15.082J, 6.855J, and ESD.78J
Sept 16, 2010

Lecture 3. Graph Search

Breadth First Search

Depth First Search

Intro to program verification

Topological Sort

Overview

Today: Different ways of searching a graph

- **a generic approach**
- **breadth first search**
- **depth first search**
- **program verification**
- **data structures to support network search**
- **topological order**

- **Fundamental for most algorithms considered in this subject**

Searching a Directed Graph

ALGORITHM SEARCH

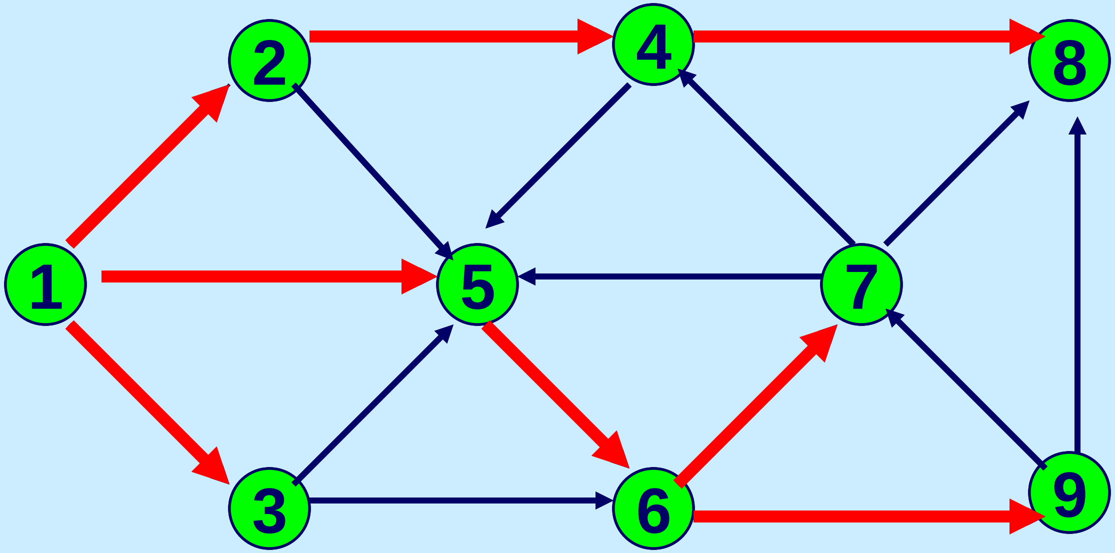
INPUT: A directed network G, and node s

OUTPUT: The set S = {j : there is a directed path from s to j in G}.

These are the nodes *reachable* from s. For each node j ∈ S\s,

pred(j) is a node that precedes j on some path from s;

