Graph Theory and Complex Networks: An Introduction

Maarten van Steen

VU Amsterdam, Dept. Computer Science
Room R4.20, steen@cs.vu.nl

Chapter 08: Computer networks

Version: March 3, 2011

vrije Universiteit amsterdam

Contents

Chapter	**Description**
01: Introduction	History, background
02: Foundations	Basic terminology and properties of graphs
03: Extensions	Directed & weighted graphs, colorings
04: Network traversal	Walking through graphs (cf. traveling)
05: Trees	Graphs without cycles; routing algorithms
06: Network analysis	Basic metrics for analyzing large graphs
07: Random networks	Introduction modeling real-world networks
08: Computer networks	The Internet & WWW seen as a huge graph
09: Social networks	Communities seen as graphs

Introduction

Observation

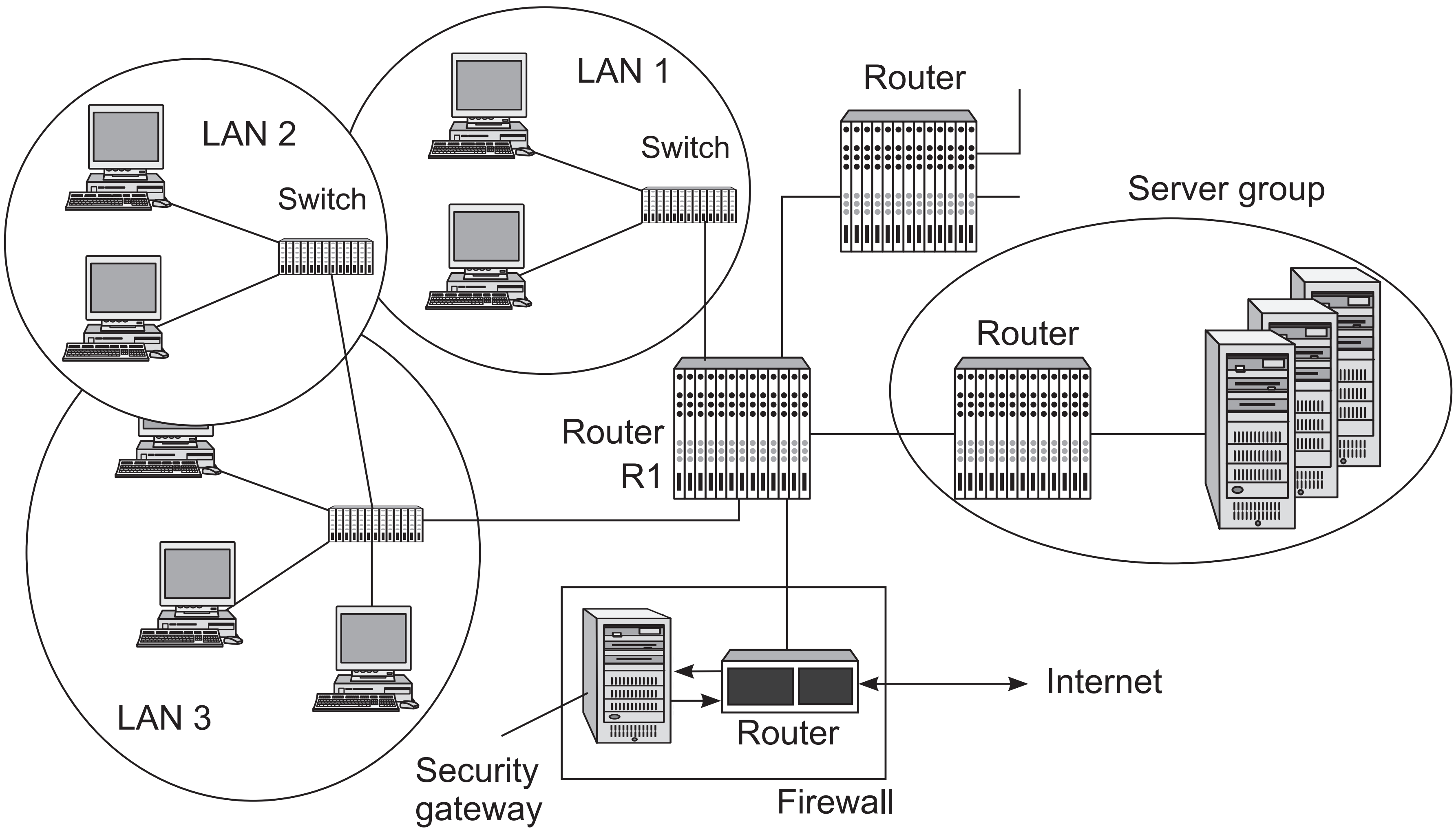
The Internet as we know it today is a communication network that allows us to exchange messages. The (World Wide) Web is a huge distributed information system, implemented on top of the Internet. The two are very different.

1. The organization and structure of the Internet
2. The organization of overlay (i.e., peer-to-peer) networks
3. The organization and structure of the Web

Computer networks: basics

- There are **many** different kinds of computer networks:
  - Traditional networks in buildings and on campus
  - Home networks (wired and wireless)
  - Networks for mobile phones
  - Access networks (with so-called hot spots)
  - Networks owned by Internet Service Providers (ISPs)
  - ...
- The Internet ties all these networks together (well, that's what we think).
- For starters: make distinction between small-area networks and large-area networks.

Small-area networks



Example: router

Example: switch

Example: security gateway

Example: server group

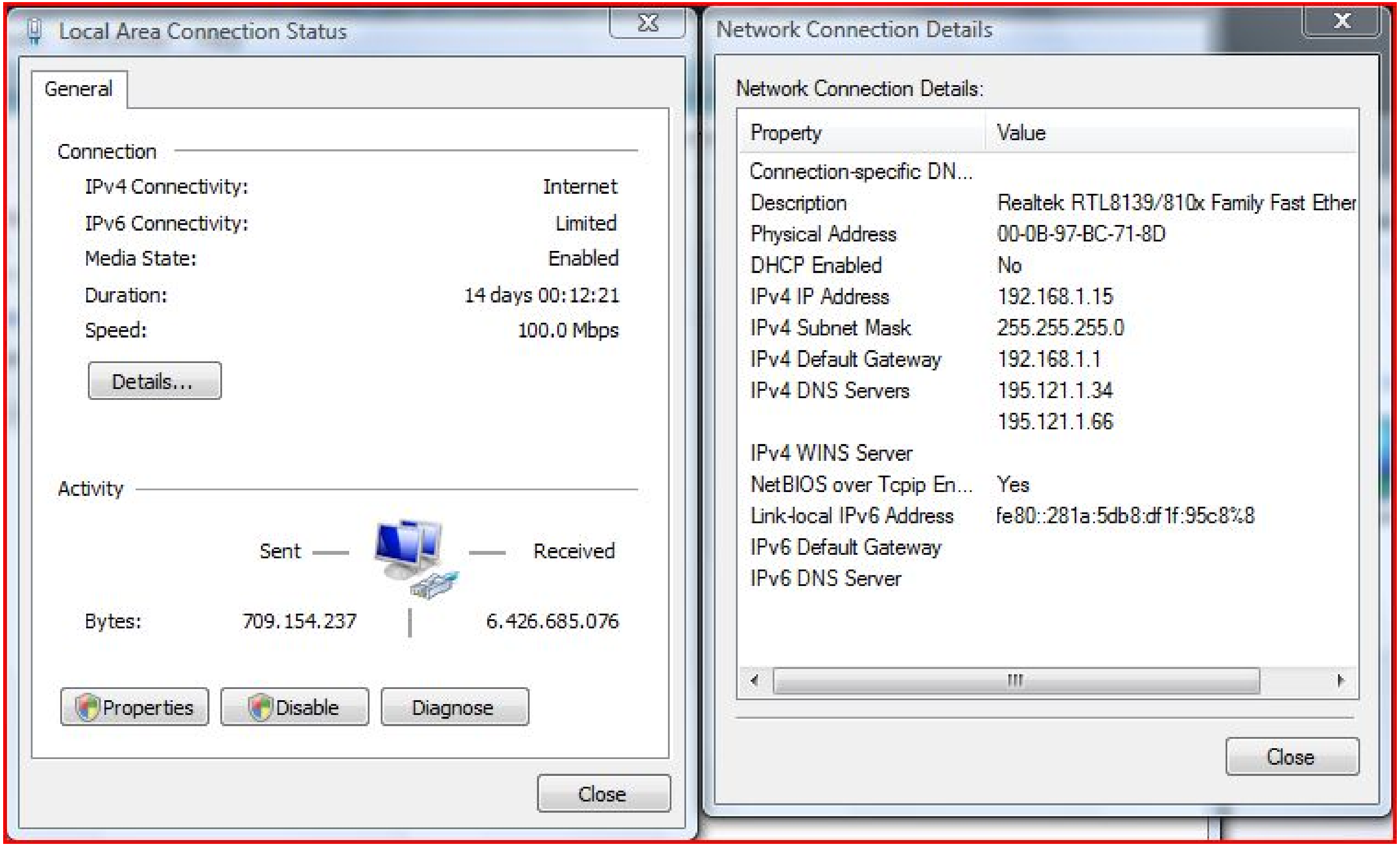
Addressing

Essence

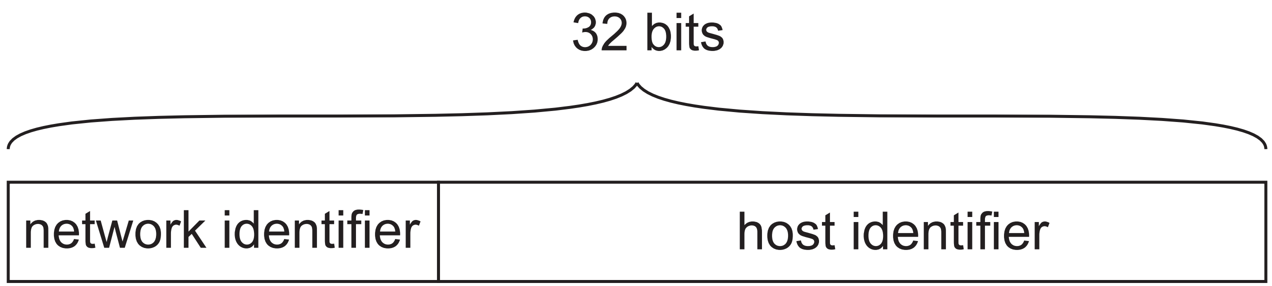
Each networked device has (in principle) a worldwide unique **low-level address**, also called its **MAC address**. The MAC address is nothing but a **device identifier**.

- When a device transmits a message, it always sends its MAC address as part of the message.
- A **switch** can connect several devices, and **discovers** the MAC addresses.
- When a MAC address has been discovered, a switch can distinctively **forward messages** to the associated device.

Assigning an Internet address



Structure of an IP address



- An IP address consists of a **network identifier** and a **host identifier**

  Network ID: worldwide unique address of a (small area) network to which messages can be **routed**

  Host ID: network-wide unique address associated with a device/host

- In the Internet, messages are always routed to a network. Internal routers handle subsequent forwarding to the hosts/devices using host IDs

IP addresses and home networks

Observation

Each home (or small organization) is assigned exactly one IP address.

