

Objectives

- Graph Connectivity, Traversal
- BFS & DFS Implementations, Analysis

Jan 23, 2013

CSCI211 - Sprenkle

1

Review

- What is a graph?
- How do we refer to a graph's number of nodes/vertices and number of edges?
- How can we represent a graph?

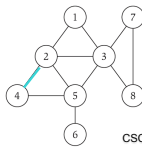
Jan 23, 2013

CSCI211 - Sprenkle

2

Graph Representation: Adjacency Matrix

- $n \times n$ matrix with $A_{uv} = 1$ if (u, v) is an edge
 - Two representations of each edge (symmetric matrix)
 - Space: $\Theta(n^2)$
 - Checking if (u, v) is an edge: $\Theta(1)$ time
 - Identifying all edges: $\Theta(n^2)$ time



Jan 23, 2013

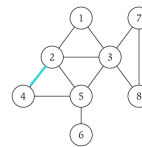
CSCI211 - Sprenkle

	1	2	3	4	5	6	7	8
1	0	1	1	0	0	0	0	0
2	1	0	1	1	1	0	0	0
3	1	1	0	0	1	0	1	1
4	0	1	0	0	1	0	0	0
5	0	1	1	1	0	1	0	0
6	0	0	0	0	1	0	0	0
7	0	0	1	0	0	0	1	1
8	0	0	1	0	0	0	1	0

3

Graph Representation: Adjacency List

- Node indexed array of lists
 - Two representations of each edge
 - Space? ← What are the extremes?
 - Checking if (u, v) is an edge?
 - Identifying all edges?



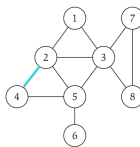
Jan 23, 2013

CSCI211 - Sprenkle

4

Graph Representation: Adjacency List

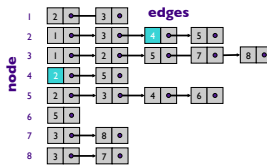
- Node indexed array of lists
 - Two representations of each edge
 - Space = $2m + n = O(m + n)$
 - Checking if (u, v) is an edge takes $O(\deg(u))$ time
 - Identifying all edges takes $\Theta(m + n)$ time



Jan 23, 2013

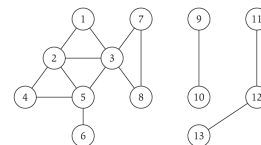
CSCI211 - Sprenkle

5



Paths and Connectivity

- Def. A **path** in an undirected graph $G = (V, E)$ is a sequence P of nodes $v_1, v_2, \dots, v_{k-1}, v_k$
 - Each consecutive pair v_i, v_{i+1} is joined by an edge in E
- Def. A path is **simple** if all nodes are *distinct*
- Def. An undirected graph is **connected** if \forall pair of nodes u and v , there is a path between u and v



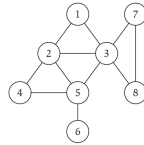
Jan 23, 2013

6

• Short path
• Distance

Cycles

- Def. A **cycle** is a path $v_1, v_2, \dots, v_{k-1}, v_k$ in which $v_1 = v_k$, $k > 2$, and the first $k-1$ nodes are all distinct



cycle $C = 1-2-4-5-3-1$

Jan 23, 2013

CSCI211 - Sprenkle

7

TREES

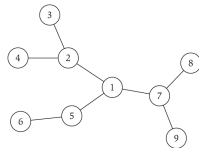
Jan 23, 2013

CSCI211 - Sprenkle

8

Trees

- Def. An undirected graph is a **tree** if it is connected and does not contain a cycle
- Simplest connected graph
 - Deleting any edge from a tree will disconnect it



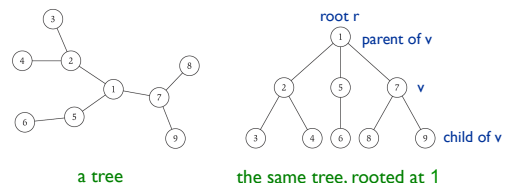
Jan 23, 2013

CSCI211 - Sprenkle

9

Rooted Trees

- Given a tree T , choose a root node r and orient each edge away from r
- Models hierarchical structure



a tree

the same tree, rooted at 1

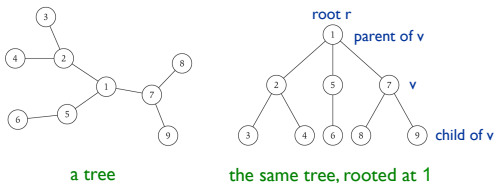
Why $n-1$ edges?

