "odd cycle" (cycle with odd # of nodes)
- Not bipartite: any graph containing odd cycle

Claim (easy): If G contains an odd cycle, it is not bipartite.

Bipartite Testing

Question Given $G = (V, E)$, is G bipartite?

Algorithm? BFS? **Idea**: run BFS from any node s

- L_0 = red
- L_1 = blue
- L_2 = red
- ...
- Even layers red, odd layers blue

What could go wrong? Edge between two nodes at same layer.

Algorithm

Run BFS from any node s
if there is an edge between two nodes in same layer **then**
  Output "not bipartite"
else
  X = even layers
  Y = odd layers
end if

Correctness? Recall: all edges between same or adjacent layers.

1. No edges between nodes in same layer $\Rightarrow$ correct labeling, G bipartite.
2. Edge between two nodes in the same layer $\Rightarrow$ G has an odd cycle, not bipartite. Proof on board.

Proof

- Let T be BFS tree of G and suppose (x, y) is an edge between two nodes in the layer j
- Let $z \in L_i$ be the least common ancestor of x and y
  - Let P_{zx} = path from z to x in T
  - Let P_{yz} = path from z to y in T
  - The path that follows P_{zx} then edge (x, y) then P_{yz} is a cycle of length $2(j - i) + 1$, which is odd
- Therefore G is not bipartite.

Directed Graphs

$G = (V, E)$

- $(u, v) \in E$ is a *directed* edge
- u points to v

Examples

- Facebook: undirected
- Twitter: directed
- Web: directed
- Road network: directed

Directed Graph Traversal

Reachability. Find all nodes reachable from some node s.

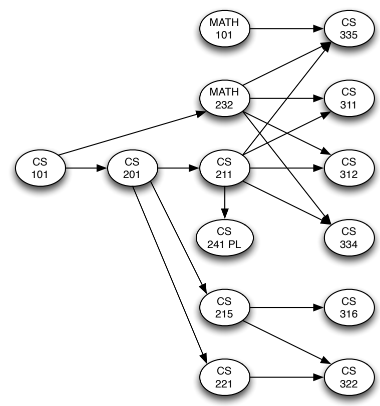
s-t shortest path. What is the length of the shortest directed path from s to t?

Algorithm? BFS naturally extends to directed graphs. Add v to L_{i+1} if there is a *directed* edge from L_i and v is not already discovered.

Directed Acyclic Graphs

Definition A directed acyclic graph (DAG) is a directed graph with no cycles.

Models *dependencies*, e.g. course prerequisites.



Topological Sorting

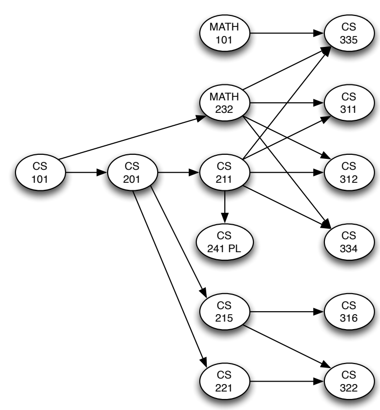
Definition A topological ordering of a directed graph is an ordering of the nodes such that all edges go "forward" in the ordering. Example on board.

- Label nodes $v_1, v_2, \ldots, v_n$ such that
- For all edges (v_i, v_j) we have $i < j$
- A way to order the classes so all prerequisites are satisfied

Topological Sorting

Exercise

1. Find a topological ordering.
2. Devise an algorithm to find a topological ordering. (Hint: it is not a modification of BFS or DFS.)



Topological Sorting

Problem Given DAG G, compute a topological ordering for G.

topo-sort(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

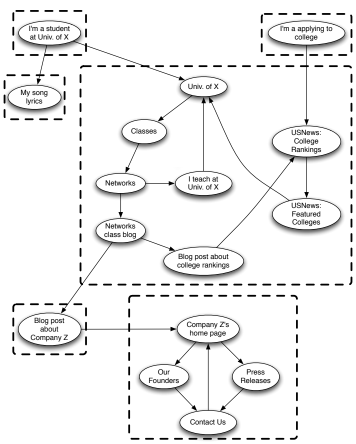
Running time? $O(m+n)$ — details omitted

Topological Sorting Analysis

- This algorithm is "obviously correct" if we can always find a node v with no incoming edges.
- **Claim**: G is a DAG $\Rightarrow$ G has a node with no incoming edges. Proof sketch on board.

Theorem: G is a DAG if and only if G has a topological ordering.

Directed Graph Connectivity

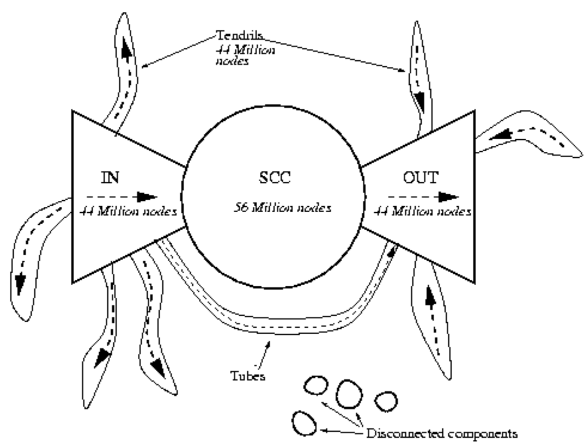


Strongly connected graph. Directed path between any two nodes.

Strongly connected component (SCC). Maximal subset of nodes with directed path between any two.

SCCs can be found in $O(m+n)$ time. (Tarjan, 1972)

Bow-Tie Structure of Web



Graphs Summary

- Graph Traversal
  - BFS/DFS, Connected Components, Bipartite Testing
  - Traversal Implementation and Analysis
- Directed Graphs
  - Directed Acyclic Graphs
  - Topological ordering
  - Strong Connectivity