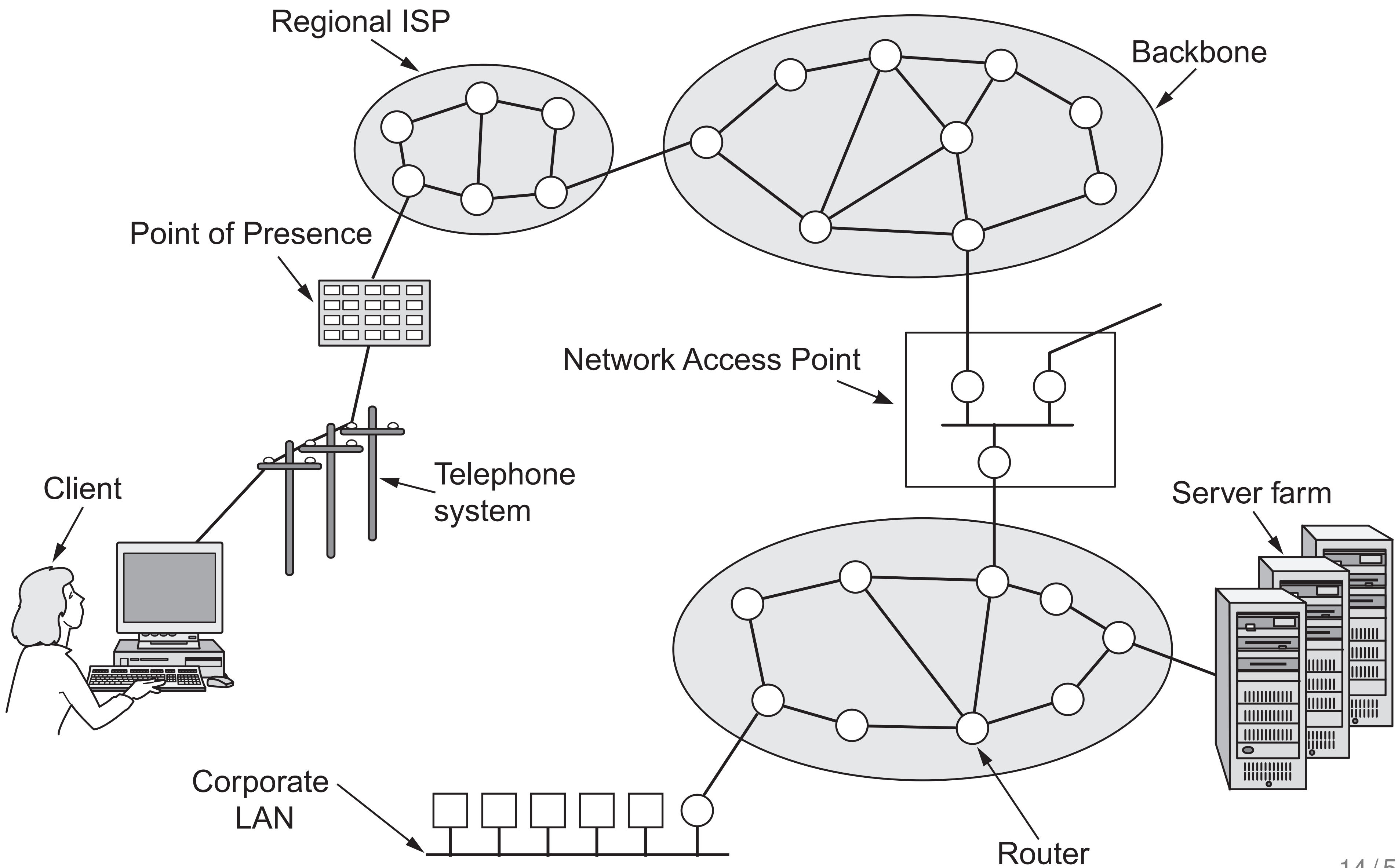
Note

Using a bag of tricks, we can **share** that address among different devices. For now, it is important to know that all your devices at home have (essentially) the same **external** IP address.

Consequence

All devices in a home network are seen **by the outside world** as being one and the same.

Large-area networks



Autonomous system

Description

An autonomous system is an organizational unit that maintains a collection of (interconnected) communication networks. An AS announces its **accessible** networks as $\langle$*AS number*, *network identifier*$\rangle$ pairs.

A simple number...

In 2009, there were approximately 25,000 ASes.

Measuring the AS topology

- Each AS i has a number of **border gateways**: a special router that can transfer messages between AS i and an AS to which that router is **linked**.
- If BG_1^i of AS i is linked to BG_1^j of AS $j \Rightarrow$ there is a physical connection between the two routers.
- Two gateways BG_1^i and BG_2^i of the same AS i, are always **internally linked**: they know how to reach each other through a communication path.
- A border gateway BG_1^i of AS i, attached to network n_i, announces $\langle i, n_i \rangle$ to its neighboring gateways.
- Assume BG_1^j of AS j is linked to $BG_1^i \Rightarrow BG_1^j$ can then announce that it knows a path to n_i: $\langle j, i, n_i \rangle$.
- Each other gateway BG_2^j of AS j can announce $\langle j, i, n_i \rangle$ to *its* linked neighbors.

Measuring the AS topology

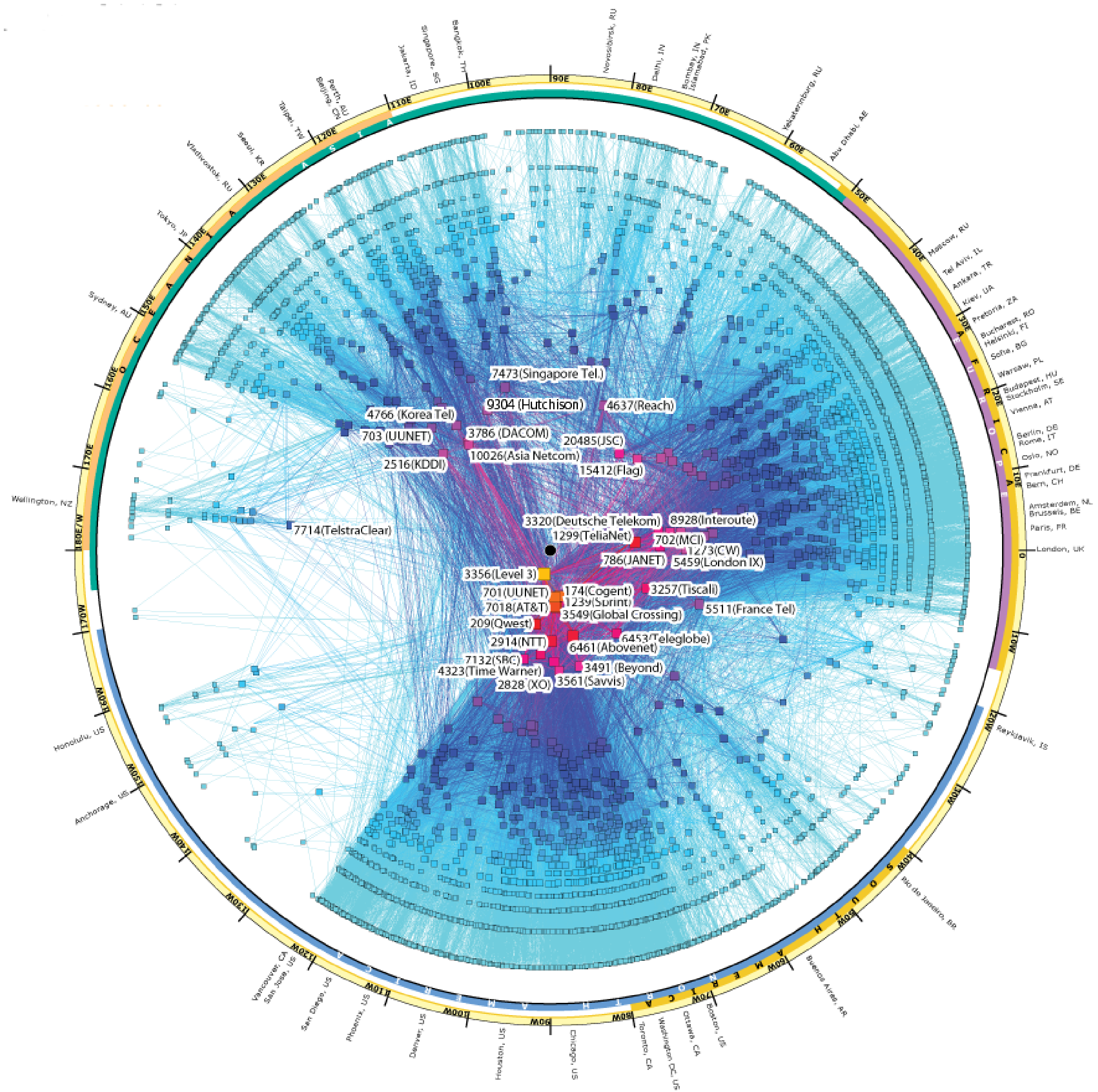
Important observations

- Gateways store and announce **entire paths** to destinations.
- For proper routing, each gateway needs to store paths to **every network in the Internet**.

Conclusion

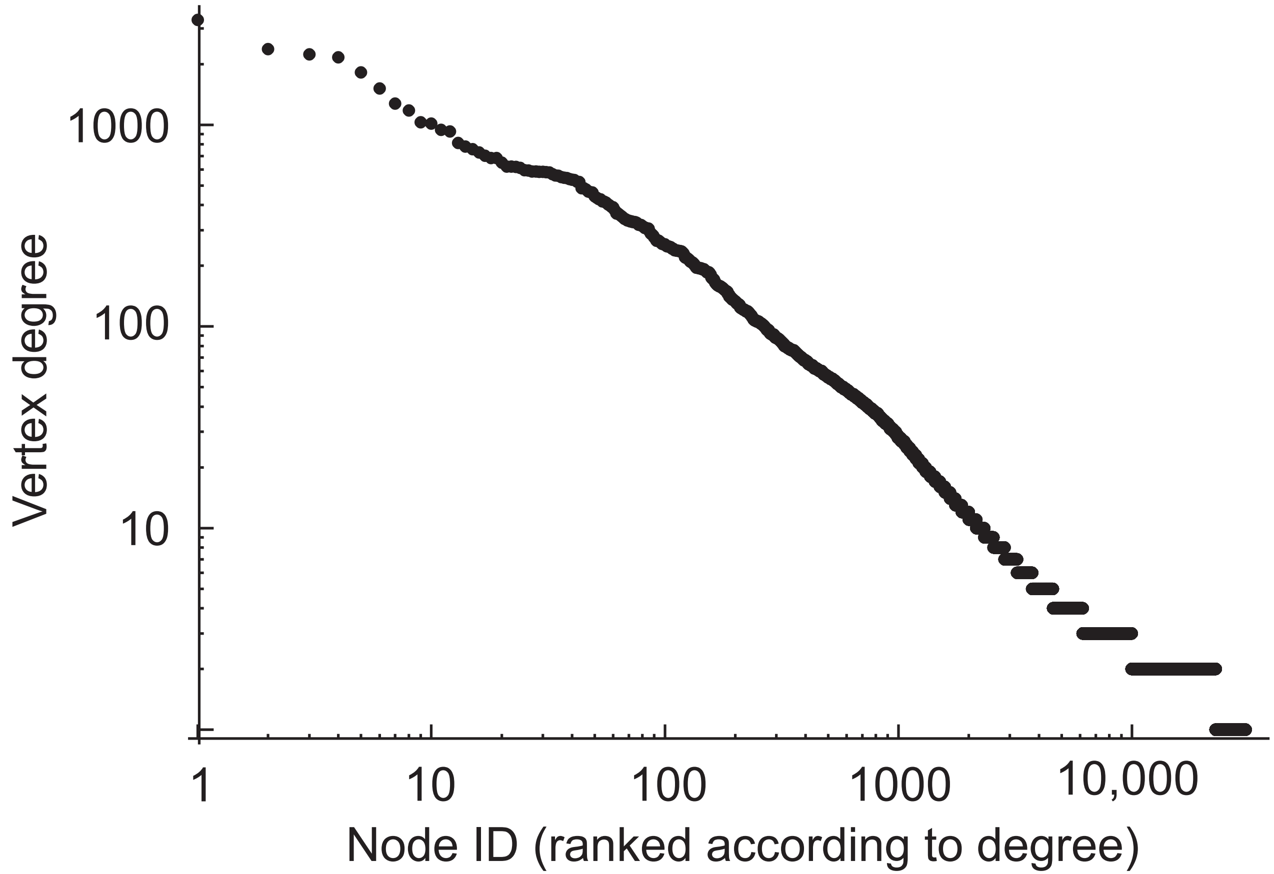
If we read the routing tables from only a few gateways, we should be able to obtain a reasonable complete picture of the **AS topology of the Internet**.