Jan 23, 2013

10

Rooted Trees

- Why $n-1$ edges?
 - Each non-root node has an edge to its parent



a tree

the same tree, rooted at 1

Jan 23, 2013

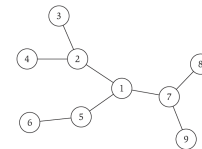
CSCI211 - Sprenkle

11

Trees

- Theorem. Let G be an undirected graph on n nodes. Any two of the following statements imply the third:

- G is connected
- G does not contain a cycle
- G has $n-1$ edges



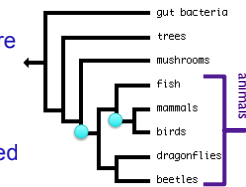
Jan 23, 2013

CSCI211 - Sprenkle

12

Phylogeny Trees

- Describe evolutionary history of species
 - mammals and birds share a common ancestor that they do not share with other species
 - all animals are descended from an ancestor not shared with mushrooms, trees, and bacteria



Tiffani Williams, Texas A&M
Computational Biology

Jan 23, 2013

CSCI211 - Sprenkle

13

GRAPH CONNECTIVITY & TRAVERSAL

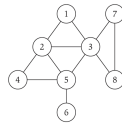
Jan 23, 2013

CSCI211 - Sprenkle

14

Connectivity

- s-t connectivity problem.** Given nodes s and t , is there a path between s and t ?
- s-t shortest path problem.** Given nodes s and t , what is the length of the shortest path between s and t ?
- Applications
 - Facebook
 - Maze traversal
 - Kevin Bacon number
 - Spidering the web
 - Fewest number of hops in a communication network



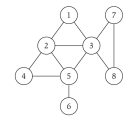
Jan 23, 2013

CSCI211 - Sprenkle

15

Application: Connected Component

- Find all nodes **reachable** from s



- Connected component containing node 1 is $\{1, 2, 3, 4, 5, 6, 7, 8\}$

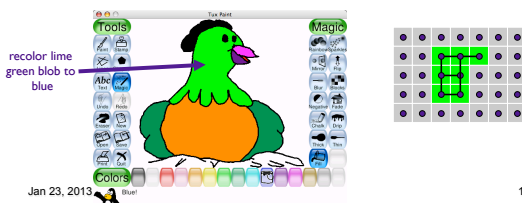
Jan 23, 2013

CSCI211 - Sprenkle

16

Application: Flood Fill

- Given lime green pixel in an image, change color of entire blob of neighboring lime pixels to blue
 - Node: pixel
 - Edge: two neighboring lime pixels
 - Blob: connected component of lime pixels

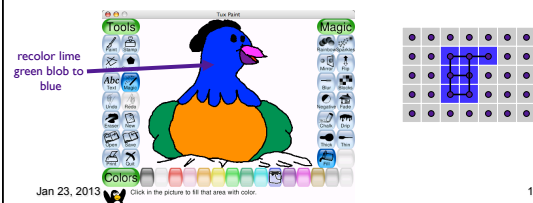


Jan 23, 2013

17

Application: Flood Fill

- Given lime green pixel in an image, change color of entire blob of neighboring lime pixels to blue
 - Node: pixel
 - Edge: two neighboring lime pixels
 - Blob: connected component of lime pixels

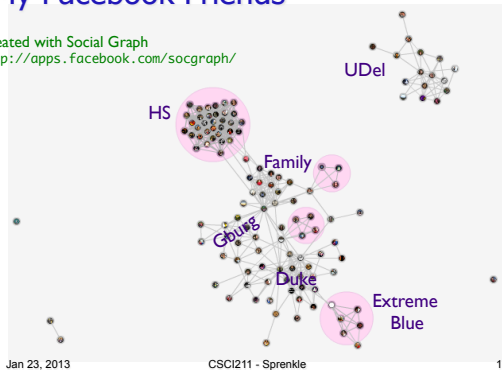


Jan 23, 2013

18

My Facebook Friends

Created with Social Graph
<http://apps.facebook.com/socgraph/>



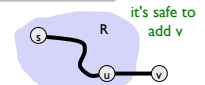
Jan 23, 2013

CSCI211 - Sprenkle

19

A General Algorithm

R will consist of nodes to which s has a path
 $R = \{s\}$
 while there is an edge (u,v) where $u \in R$ and $v \notin R$
 add v to R



- R will be the **connected component** containing s
- Algorithm is underspecified

In what order should we consider the edges?

Jan 23, 2013

CSCI211 - Sprenkle

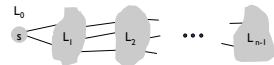
20

Breadth-First Search

- Intuition.** Explore outward from s in all possible directions (edges), adding nodes one "layer" at a time

- Algorithm**

- $L_0 = \{s\}$
- $L_1 =$ all neighbors of L_0
- $L_2 =$ all nodes that have an edge to a node in L_1 and do not belong to L_0 or L_1
- $L_{i+1} =$ all nodes that have an edge to a node in L_i and do not belong to an earlier layer



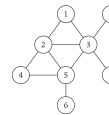
Jan 23, 2013

CSCI211 - Sprenkle

21

Run BFS on This Graph

$s = 1$



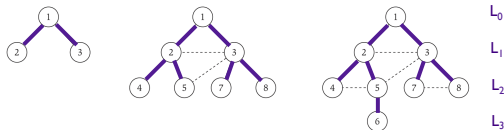
Jan 23, 2013

CSCI211 - Sprenkle

22

Example of Breadth-First Search

$s = 1$



Creates a tree
 -- is a node in the graph that is not in the tree

Jan 23, 2013

CSCI211 - Sprenkle

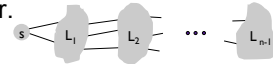
23

Breadth-First Search

- Theorem.**

For each i , L_i consists of all nodes at distance exactly i from s .

