The Structure of the Web

CMSC 498J: Social Media Computing

Department of Computer Science
University of Maryland
Spring 2016

Hadi Amiri
hadi@umd.edu
Announcement

• **HW2 is out**
  ▫ **Due Date: 04/05/2016 11:00AM**
Lecture Topics

• Information Networks
• Web Structure
• Finding SCCs in Directed Graphs
Information Networks

• Information Network
  ▫ Nodes are pieces of information and Edges join the related ones!

• Examples of information networks?
  ▫ The Web
  ▫ Citation networks
  ▫ Encyclopedia References
  ▫ Wireless communication
  ▫ Etc.
Information Networks- Cnt.

- Sample Citation Net

Strength of week ties
Triadic closure
Small-world phenomenon
Structural balance
Homophily
Information Networks- Cnt.

- The Web is arguably the most prominent example of information networks.
  - Let’s take a closer look at it.
The World Wide Web

- Created by Tim Berners-Lee during 1989-1991 in CERN:
  - CERN is in Geneva, Switzerland

Q: Did you invent the internet?

A:

No, no, no!

When I was doing the WWW, most of the bits I needed were already done.

Vint Cerf and people he worked with had figured out the Internet Protocol, and also the Transmission Control Protocol.

Paul Mockapetris and friends had figured out the Domain Name System.

People had already used TCP/IP and DNS to make email, and other cool things. So I could email other people who maybe would like to help work on making the WWW.

I didn't invent the hypertext link either. The idea of jumping from one document to another had been thought about lots of people, including Vannevar Bush in 1945, and by Ted Nelson (who actually invented the word hypertext). Bush did it before computers really existed. Ted thought of a system but didn't use the internet. Doug Engelbart in the 1960's made a great system just like WWW except that it just ran on one [big] computer, as the internet hadn't been invented yet. Lots of hypertext systems had been made which just worked on one computer, and didn't link all the way across the world.

I just had to take the hypertext idea and connect it to the TCP and DNS ideas and -- ta-dal -- the World Wide Web.
The World Wide Web - Cnt.

- Read some history at
  - 40 maps: http://www.vox.com/a/internet-maps

Source: http://www.vox.com/a/internet-maps
The Web as a Graph

• Let’s say we have a set of Web pages

  I teach a class on Networks.

  Networks Course: We have a class blog

  Networks Class Blog: This blog post is about Microsoft

• How can we organize this information?
The Web as a Graph - Cnt.

• How can we organize this information?
  ▫ According to a classification system
    • books in a library
  ▫ As a series of folders
    • files on our computers
  ▫ Even purely alphabetically!
    • terms in an index
    • names in a phone directory

• Any of these organizational systems could have been used for the Web!
The Web as a Graph- Cnt.

• How can we organize this information?
  ▫ traditional methods for storing information in a book, a library, or a computer memory are **linear**:
    • Consist of a collection of items sorted in some sequential order.
The Web as a Graph- Cnt.

• How can we organize this information?
  ▫ Memex (1945): A more complex way of organizing information: **Associative Memory**
    • Just like what a semantic network represents!
    • you think of one thing; it reminds you of another

A device that provides an “enlarged intimate supplement to one's memory” by storing books, records, and communications. A desk with screen, keyboard, and microfilms!

Memex (1945), "As We May Think", Vannevar Bush in The Atlantic Monthly article!
Viewing the Web as a graph allows us to:

- better understand the **logical relationships** in its links,
- break its structure into **smaller, cohesive units**, and
- **identify important pages** as a step in organizing the results of Web searches.
The Web as a Graph - Cnt.

- I'm a student at Univ. of X
- My song lyrics
- Classes
- Networks
- Networks class blog
- Blog post about Company Z
- Blog post about college rankings
- I teach at Univ. of X
- Company Z's home page
- Our Founders
- Press Releases
- Contact Us
- USNews College Rankings
- USNews Featured Colleges
- I'm applying to college

**Node**

**Hyperlink**
The Web as a Graph - SCC
Web Structure

• How does the Web look like?
  ▫ Broder et al., Graph structure in the Web. WWW 2000:
    • Altavista data
    • Crawl from October, 1999 containing
      • 203 million URLs
      • 1,466 million links.
Web Structure- Cnt.

- Running BFS starting from a random node
  - undirected.

Running BFS starting from a random node
  - follows in-links, and out-links.

Web Structure- Cnt.

- Distribution of WCCs and SCCs on the web.