Visualizing the AS topology (CAIDA)



Example topology: October 2008

- Over 30,000 registered autonomous systems (including “double” registeries).
- Over 100,000 edges. **Note:** we may be missing more than 30% of all existing links!



Example topology: October 2008

Rank:	1	2	3	4	5	6	7	8	9	10
Degree:	3309	2371	2232	2162	1816	1512	1273	1180	1029	1012

Some observations

- Very high clustering coefficient for top-1000 hubs: an almost complete graph!
- Most paths no longer than 3 or 4 hops.
- Most ASes separated by shortest path of max. length 6.

Peer-to-peer overlay networks

Issue

Large-scale **distributed computer systems** are spread across the Internet, yet their constituents need to communicate directly with each other $\Rightarrow$ organize the system in an **overlay network**.

Overlay network

Collection of **peers**, where each peer maintains a **partial view** of the system. View is nothing but a list of other peers with whom communication connections can be set up.

Observation

Partial views may change over time $\Rightarrow$ an ever-changing overlay network.

Structured overlay network: Chord

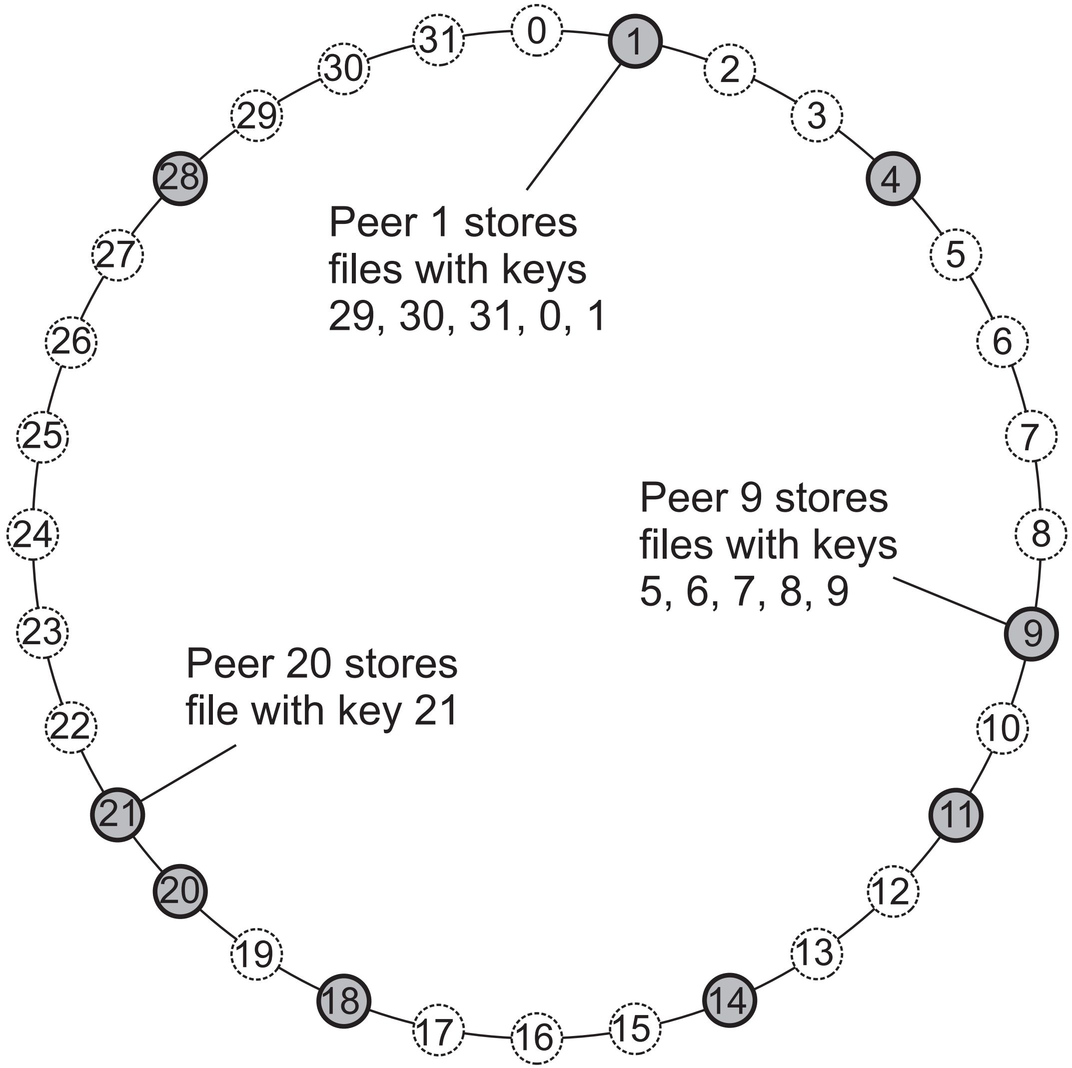
Basics

- Each peer is assigned a unique **m-bit identifier** id.
- Every peer is assumed to store data contained in a file.
- Each file has a unique **m-bit key** k.
- Peer with smallest identifier $id \geq k$ is responsible for storing file with key k.
- $succ(k)$: The peer (i.e., node) with the smallest identifier $p \geq k$.

Note

All arithmetic is done modulo $M = 2^m$. In other words, if $x = k \cdot M + y$, then $x \bmod M = y$.

Example



0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
Peer 1 stores
files with keys
29, 30, 31, 0, 1
Peer 9 stores
files with keys
5, 6, 7, 8, 9
Peer 20 stores
file with key 21

Efficient lookups

Partial view = finger table

- Each node p maintains a **finger table** $FT_p[]$ with at most m entries:

$$FT_p[i] = succ(p + 2^{i-1})$$

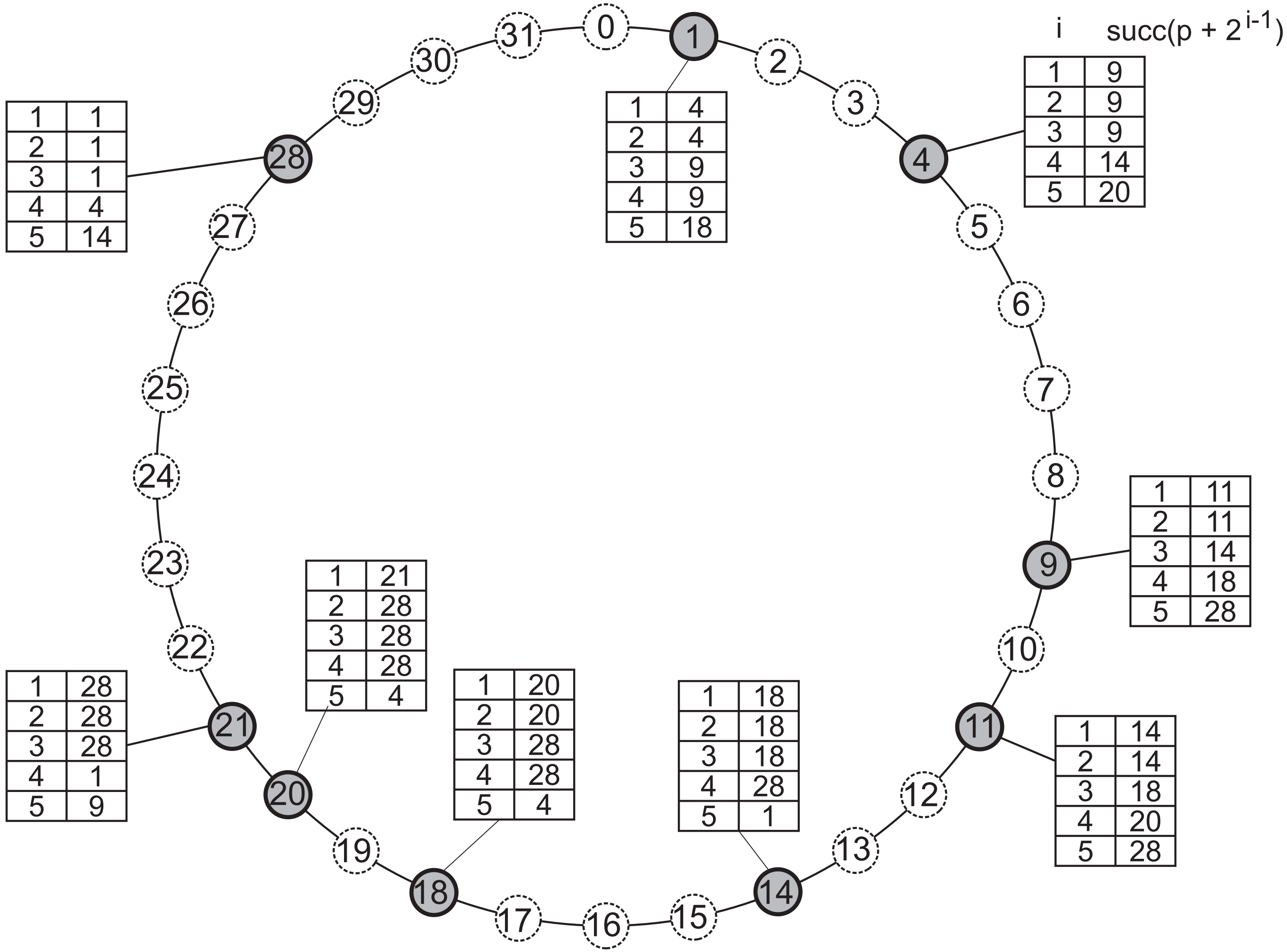
Note: $FT_p[i]$ points to the first node succeeding p by at least 2^{i-1}.