There is a path from s to t iff t appears in some layer.



- What does this theorem mean?
- Can we determine the distance between s and t ?

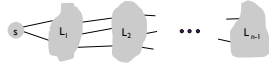
Jan 23, 2013

CSCI211 - Sprenkle

24

Breadth-First Search

- **Theorem.** For each i , L_i consists of all nodes at distance exactly i from s . There is a path from s to t iff t appears in some layer.
 - Shortest path to t from s , is the i from L_i
 - All nodes **reachable** from s are in L_1, L_2, \dots, L_{n-1}



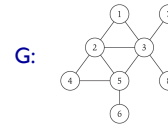
Jan 23, 2013

CSCI211 - Sprenkle

25

Breadth-First Search

- **Property.** Let T be a BFS tree of $G = (V, E)$, and let (x, y) be an edge of G . Then the level of x and y **differ** by **at most** 1.



If x is in L_i ,
then y must be in ???

Jan 23, 2013

CSCI211 - Sprenkle

26

Connected Component: BFS

- Find all nodes **reachable** from s

In general...

```
R will consist of nodes to which s has a path
R = {s}
while there is an edge (u,v) where u ∈ R and v ∉ R
  add v to R
```

In what order does BFS consider edges?

Jan 23, 2013

CSCI211 - Sprenkle

27

Connected Component: BFS vs DFS

- Find all nodes **reachable** from s

In general...

```
R will consist of nodes to which s has a path
R = {s}
while there is an edge (u,v) where u ∈ R and v ∉ R
  add v to R
```

- **Theorem.** Upon termination, R is the connected component containing s
 - BFS = explore in order of distance from s
 - DFS = explore until hit "deadend"

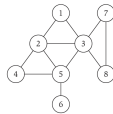
Jan 23, 2013

CSCI211 - Sprenkle

28

Depth-First Search

- Need to keep track of where you've been
- When reach a "dead-end" (already explored all neighbors), backtrack to node with unexplored neighbor
- **Algorithm:**



```
DFS(u):
  Mark u as "Explored" and add u to R
  For each edge (u, v) incident to u
    If v is not marked "Explored" then
      DFS(v)
```

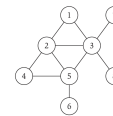
Jan 23, 2013

CSCI211 - Sprenkle

29

Depth-First Search

- How does DFS work on this graph?
 - Starting from node 1



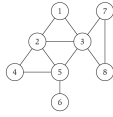
Jan 23, 2013

CSCI211 - Sprenkle

30

DFS vs BFS

- Compare the resulting trees



Jan 23, 2013

CSCI211 - Sprenkle

31

DFS Analysis

- Let T be a depth-first search tree, let x and y be nodes in T , and let (x, y) be an edge of G that is not an edge of T . Then one of x or y is an ancestor of the other in T .

“equivalent” of BFS: connected nodes are at most one layer apart

Jan 23, 2013

CSCI211 - Sprenkle

32

DFS Analysis

- Let T be a depth-first search tree, let x and y be nodes in T , and let (x, y) be an edge of G that is not an edge of T . Then one of x or y is an ancestor of the other in T .
- Proof.**
 - Suppose that $x-y$ is an edge in G but not in T . (From problem statement)
 - WLOG, assume that DFS reaches x before y
 - When edge $x-y$ is considered in the DFS algorithm, we don't add it to T (from problem statement), which means that y must have been explored.
 - But, since we reached x first, y had to be discovered between invocation and end of the recursive call $\text{DFS}(x)$
 - i.e., y is a descendent of x

Jan 23, 2013

CSCI211 - Sprenkle

33

Notes on Assignments

- Designing algorithms
 - Be as descriptive as possible, provide intuition
 - Explain running time
 - Match prescribed running time
 - Or what you think the running time is

Jan 23, 2013

CSCI211 - Sprenkle

34

Problem Set #1

- Follow template for induction hypothesis
- $\sqrt{2}n < n + 10$
- $n^2 \log n < n^{2.5}$
 - $\log n < n^{.5}$ (divide by n^2)
 - $\log \log n < .5 * \log n$ (take log of each)
- Similar to solved problem in Chapter 2

Jan 23, 2013

CSCI211 - Sprenkle

35

Looking Ahead

- Friday: Problem Set 2 due
- Continue reading Chapter 3

Jan 23, 2013

CSCI211 - Sprenkle

36