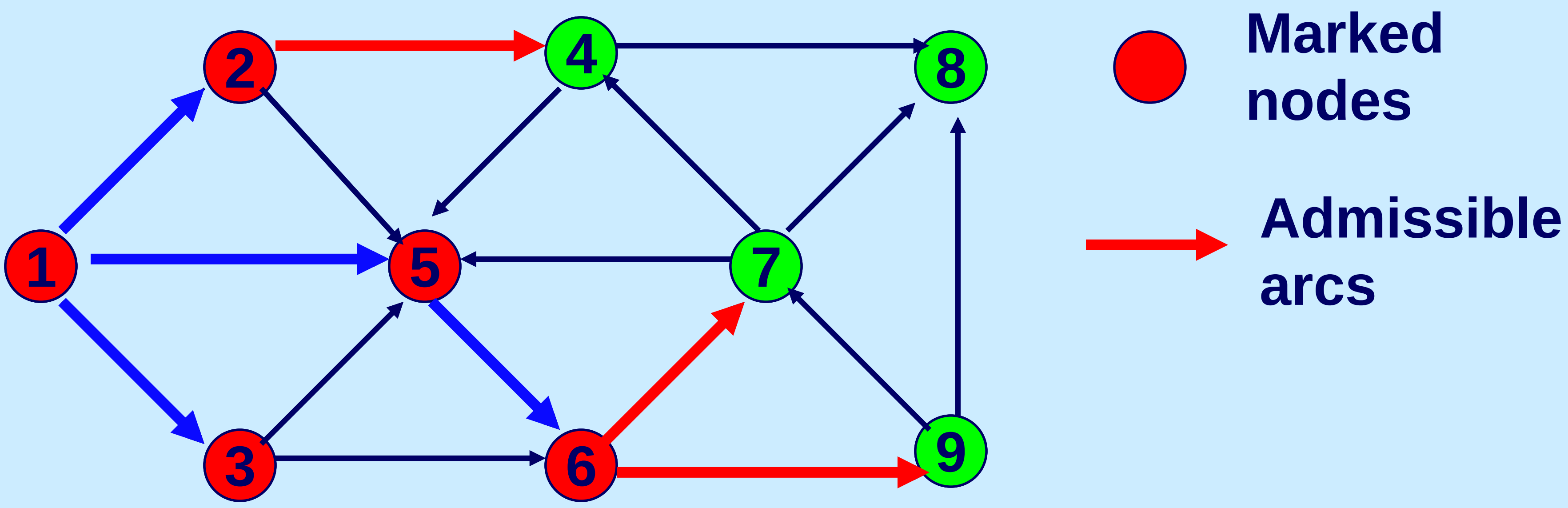
e.g. (pred(2) = 1, pred(8) = 4



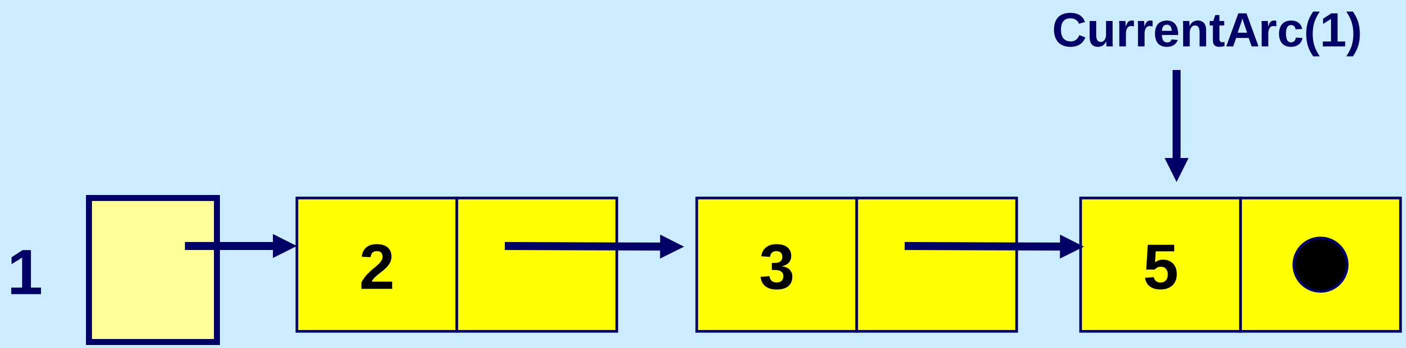
Marked nodes, admissible arcs

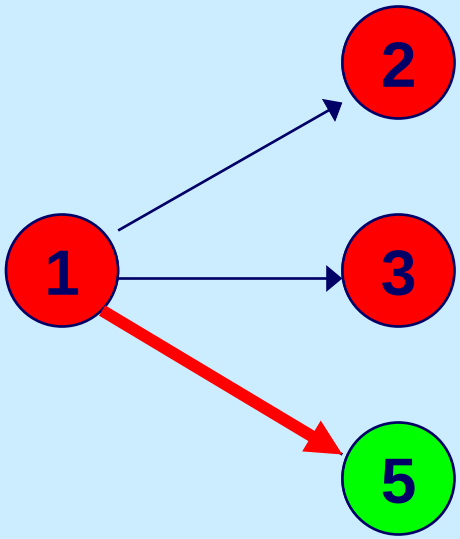
A node is either *marked* or *unmarked*. Initially only node s is marked. If a node is marked, it is reachable from node s.