- To look up a key k, node p forwards the request to node with index j satisfying

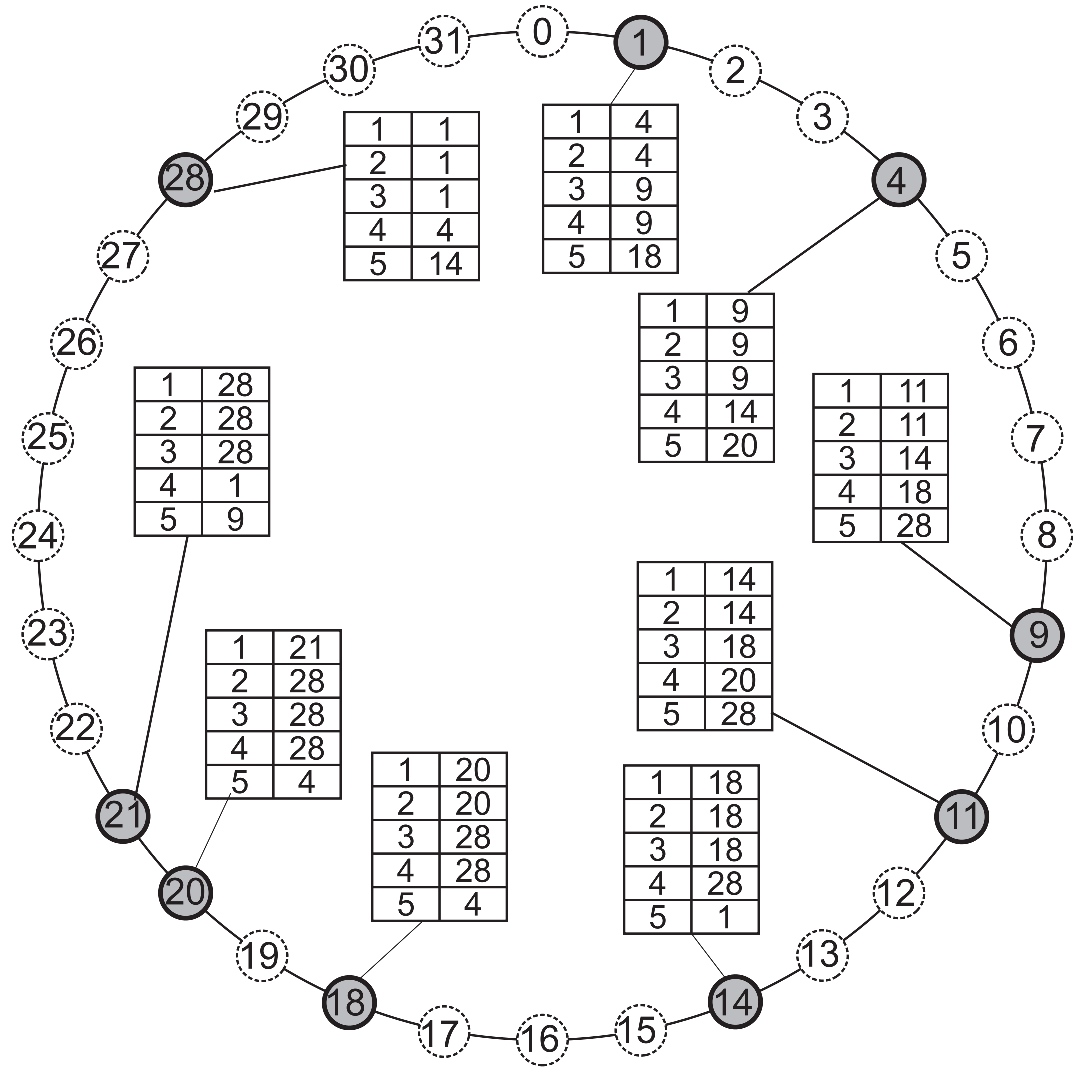
$$q = FT_p[j] \leq k < FT_p[j+1]$$

- If $p < k < FT_p[1]$, the request is also forwarded to $FT_p[1]$

Example finger tables

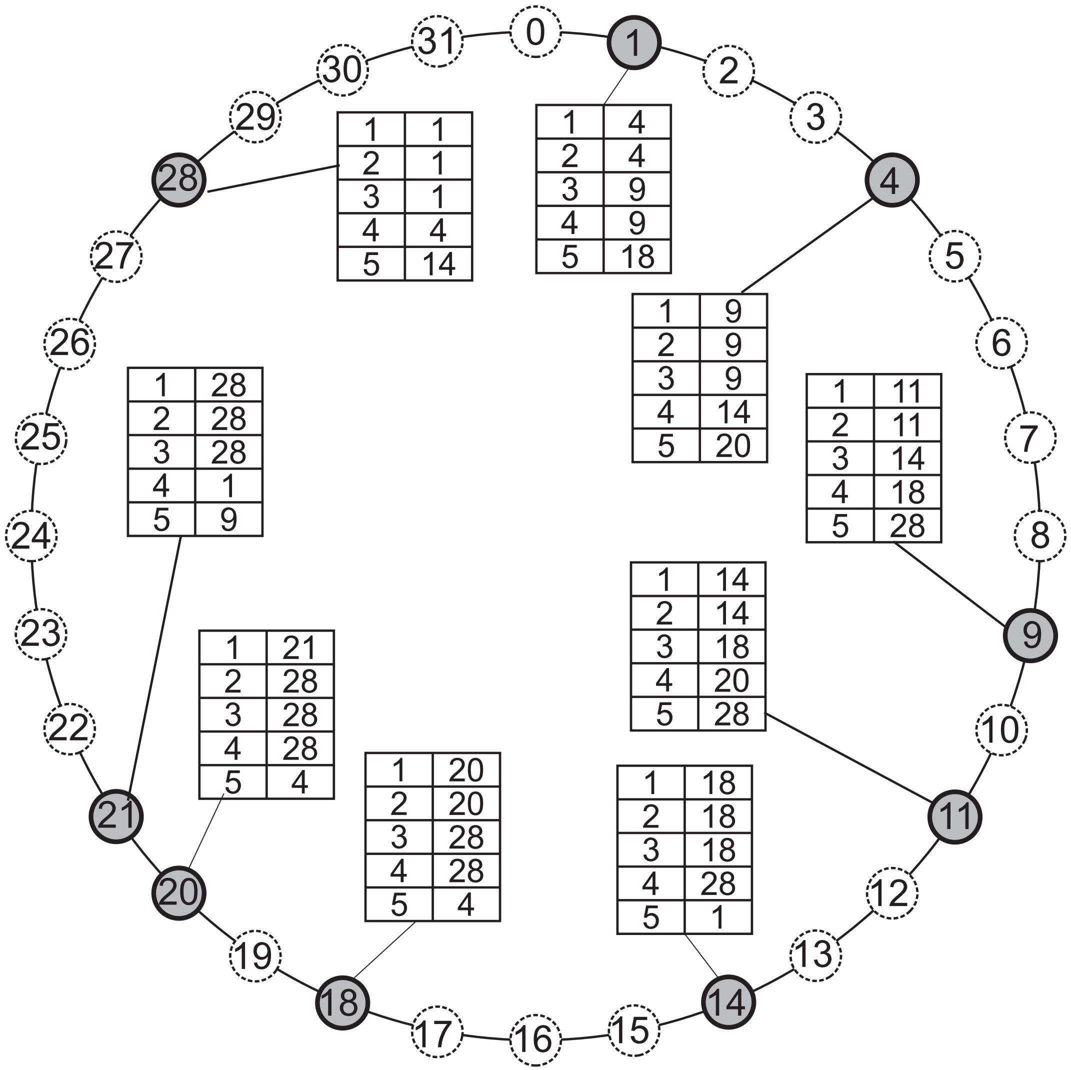


Example lookup: 15@4



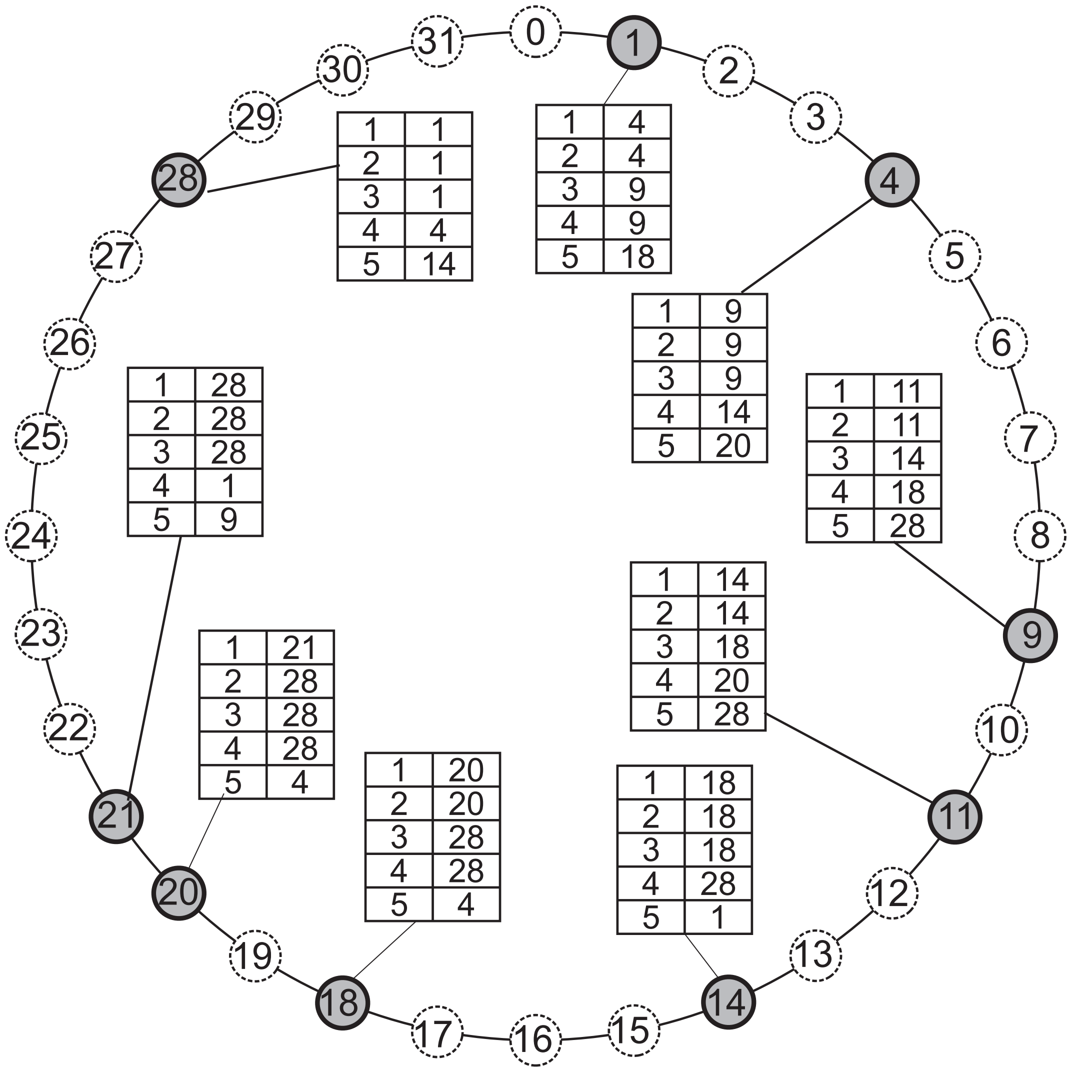
1. $FT_4[4] \leq 15 < FT_4[5]$
   $\Rightarrow 4 \rightarrow 14$
2. $p = 14 < 15 < FT_p[1]$
   $\Rightarrow 14 \rightarrow 18$

Example lookup: 22@4



1. $FT_4[5] \leq 22$
   $\Rightarrow 4 \rightarrow 20$
2. $FT_{20}[1] \leq 22 < FT_{20}[2]$
   $\Rightarrow 20 \rightarrow 21$
3. $p = 21 < 22 < FT_{21}[1]$
   $\Rightarrow 21 \rightarrow 28$

