Graph Algorithms

• Another name for the lecture is “Google I”. We’re going to look at several critical ideas in algorithm design and use the example of Google to motivate them.

• Our question: How does Google find stuff?

The Internet (a piece)

athos.rutgers.edu  porthos.rutgers.edu  dir.yahoo.com
www.cs.rutgers.edu  www.dcis.rutgers.edu
paul.rutgers.edu  google.com  patmedia.net

(domain) name
computer
network
A Conversation

porthos.rutgers.edu
dir.yahoo.com
To: porthos.rutgers.edu
What gives?
To: dir.yahoo.com
I’m told you have a web page called "Science/Computer_Science/College_and_University_Departments/?b=20". Can you send me a copy?
To: porthos.rutgers.edu
Sure:

http://dir.yahoo.com/Science/Computer_Science/College_and_University_Departments/?b=20

- Georgia Institute of Technology (Georgia Tech) - College of Computing
  dir.yahoo.com/.../Departments_and_Programs/College_of_Computing
- Swiss Federal Institute of Technology - Eidgenossische Technische Hochschule Zurich (ETH)
  www.inf.ethz.ch
- University of California, Irvine - School of Information and Computer Science
  dir.yahoo.com/.../Departments_and_Programs/School_of_Information_and_Computer_Science
Web Search

- All web browsing consists of requests for specific pages.
- But, what happens if we don’t know what we want?
- A “search engine” is just a web server that can respond to a particular request for web pages.

Another Conversation

To: porthos.rutgers.edu
Whazzup?

To: google.com
Can I have a copy of the page "search?hl=en&q=rutgers+computer+science&btnG=Google+Search"?

To: porthos.rutgers.edu
Sure:
Indirection

- Porthos asked google where it could find pages about “rutgers computer science”.
- Google responded with a page that included addresses of other pages.
- Porthos can now request those pages directly from the web servers that “host” (store and dispense) them.

How Does Google Know?

- So, somehow, Google has to put together a web page in response to any query, which includes a list of names of pages that contain those terms.
- But, how does it know which pages contain which terms?
- Theories?
  1.
  2.
  3.
An Experiment
http://www.cs.rutgers.edu/~mlittman/courses/cs442-06/googletest1.html

How Does Google Find Pages?

This page has little purpose other than to include the word "googlediscovery". This is a word (sort of) that I concocted on January 3, 2006 and verified that it was unknown to Google. (Did you mean: "google discovery"). This page has a direct link from the course homepage. Sure, it's no "truthiness", but it's still useful scientifically.

There is a secondary page, http://www.cs.rutgers.edu/~mlittman/courses/cs442-06/googletest2.html, that does not have an explicit link from anywhere. It has its own special term, which consists of concatenating "google" and "blackout".

I also made a third term, formed from "google" and "abyss", that I do not plan on putting on any page. (Did you mean: google?)

- 1/3/06 (8:01am): Saved this page.
- 1/3/06 (8:03am): "-discovery" (0), "-blackout" (0), "-abyss" (0).
- 1/4/06 (8:00am): "-discovery" (0), "-blackout" (0), "-abyss" (0).
- 1/5/06 (8:10am): "-discovery" (0), "-blackout" (0), "-abyss" (0).
- 1/6/06 (9:18am): "-discovery" (0), "-blackout" (0), "-abyss" (0).
- 1/7/06 (8:39am): "-discovery" (0), "-blackout" (0), "-abyss" (0).
- 1/8/06 (8:00am): "-discovery" (1), "-blackout" (0), "-abyss" (0).

What Do You Think Now?

- Google knew the word “googlediscovery” five days after I put up a web page with the word and linked it to the course web page.

- Google still didn’t know the word “googleblackout” more than a month later in spite of being on a similar (but unlinked) page at the same time.

- We need to understand how pages link to each other.
A Piece of the Web

1. www.cs.rutgers.edu/~mlittman/courses/cs442-06/: 2, 3, 1, 2, 4, 5
2. www.cs.rutgers.edu/~mlittman/: 1, 6, 7, 10
3. paul.rutgers.edu/~babes/: 1
4. www.cs.rutgers.edu/~mlittman/courses/cs442-06/python/
5. www.cs.rutgers.edu/~mlittman/courses/cs442-06/googletest1.html: 7, 1
6. www.cs.rutgers.edu/rl3/: 8, 10
7. www.cs.rutgers.edu/~mlittman/topics/googletest1.html: 7, 1
9. www.cs.rutgers.edu/~mlittman/courses/cs442-06/googletest2.html: 1

Pictorial Representation

rl3 people page

Michael’s homepage

my "google-whack" page

rl3: my lab homepage

my research papers page

our course webpage

Monica’s homepage

our course python page

googletest1 “google-discovery”

googletest2 “google-blackout”
Graphs

- In CS and discrete math, this kind of structure is known as a graph.
- Nodes: Web pages, in this case.
- Links: Pointer from one web page to another, in this case.

Some Graph Terms

- source: a node with no incoming links.
- sink: a node with no outgoing links.
- path: a list of nodes such that each adjacent pair of nodes has a link from the first to the second.
- cycle: a path in which the first and last node are the same.
- connected component: a set of nodes such that there is a path from any node to any other node in the set.
- tour: a cycle including all nodes in the graph.
**Graphs Are Everywhere**

- What are the nodes, links, paths, source, sinks, connected components of each?

- Two more definitions: A graph is **undirected** if each connected pair of nodes is connected in both directions.

- An undirected graph is a **tree** if it has no cycles.

- Is each example directed or undirected? Tree or not?