An arc (i,j) ∈ A is *admissible* if node i is marked and j is not.



Scanning arcs





Scan through the arc list for the selected node, and keep track using current arc. Stop when an admissible arc is identified or when the arc list is fully scanned

Algorithm Search

Initialize as follows:

unmark all nodes in N;

mark node s;

pred(s) = 0; {that is, it has no predecessor}

LIST = {s}

while LIST ≠ ø do

select a node i in LIST;

if node i is incident to an admissible arc (i,j) then

mark node j;

pred(j) := i;

add node j to the end of LIST;

else delete node i from LIST

The algorithm in the book also keeps track of the order in which nodes are marked.

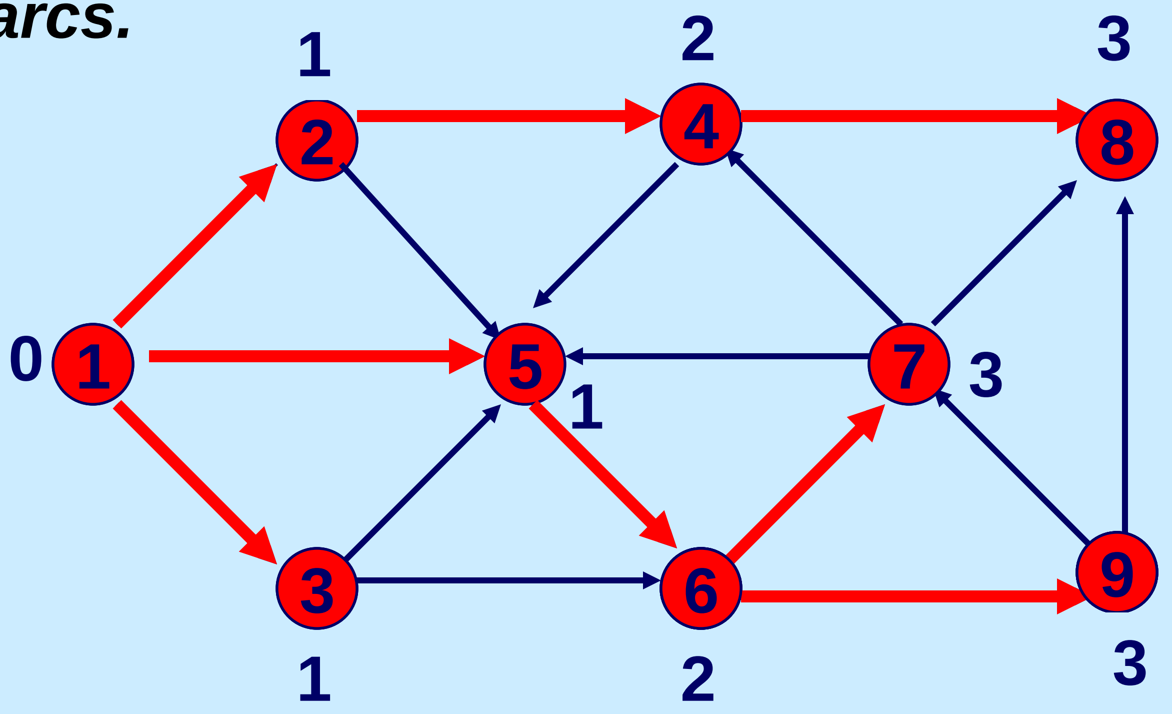
Breadth first search

It is a breadth first search (bfs) if the selected node is the first node on LIST.

Breadth First Search Animation

More on Breadth First Search

Theorem. ***The breadth first search tree is the "shortest path tree", that is, the path from s to j in the tree has the fewest possible number of arcs.***



The numbers next to the nodes are the distances from node 1.