Example lookup: 18@20

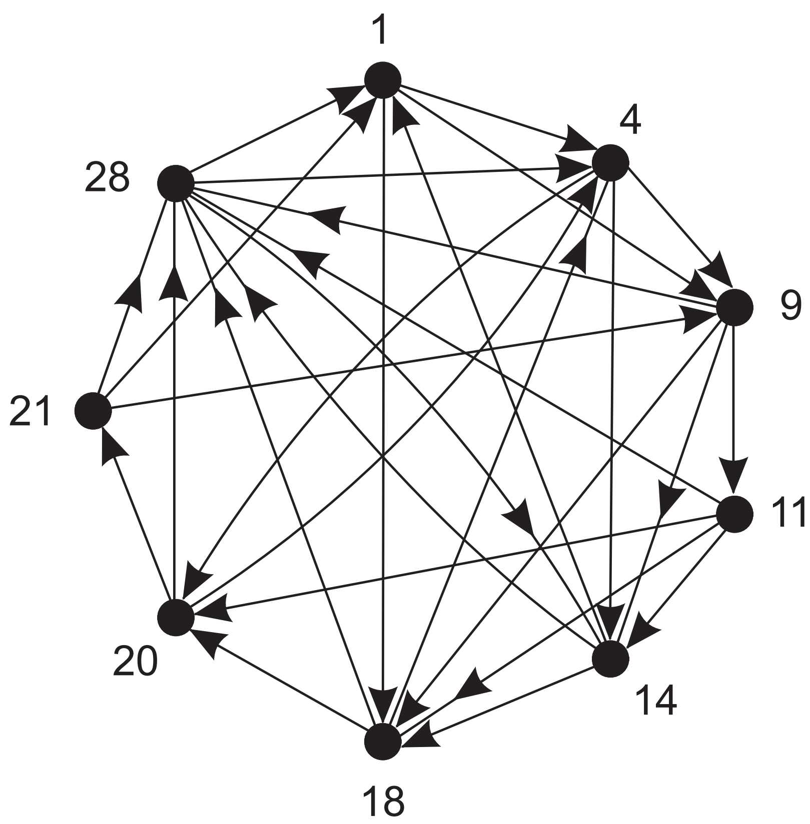


1. $p = 20 \not< 18 < FT_p[1]$
   $\not\Rightarrow 20 \rightarrow 21$
2. $FT_{20}[5] < 18$
   $\Rightarrow 20 \rightarrow 4$
3. $FT_4[4] \leq 18 < FT_4[5]$
   $\Rightarrow 4 \rightarrow 14$
4. $p = 14 < 18 < FT_p[1]$
   $\Rightarrow 14 \rightarrow 18$

The Chord graph

Essence

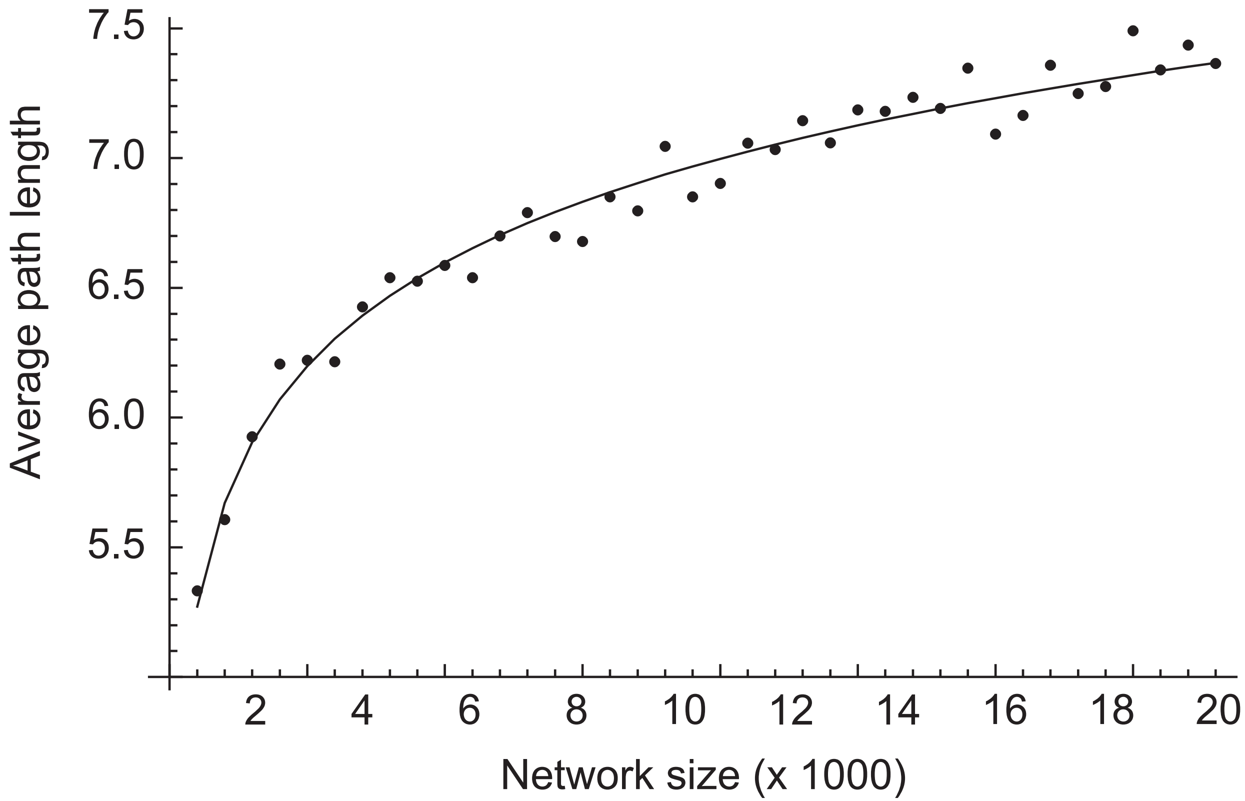
Each peer represented by a vertex; if $FT_p[i] = j$, add arc $\langle \overrightarrow{i,j} \rangle$, but keep directed graph strict.



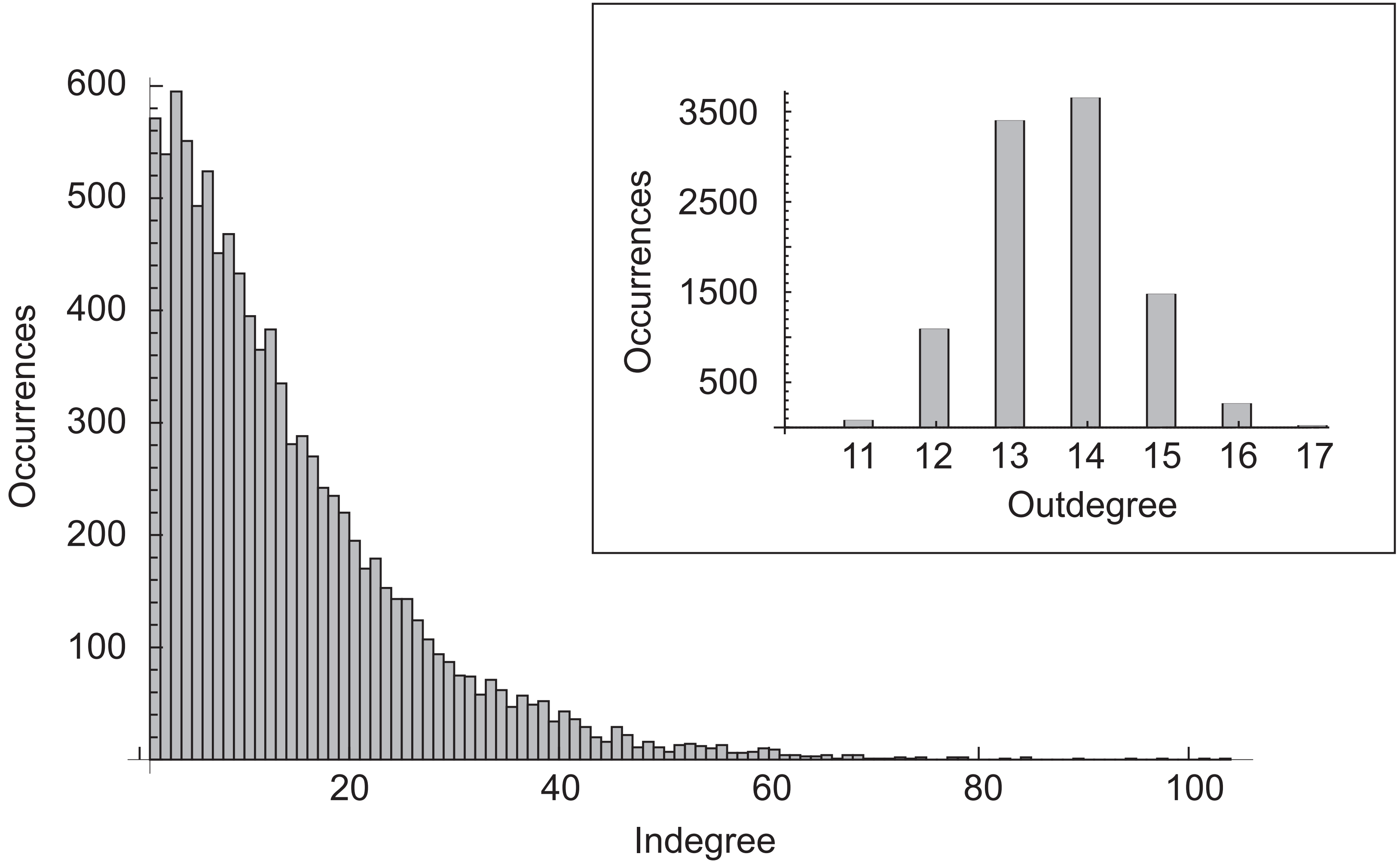
Chord: path lengths

Observation

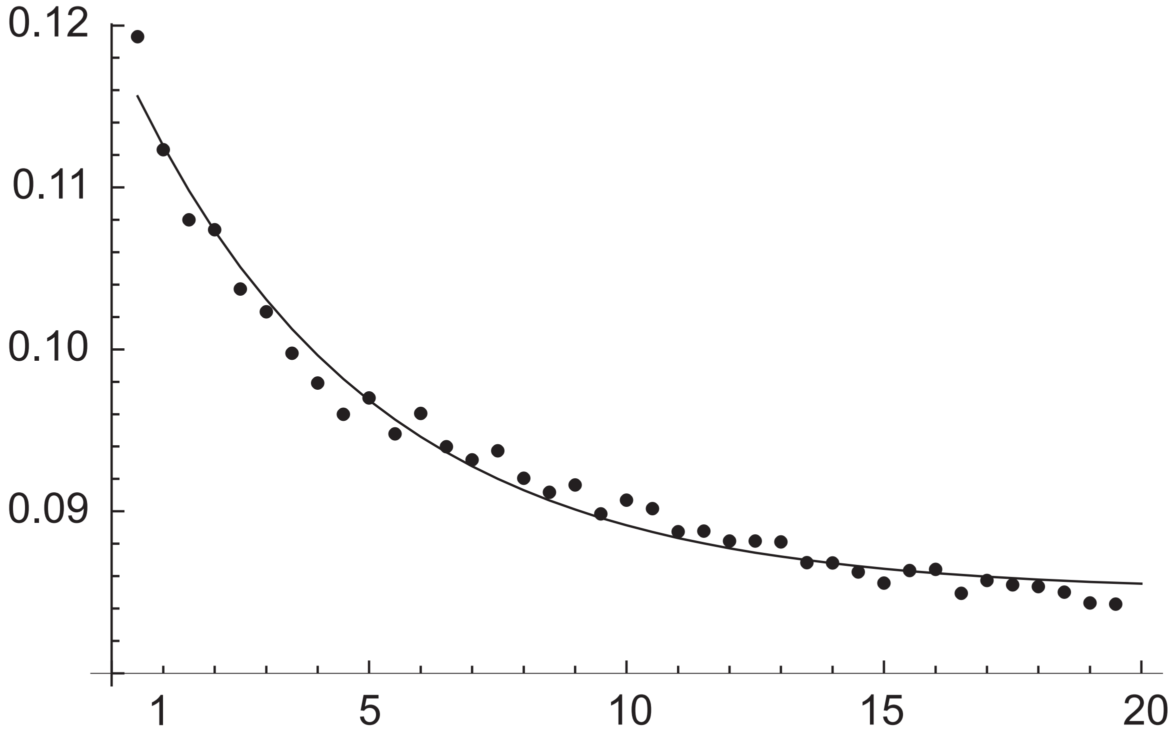
With $d_2^n(i,j) = \min\{|i-j|, n-|i-j|\}$, we can see that every peer is joined with another peer at distance $\frac{1}{2}n, \frac{1}{4}n, \frac{1}{8}n, \ldots, 1$.