---

**Tournament**

[Diagram showing a tournament with characters and their connections, similar to March Madness.]
Rail Map

Circuit Diagram

DC motor direction controller

DIODES: 1N4001

CAUTION: The max motor current rating not to exceed max. SL5/SK100 rating 15A with heat sink.
A Maze

Sexual Contact Network
We can represent a graph in the computer by a list of nodes, and a function that, given a node \( i \), returns the list of nodes to which \( i \) is linked.

```python
g = [[6],[1,2,3,4,5],[0,1,6,7], [1], [],[1,7],[0,8],[],[2],[1]]

def links(i):
    global g
    print str(i) + " links to:",
    for k in g[i]:
        print " " + str(k)

>>> links(5)
5 links to:
 1
 7
```
checkPath

- Given a list of nodes, checks if it’s a path.

```python
# Is there a link from i to j?
def checkLink(i,j):
    global g
    for k in g[i]:
        if k == j: return True
    return False

# Does the list of links make a path?
def checkPath(l):
    global g
    for i in range(len(l)-1):
        if checkLink(l[i],l[i+1]) == False:
            return False
    return True
```

```python
>>> checkLink(1,2)
True
>>> checkLink(1,6)
False
>>> checkPath([1,6])
False
>>> checkPath([1,2,7])
True
>>> checkPath([1,2,7,8])
False
```

Reachable

- A node \( j \) is reachable from a node \( i \) if there is a path that begins at \( i \) and ends at \( j \).

- Let’s list all the nodes reachable from \( i \).

- Any node that is reachable from a node that \( i \) is linked to is also reachable.

```python
def reachable(i):
    global g
    print str(i) + " is reachable"
    for j in g[i]:
        reachable(j)

>>> reachable(4)
4 is reachable
6 is reachable
0 is reachable
6 is reachable
0 is reachable
6 is reachable
0 is reachable
```
• What goes wrong? Once we realize we can reach some node, we should mark it as “reached” and never pursue it again.

Reachable Version 2

```python
reached = []
def reachable(i):
    global g, reached
    reached = list(range(len(g)))
    for j in range(len(g)):
        reached[j] = False
    reachable_recursive(i)

def reachable_recursive(i):
    global g, reached
    if reached[i] == False:
        print(str(i) + " is reachable")
        reached[i] = True
    for j in g[i]:
        reachable_recursive(j)
```

```python
>>> reachable(6)
6 is reachable
0 is reachable
8 is reachable
2 is reachable
1 is reachable
3 is reachable
4 is reachable
5 is reachable
7 is reachable
>>> reachable(4)
4 is reachable
```
A Mazing Example

\[ g = [[1,2],[0,3],[0,4],[1,5],[2,6],[3,7],[5,7],[5,6]] \]

```python
>>> reachable(0)
0 is reachable
1 is reachable
3 is reachable
5 is reachable
7 is reachable
6 is reachable
2 is reachable
4 is reachable
```

Depth First Search

- The algorithm in "reachable" is sometimes called DFS, because it decides which way to explore and keeps going until it hits a dead end.
- Sometimes, it’s better to go “breadth first” in that we check nearby nodes before pursuing farther ones.
# BFS Algorithm

```python
def bfs(i):
    global g
    reached = range(len(g))
    for j in range(len(g)):
        reached[j] = False
    todoList = [i]
    steps = 0
    while todoList != []:
        doNext = []
        for j in todoList:
            if reached[j] == False:
                print(j, "reached in", steps, "steps")
                doNext = doNext + g[j]
                reached[j] = True
        todoList = doNext
        steps = steps + 1
```

```python
>>> bfs(0)
0 reached in 0 steps
1 reached in 1 steps
2 reached in 1 steps
3 reached in 2 steps
4 reached in 2 steps
5 reached in 3 steps
6 reached in 3 steps
7 reached in 4 steps
```

---

## BFS: Shortest Path

- Breadth-first search finds nodes in shortest-path order.
- That is, if BFS finds a node \( j \) in 5 steps, there is no 4, 3, 2, or 1-step path to \( j \).
BFS/DFS Comparison

• BFS and DFS are two algorithms for finding all the nodes in a graph reachable from a given starting node.

• Which would you prefer if the graph has no cycles? Why?

• Which would you prefer if the graph has a huge (infinite?) number of nodes? Why?

Back to Google

• So, how does Google do it?

  I. Web crawl: download known pages, collect links to other pages, repeat

  II. Indexing: Build a giant index that associates each word with a list of pages on which it appears.

  III. Distributed search: Use lots and lots and lots of computers to do fast lookups.
Next Time

- Google II: Sorting (for indexing).
- Read sorting section from Hillis.

“Understanding” a Maze?

We can “see” a path through the maze, but what happened to step 2?

Step 1: Look at the maze
Step 2: ?
Step 3: Profit!
Maybe Step 2 is Important

Sometimes it is hard to see a path right away. I'm lost :(

(One) Understanding of Maze

data representation (a graph!)
A Special Graph Type

One special kind of graph is called a tree. What makes a tree different is that not all edges mean the same thing.

As many (or few) children as we want.

A Tree Example

Why are some nodes different colors?
What is the red node?
What are the green nodes?

leaf nodes, or the “fringe”
Why Do We Care About Trees?

Finding the path from any node to the root is very easy.

Simply follow the parent edges.

Trees For Solving a Maze

Let's build a tree, using this maze.

We found the exit!
Now how does this tree give us a path?
Let's build a tree, using this maze.

We found the exit!
Now how does this tree give us a path?
Follow the parent edges to the root.
Now, We Have Step 2

Thanks, BFS!