Depth first search

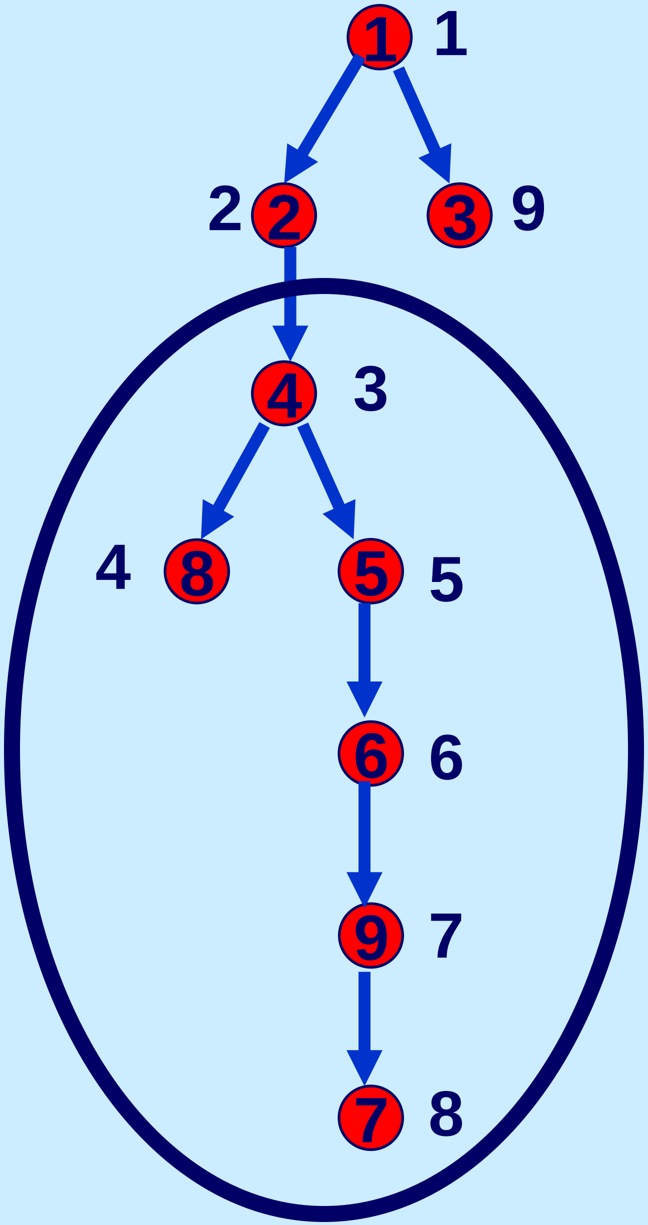
It is a depth first search (dfs) if the selected node is the last node on LIST.

Depth First Search Animation

The depth first search tree

Note that each induced subtree has consecutively labeled nodes.

(The descendents are visited in order.)



Algorithm Analysis

How does one prove that an algorithm is correct?

How does one prove that it terminates in finite time?

How does one obtain a tight upper bound on running time?

Some useful approaches

Subdivide the algorithm into "chunks".