Chord: degree distribution



Chord: clustering coefficient



Note

CC is computed over undirected Chord graph; *x*-axis shows number of 1000 nodes.

Epidemic-based networks

Basics

Consider a collection of peers $\mathbf{P} = \{p_1, \ldots, p_n\}$. Each peer can store lots of files. Each file f has a **version** $v(f)$. The **owner** of f is a single, unique peer, $own(f)$ who can update f.

Goal

We want to propagate updates of file f through a network of peers. $v(f, p)$ denotes version of file f at peer p. $FS(p)$ is set of files at peer p. If $f \notin FS(p) \Rightarrow v(f, p) = 0$.

$$\forall f, p : v(f, own(f)) \geq v(f, p)$$

Epidemic protocol

The core

Each peer $p \in \mathbf{P}$ **periodically** does the following:

1. $\forall f \in FS(p) : v(f,p) > v(f,q) \Rightarrow FS(q) \leftarrow FS(q) \cup \{f@p\}$
2. $\forall f \in FS(q) : v(f,p) < v(f,q) \Rightarrow FS(p) \leftarrow FS(p) \cup \{f@q\}$

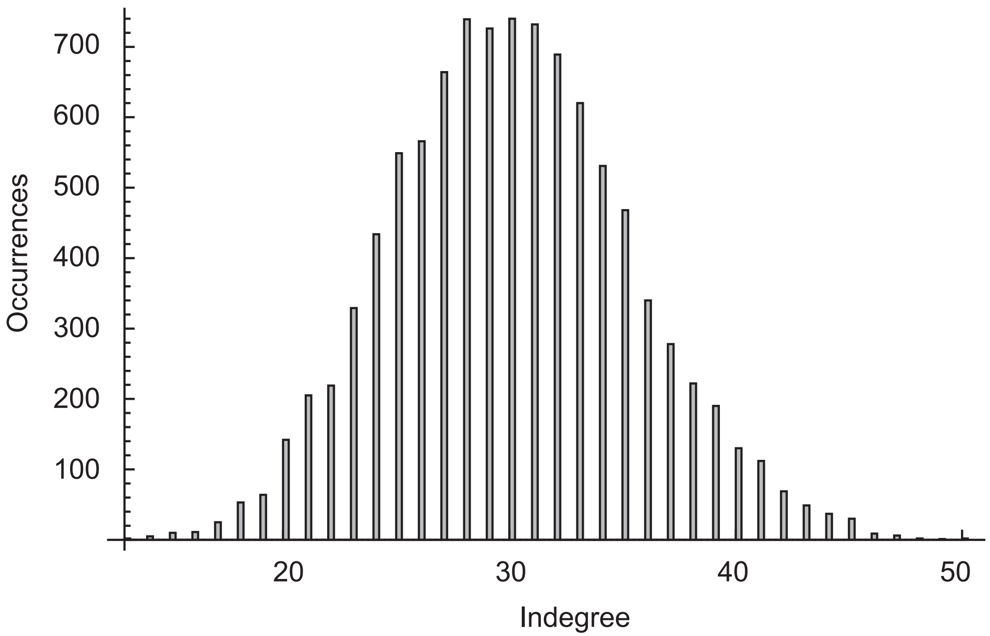
General framework

Active part	Passive part
repeat	**repeat**
wait T	**skip**
$q \leftarrow$ **select** 1 **from** PV_p	**skip**
$R_p \leftarrow$ **select** s **from** PV_p	**skip**
send $R_p \cup \{p\} \setminus \{q\}$ **to** q	**receive** R_p^q **from any** p
skip	$R_q \leftarrow$ **select** s **from** PV_q
receive R_q^p **from** q	**send** $R_q \cup \{q\} \setminus \{p\}$ **to** p
$PV_p \leftarrow$ **select** m **from** $PV_p \cup R_q^p$	$PV_q \leftarrow$ **select** m **from** $PV_q \cup R_p^q$
until forever	**until** forever

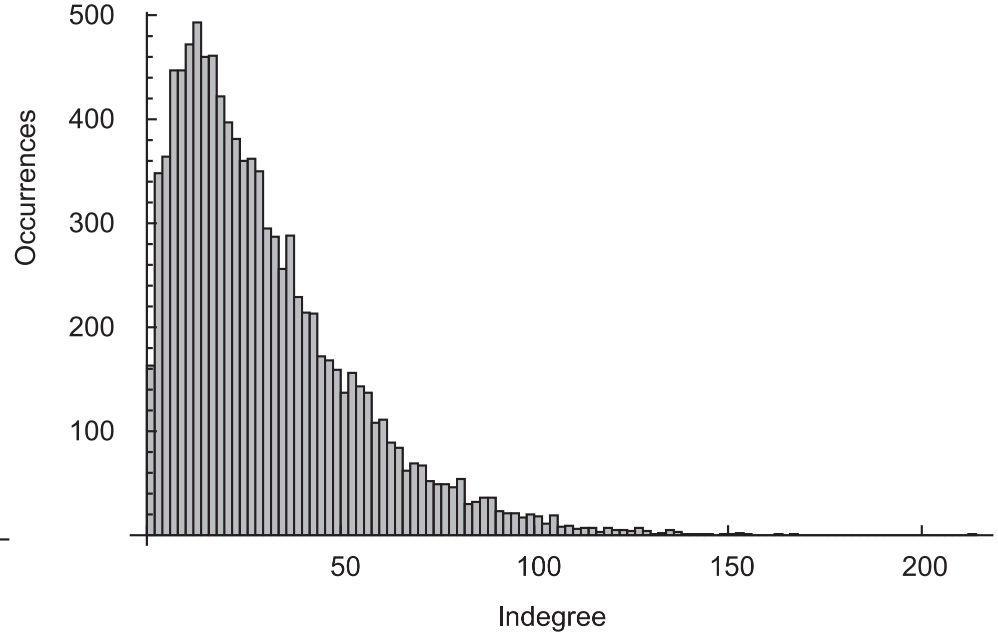
Newscast

Issue	Policy	Description
view size	$m = 30$	Each partial view has size 30
peer selection	random	Each peer uniformly at random selects a peer from its partial view
reference selection	random	A random selection of s peers is selected from a partial view to be exchanged with the selected peer
view size reduction	random	If the view size has grown beyond m, a random selection of references is removed to bring it back to size m