Web Structure- Cnt.

• **Findings:**
  
  - The Web contains a giant SCC.
    
    • If there were 2 giant SCCs, X and Y
    
    • a single link from any node in X to any node Y, and another link from any node in Y to any node in X is enough to merge X and Y to become part of a single SCC.
  
  - The Web has a Bow-Tie Structure!

IN nodes: can reach SCC but cannot be reached from it.
OUT nodes: can be reached from SCC but cannot reach it.
Tendrils nodes: (a) reachable from IN but cannot reach SCC, (b) can reach OUT but cannot be reached from SCC.
Tendrils nodes satisfying both a & b, travel in tube from IN to OUT without touching SCC.
Disconnected nodes: have no path to SCC ignoring directions

The Web structure is relatively stable despite the fact that the constituent pieces of the bow-tie are constantly shifting their boundaries, with nodes entering (and also leaving) the giant SCC over time.

Bow-tie structure provides a global view of the Web, but it doesn’t provide insight into patterns of connections within the parts.

Finding SCCs

- Given digraph \( G = (V, E) \), a SCC of \( G \) is a maximal set of nodes \( C \) subset of \( V \), such that for all \( u, v \) in \( C \), both \( u \) and \( v \) are reachabe from each other.

If \( G \) has an edge from some node in \( C_i \) to some node in \( C_j \) where \( i \neq j \), then one can reach any node in \( C_j \) from any node in \( C_i \) but not return! For example, one can reach any node in \( C_2 \) from any node in \( C_1 \) but cannot return to \( C_1 \) from \( C_2 \).
Finding SCCs- Cnt.

• Need to know 2 concepts to find SCCs in G.
  1. Transpose Graph
  2. Component Graph

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs- Cnt.

Transpose Graph \((G^T)\)

- The transpose of a given graph \(G\) is defined as:
  - \(G^T = (V, E^T)\), where \(E^T = \{(u, v): (v, u) \text{ in } E\}\).
  - \(G^T\) is \(G\) with all edges reversed.
  - Given \(G\), one can create \(G^T\) in linear time, i.e., \(\Theta(|V| + |E|)\), using adjacency lists.

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs- Cnt.

**Transpose Graph (Gᵀ)**

- Claim: G and Gᵀ have the same SCCs.
  - Meaning that nodes u and v are reachable from each other in G if and only if reachable from each other in Gᵀ.
- How to prove it?

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs- Cnt.

Component Graph ($G^{SCC}$)

- Graph with SCCs as nodes
- $G^{SCC} = (V^{SCC}, E^{SCC})$, where $V^{SCC}$ has one node for each SCC in $G$ and $E^{SCC}$ has an edge if there's an edge between the corresponding SCC's in $G$.

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs- Cnt.

Component Graph ($G^{SCC}$)

• **Claim:** $G^{SCC}$ is a DAG!
  ▫ Let $C$ and $C'$ be distinct SCC's in $G$,
  ▫ Let $u, v \in C$,
  ▫ Let $u', v' \in C'$,
  ▫ Suppose there is a path $u \rightarrow u'$ in $G$. Then there cannot also be a path $v' \rightarrow v$ in $G$.

• **Proof:**
  ▫ Suppose there is a path $v' \rightarrow v$ in $G$. Then
    • both $u \rightarrow u' \rightarrow v'$ and $v' \rightarrow v \rightarrow u$ are in $G$.
    • Therefore, $u$ and $v'$ are reachable from each other, so they cannot be in separate SCC's. #

Source: http://www.personal.kent.edu/~rmuhemma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs - Cnt.

**Algorithm**

1. Call **DFS**(G) to compute finishing times $f[u]$ for all $u$
2. Compute $G^T$
3. Call **DFS**(G$^T$) while considering nodes in order of decreasing $f[.]$ (as computed in **DFS**(G))
4. Output nodes in each tree of the depth-first forest formed in **DFS**(G$^T$) as a separate SCC.

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs - Cnt.

1. Call DFS(G)

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs - Cnt.

2. Compute $G^T$

Source: http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm
Finding SCCs- Cnt.

3. Call DFS($G^T$) considering nodes in order to decreasing $f[.]$.

4. Output nodes in each tree formed in DFS($G^T$) as a separate SCC.
   \{a, b, e\}, \{c, d\}, \{f, g\}, and \{h\}
Questions?
Reading

• Ch.13 The Structure of the Web [NCM]
• Strongly Connected Components
  ▫ http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/strongComponent.htm