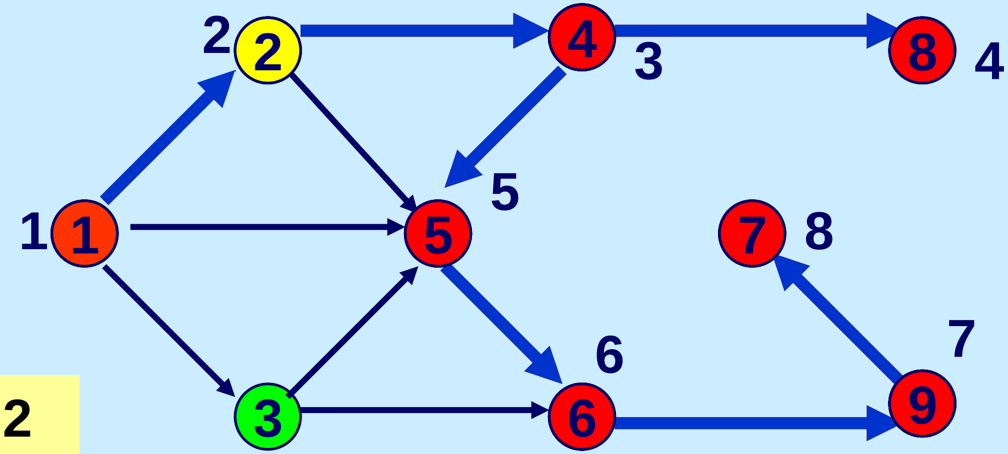
Invariants: **properties that are true throughout the running of the algorithm**

Things that change: **functions that increase monotonically every time the algorithm reenters the same loop.**

Algorithm Search

```
        Initialize
loop    while LIST ≠ ø do
            select a node i in LIST;
            if node i is incident to an admissible arc (i,j) then
                mark node j;
                pred(j) := i;
                add node j to the end of LIST;
            else delete node i from LIST
```

Algorithm Invariants



Select Node 2

LIST

1	2							

Invariants: whenever control of the program is at loop:

1. **Any marked node is reachable from s.**
2. **All nodes on LIST are marked.**
3. **If a marked node j is not on LIST, then A(j) has been fully scanned.**

Proving the correctness of the invariants

An Important step in proving algorithm correctness:

Prove that the algorithm invariants are true using induction.

- **Prove that they are true after the initialization**
- **Assuming that they are true at the beginning of the k-th iteration of the while loop, prove that they are true at the beginning of the subsequent iteration of the while loop.**

Things that change and proof of finiteness

Things that change between successive times that the control of the program is at loop:

1. **Either a new node is marked and added to LIST, or a new node is fully scanned and deleted from LIST.**

Number of iterations of while loop is at most 2n.

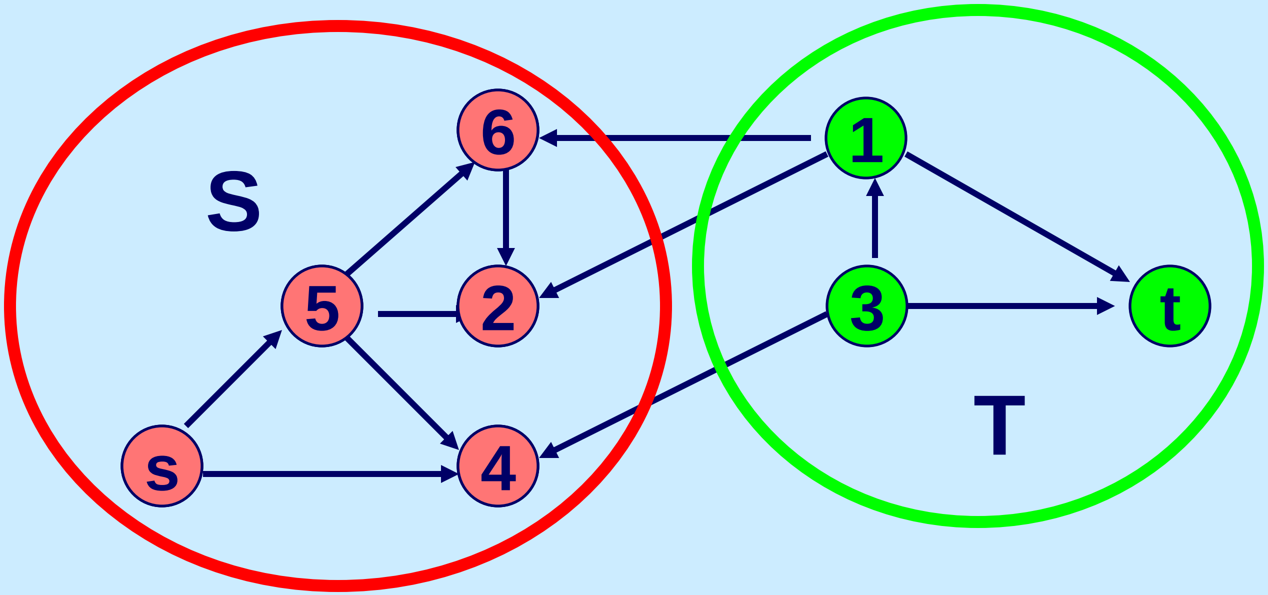
Therefore the algorithm terminates in O(n) calls of the "while loop."