Newscast: evolution indegree distribution

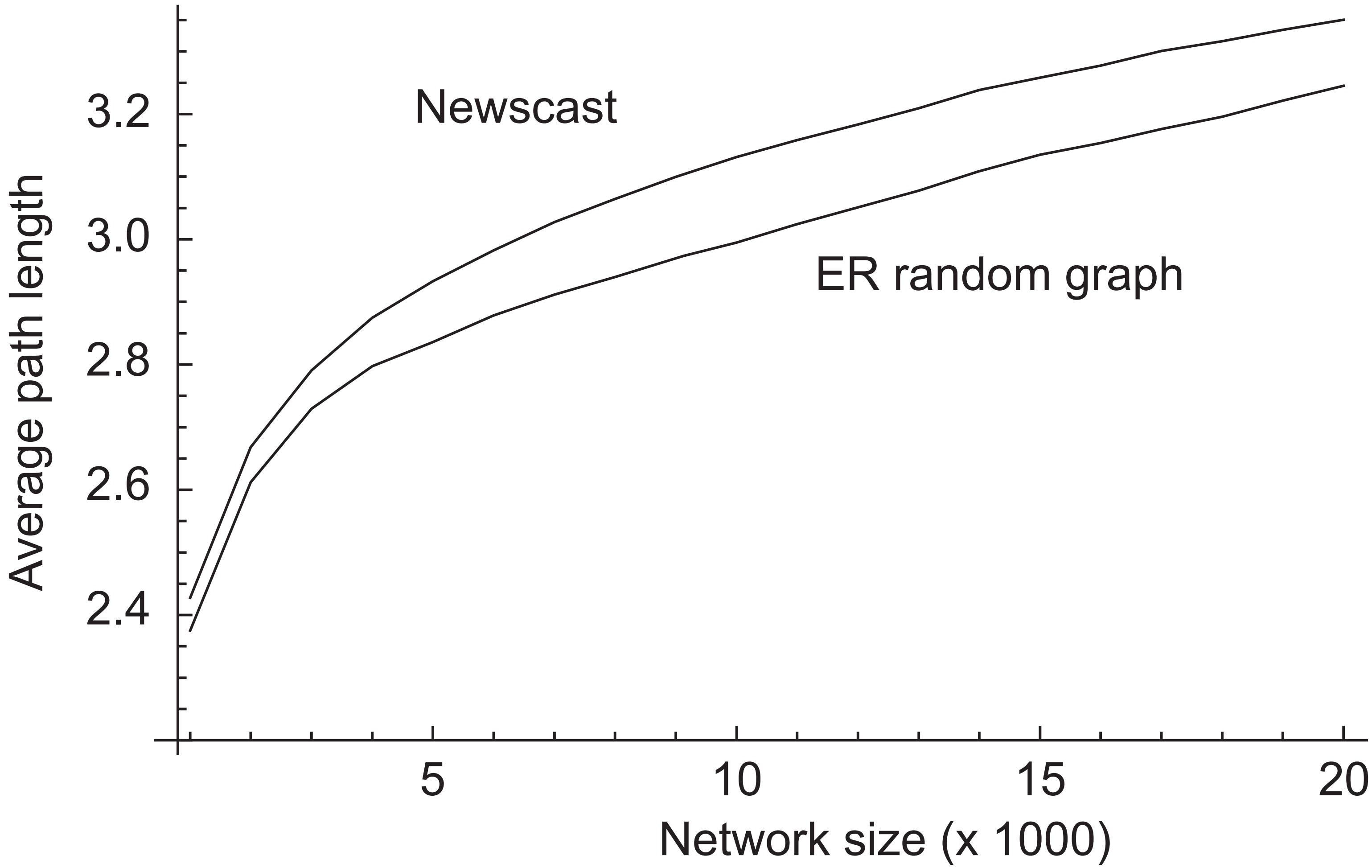


Initially

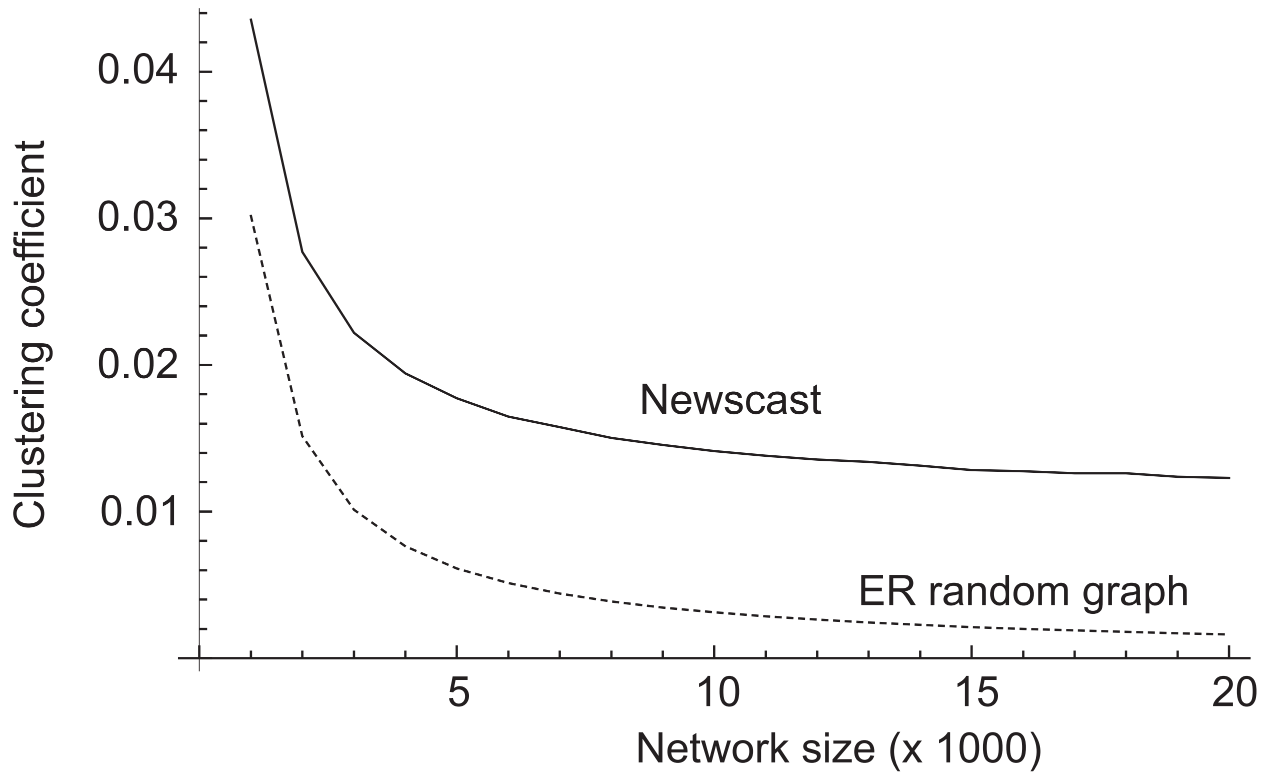


After 200 rounds

Newscast: evolution path length



Newscast: evolution cluster coefficient



Question

For which kind of $ER(n, p)$ graphs is this a fair comparison?

The Web

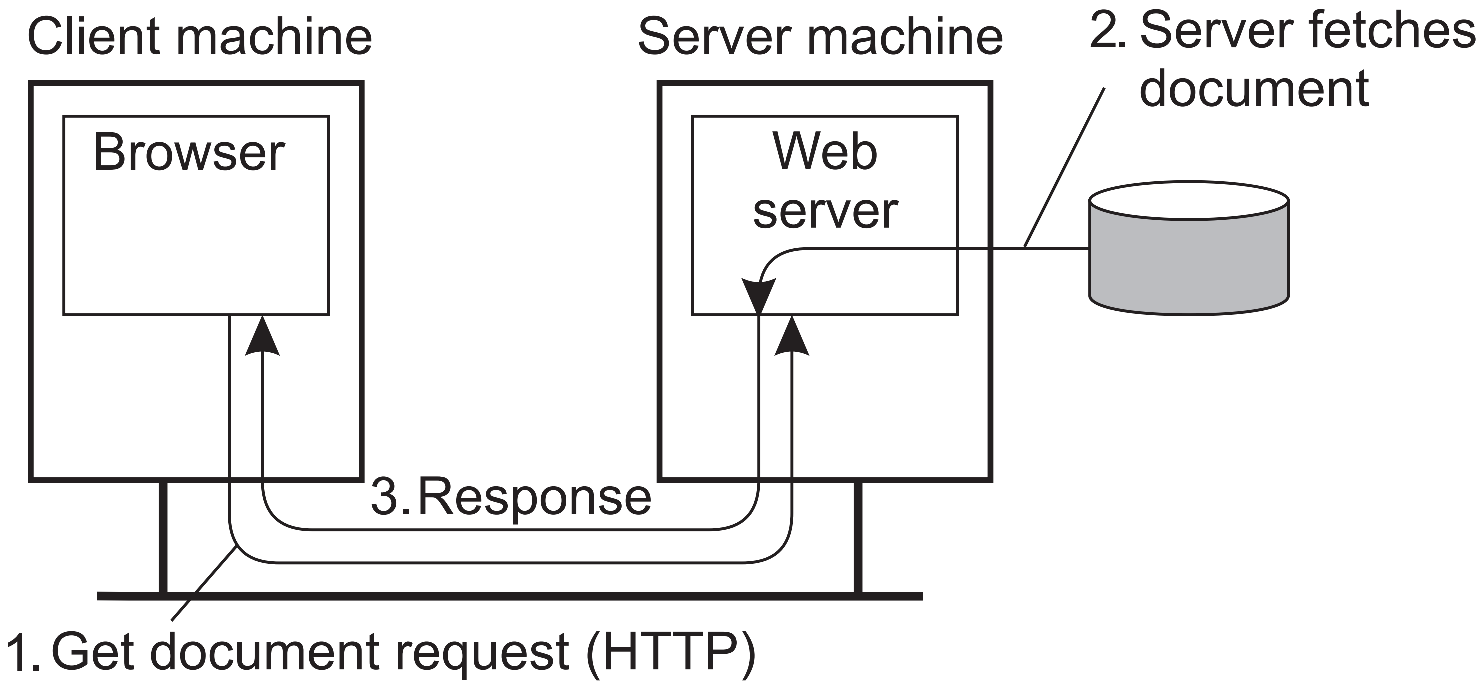
Web basics

- **Simple view**: the Web consists of **hyperlinked documents**.
- Hyperlinked: document *A* carries a reference to document *B*. When reference is **activated**, browser fetches document *B*.
- Collection of documents forms a **site**, with its own associated **domain name**.

Some numbers

It has been estimated that by 2008, there were at least 75 million Web sites from which Google had discovered more than a trillion Web pages.

Web basics



Client machine
Server machine
2. Server fetches document
Browser
Web server
3. Response
1. Get document request (HTTP)

Measuring the topology of the Web

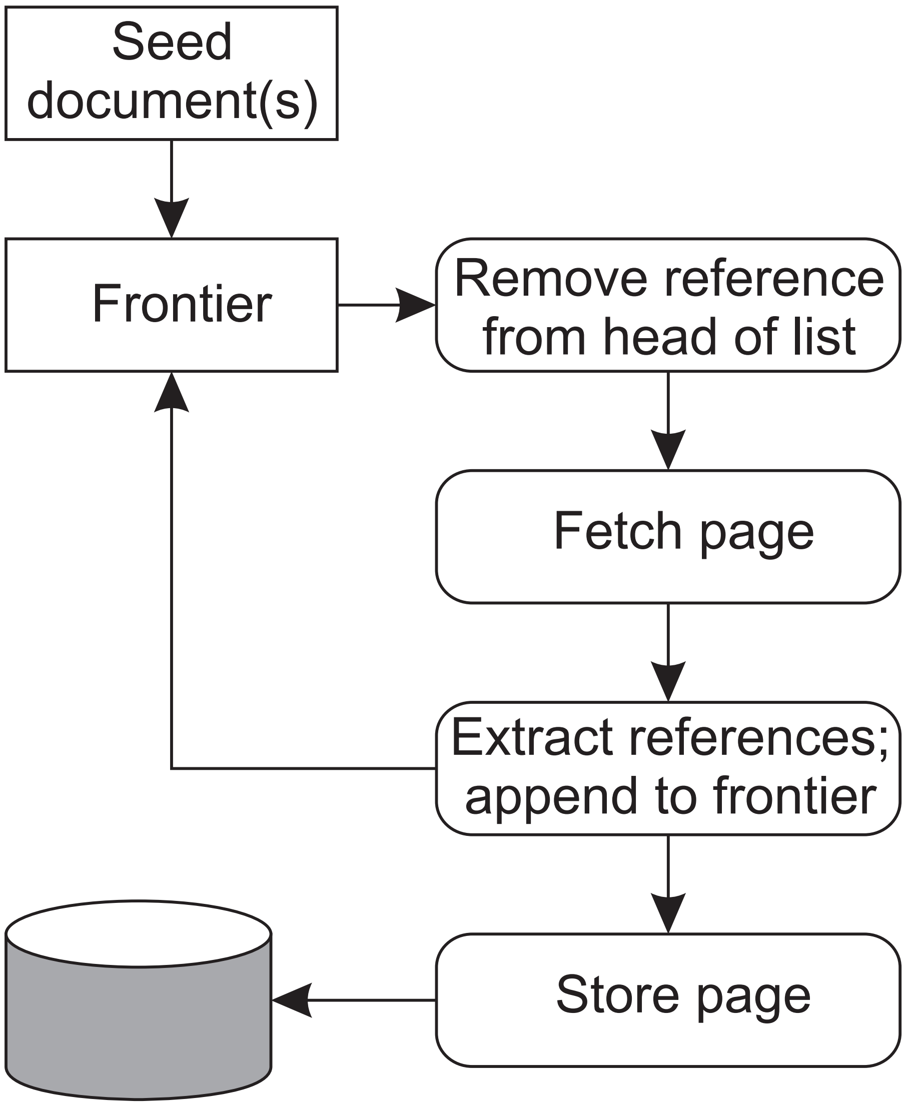
Problem

With an estimated size of over a trillion Web pages, pages coming and going, and links changing all the time, how can we ever get a **snapshot** of the Web? **We can't**.

Practical issue: crawling the Web

In order to measure anything, we need to be able to identify pages and the links that refer to them.

Web crawler

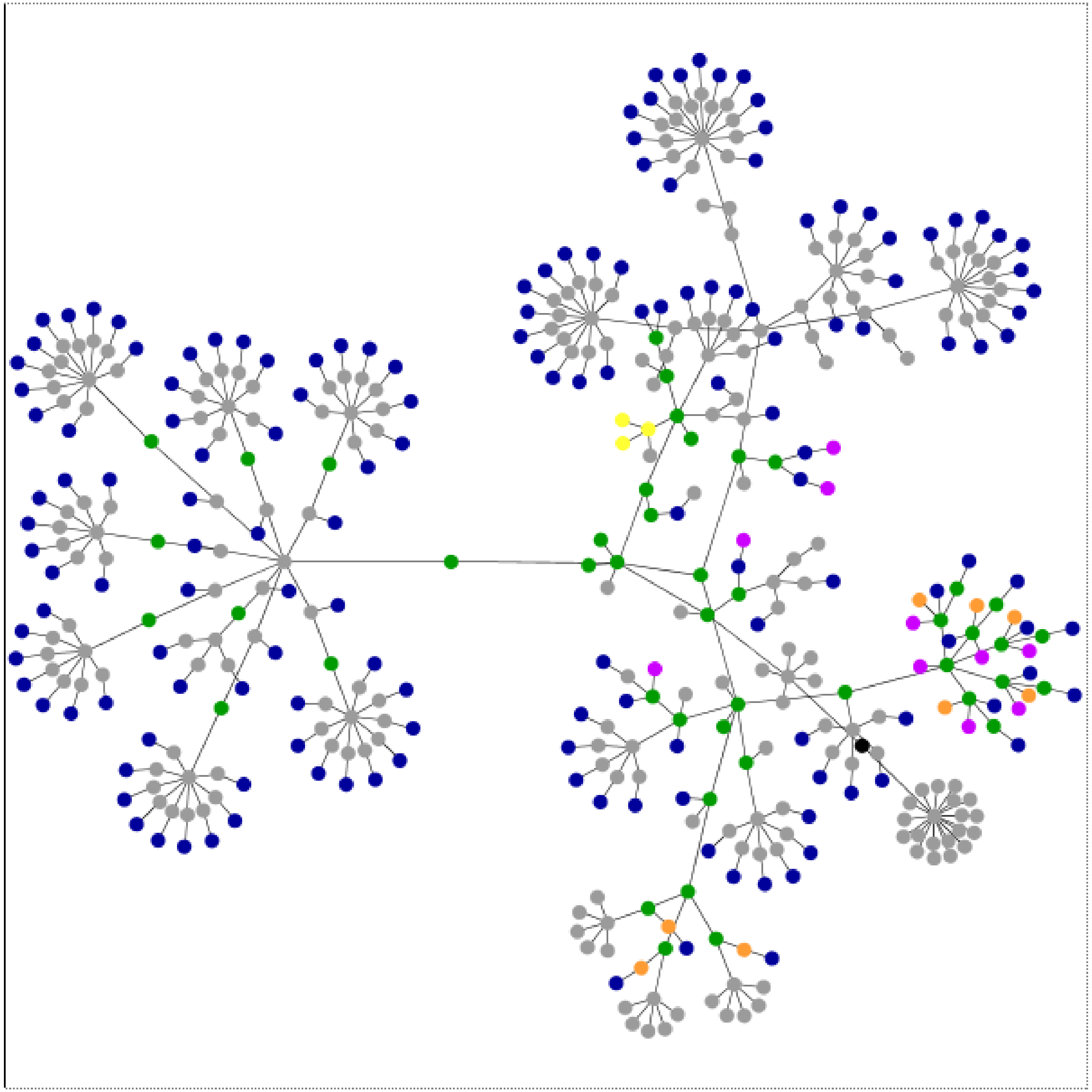


- Start with seed pages
- Store pages to inspect in frontier
- Analyze page and store found references in frontier

Observation

Using seed documents, we are accessing the **reachable pages**.