Proof of Correctness

The algorithm terminates when LIST = ∅.

Let S = marked nodes.

Let T = unmarked nodes.



By invariant 1, all nodes in S are reachable from s.

By invariant 3, all arcs out of S have been scanned.

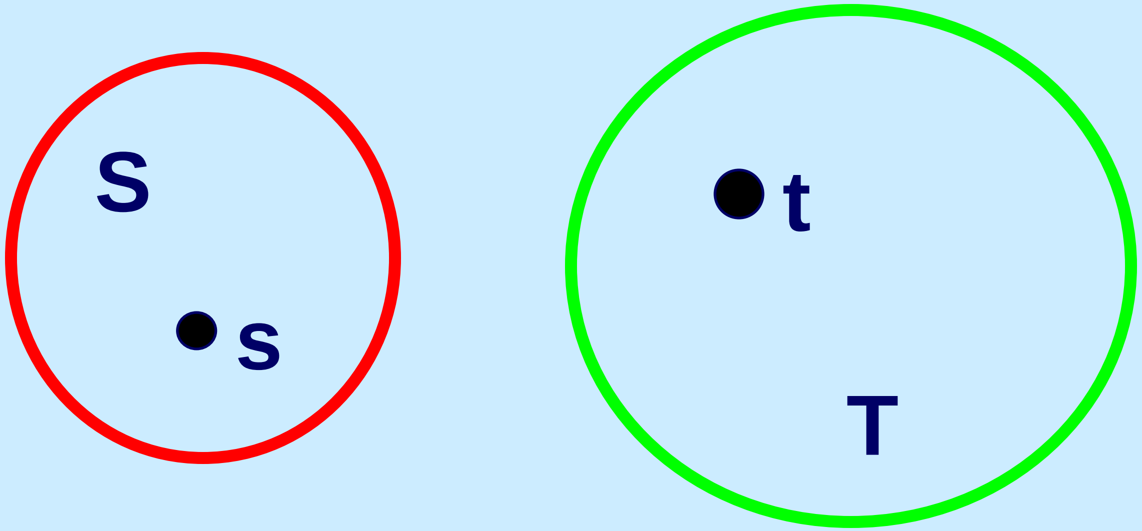
Then no arc (i,j) is directed from S to T. Otherwise, by Invariant 2, j would have been marked when (i, j) was scanned.

Therefore, no node in T is reachable from s.

Cutset Theorem

Corollary of algorithm's correctness. There is no directed path from s to t if and only if the following is true:

there is a partition of the node set N into subsets S and T = N - S such that there is no arc directed from a node in S to a node in T.



Running time analysis

```
        Initialize.
loop    while LIST ≠ ø do
            select a node i in LIST;
            if node i is incident to an admissible arc (i,j) then
                mark node j;
                pred(j) := i;
                add node j to the end of LIST;
            else delete node i from LIST
```

Total time spent in while loop (other than arc scans)

- O(1) time per loop
- < 2n iterations of the loop
- O(n) time in total

Total time spent in scanning arcs

- O(1) time per arc scanned
- m arcs
- O(m) time in total.

Running time: O(n + m)

Initialize

Initialize

begin

  unmark all nodes in N;

  mark node s;

  pred(s) = 0;   {that is, it has no predecessor}

  LIST = {s}

end

Unmarking takes O(n)
All else takes O(1)

Theorem*. *Algorithm search determines all nodes reachable from node s in O(n + m) time.

Mental Break

All U.S. Presidents have worn glasses

True

No President (before Obama) has been an only child

True

No President was a bachelor

False. James Buchanan was a bachelor.

George Washington grew marijuana on his Plantation.

True

Mental Break

George Washington's false teeth were made of wood.

False. They were made of whale bone.

Three of the first 10 Presidents died on July 4.

True. Thomas Jefferson, John Adams, James Monroe

John Quincy Adams kept a pet alligator in the East Room of the White House

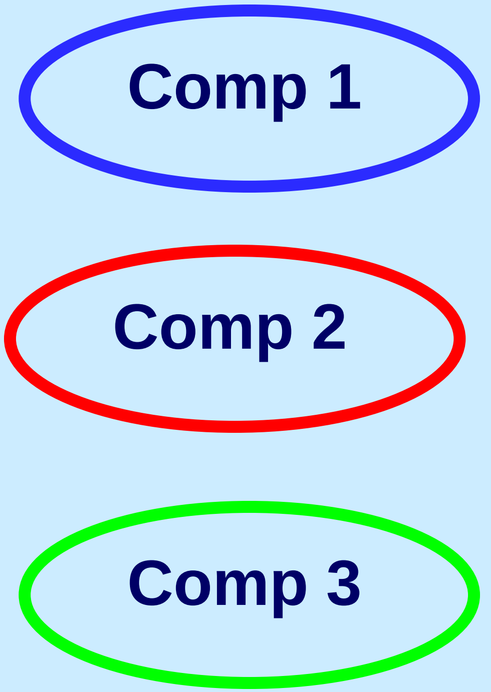
True.

Calvin Coolidge believed that the world was flat.

False. But Andrew Jackson did.

Finding all connected components in an undirected graph

Breadth first search will find a connected component of an undirected graph in time proportional to the number of arcs in the component. (A component of an undirected graph is a maximally connected subgraph.)



To find all components: Maintain a set U of unmarked nodes.

- Delete a node from U after it is marked.
- After each component is searched, select a node of U and begin a search.
- Running time O(1) per selection and deletion

Proof of bfs Theorem

Theorem. ***The breadth first search tree is the "shortest path tree", that is, the path from s to j in the tree has the fewest possible number of arcs.***

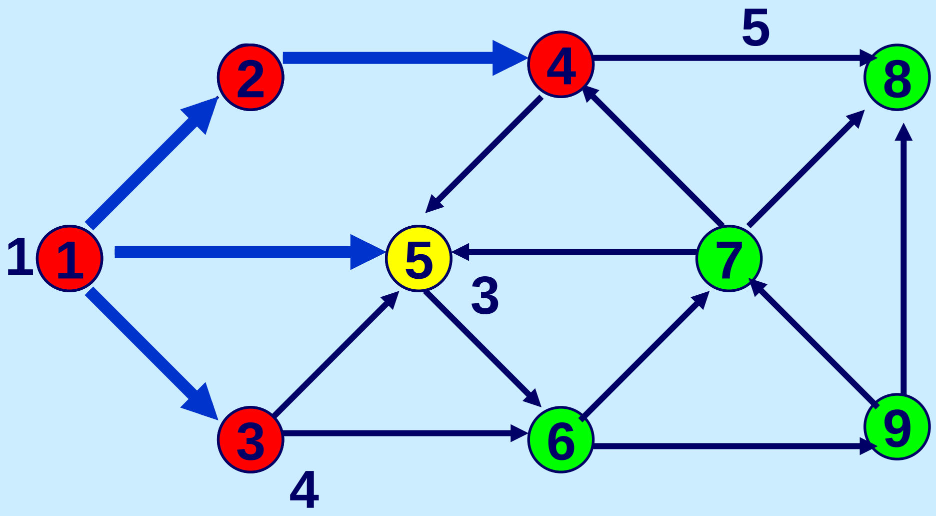
Let d(j) be the fewest number of arcs on a path from s to j.

Suffices to prove a 4th invariant:

4. If d(i) < d(j), then i is marked before j.

If the 4th invariant is true, then nodes are marked in order of increasing distance from node s.

Select node 5



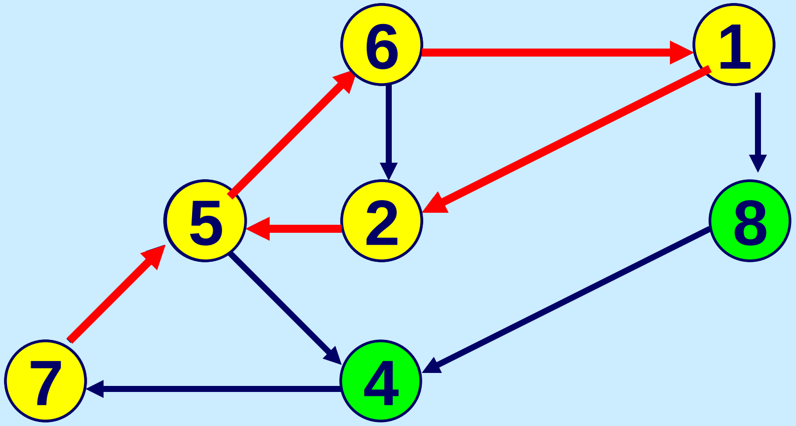
LIST

5	3	4				

4. If d(i) < d(j), then i is marked before j. (Note that no unmarked node has a distance that is less than d(4).)

Preliminary to Topological Sorting

LEMMA. If each node has at least one arc going out, then the first inadmissible arc of a depth first search determines a directed cycle.



COROLLARY 1. If G has no directed cycle, then there is a node in G with no arcs going out. Similarly, there is at least one node in G with no arcs coming in.

COROLLARY 2. If G has no directed cycle, then one can relabel the nodes so that for each arc (i,j), i < j.

Topological ordering

Animation

```
INITIALIZE as follows:
    for all i ∈ N do indegree(i) := 0;
    for all (i,j) ∈ A do indegree(j) := indegree(j) + 1;
    LIST := ∅;
    next := 0;
    for all i ∈ N do if indegree(i) = 0, then LIST := LIST ∪ { i };
while LIST ≠ ∅ do
  select a node i from LIST and delete it from LIST;
  next := next + 1;
  order(i) := next;
  for all (i,j) ∈ A(i) do
       indegree(j) := indegree(j) – 1;
       if indegree(j) = 0 then LIST := LIST ∪ { j };
if  next < n then the network contains a directed cycle
else the network is acyclic and the order is topological
```

Invariant For Topological Sorting

A node is called marked when it receives an order.

INVARIANT (at the beginning of the while loop)

1. indegree(i) is the number of arcs directed to i from nodes that are not marked.

Thus if j is on LIST, then there are no arcs into node j from unmarked nodes.

If the algorithm ends before labeling all nodes, then there is a directed cycle in the unmarked nodes.

Every unmarked node has at least one incoming arc, and so there is a directed cycle.

More on Topological Sorting

Runs in O(n+m) time.

Useful starting point for many algorithms that involve acyclic graphs.

Summary on Graph Search

- Graph Search
  - Finds all nodes reachable from s in O(m) time.
  - Determine the connected components of an undirected graph.
  - Breadth first search
  - Depth first search

Algorithm Validation (proofs of correctness).

- Prove invariants using induction
- Establish things that change

Summary on Graph Search

- **Topological sort (or order);**
  - **Running time is O(n+m) using simple data structures and algorithms.**
  - **Very important for preprocessing.**

15.082J / 6.855J / ESD.78J Network Optimization
Fall 2010