Webpages as graphs: The VU website



Sampling the Web

Observation

The Web is so huge, that we can only hope to **draw a reasonable sample**, and hope that this sample represents the structure of the actual Web. **We are asking for trouble**.

Starting point

Let us try to represent the Web as a **bowtie**:

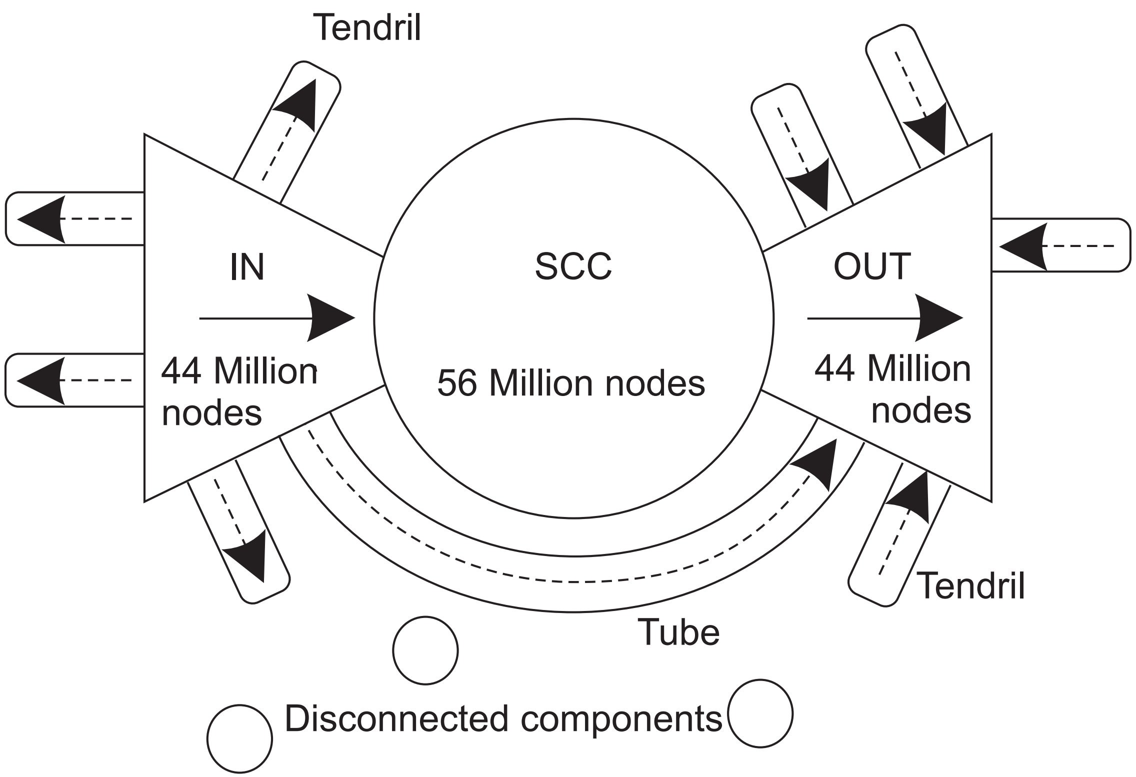
SCC: $\forall v, w \in SCC, \exists (v, w)$-path of hyperlinks.

IN: $\forall v \in IN, w \in SCC : \exists (v, w)$-path, but no (w, v)-path.

OUT: $\forall v \in SCC, w \in OUT : \exists (v, w)$-path, but no (w, v)-path.

TENDRILS: Essentially: the rest.

The Web as a bowtie: Starting from AltaVista



Sampling the Web

Observation

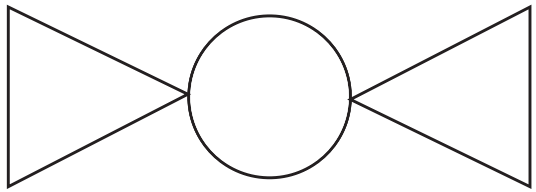
It turns out that for different seeds, we do obtain different bowties.

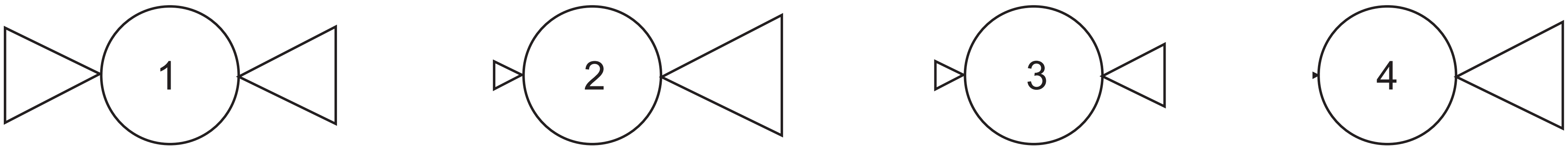
Component	Sample 1	Sample 2	Sample 3	Sample 4
SCC	56.46%	65.28%	85.87%	72.30%
IN	17.24%	1.69%	2.28%	0.03%
OUT	17.94%	31.88%	11.26%	27.64%
Other	8.36%	1.15%	0.59%	0.02%
Total size	80.57M	18.52M	49.30M	41.29M

Sampling the Web

Component	Sample 1	Sample 2	Sample 3	Sample 4
SCC	56.46%	65.28%	85.87%	72.30%
IN	17.24%	1.69%	2.28%	0.03%
OUT	17.94%	31.88%	11.26%	27.64%
Other	8.36%	1.15%	0.59%	0.02%
Total size	80.57M	18.52M	49.30M	41.29M

AltaVista

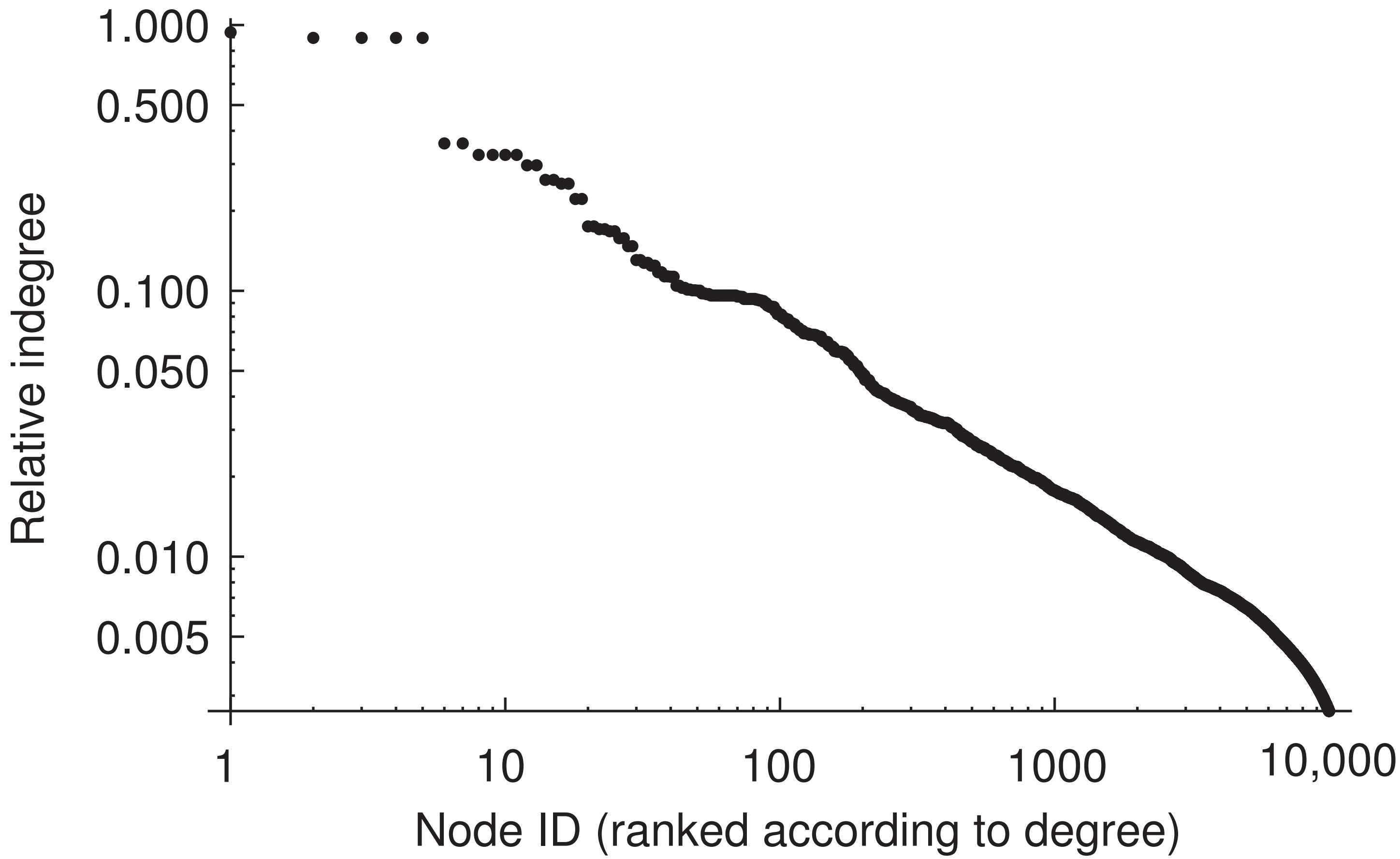




Question

Which conclusion can we draw from these samples?

Web graphs: indegree distribution



Observation

It turns out that $\mathbb{P}[\delta_{in} = k] \propto \frac{1}{k^{2.1}} \Rightarrow$ another scale-free network.

Side step: Google's PageRank

Observation

Google uses hyperlinks *to* a page p as a criterion for the *importance* of a page:

$$rank(p) = (1 - d) + d \sum_{\langle \overrightarrow{q,p} \rangle \in E} \frac{rank(q)}{\delta_{out}(q)}$$

where $d \in [0, 1)$ is a constant (probably 0.85 in the case of Google).

Question

This is a **recursive** definition. What's going on?

Side step: Google's PageRank

Observation

PageRank is clearly based on indegrees, yet the rank of a page and its indegree turn out to be only weakly correlated.

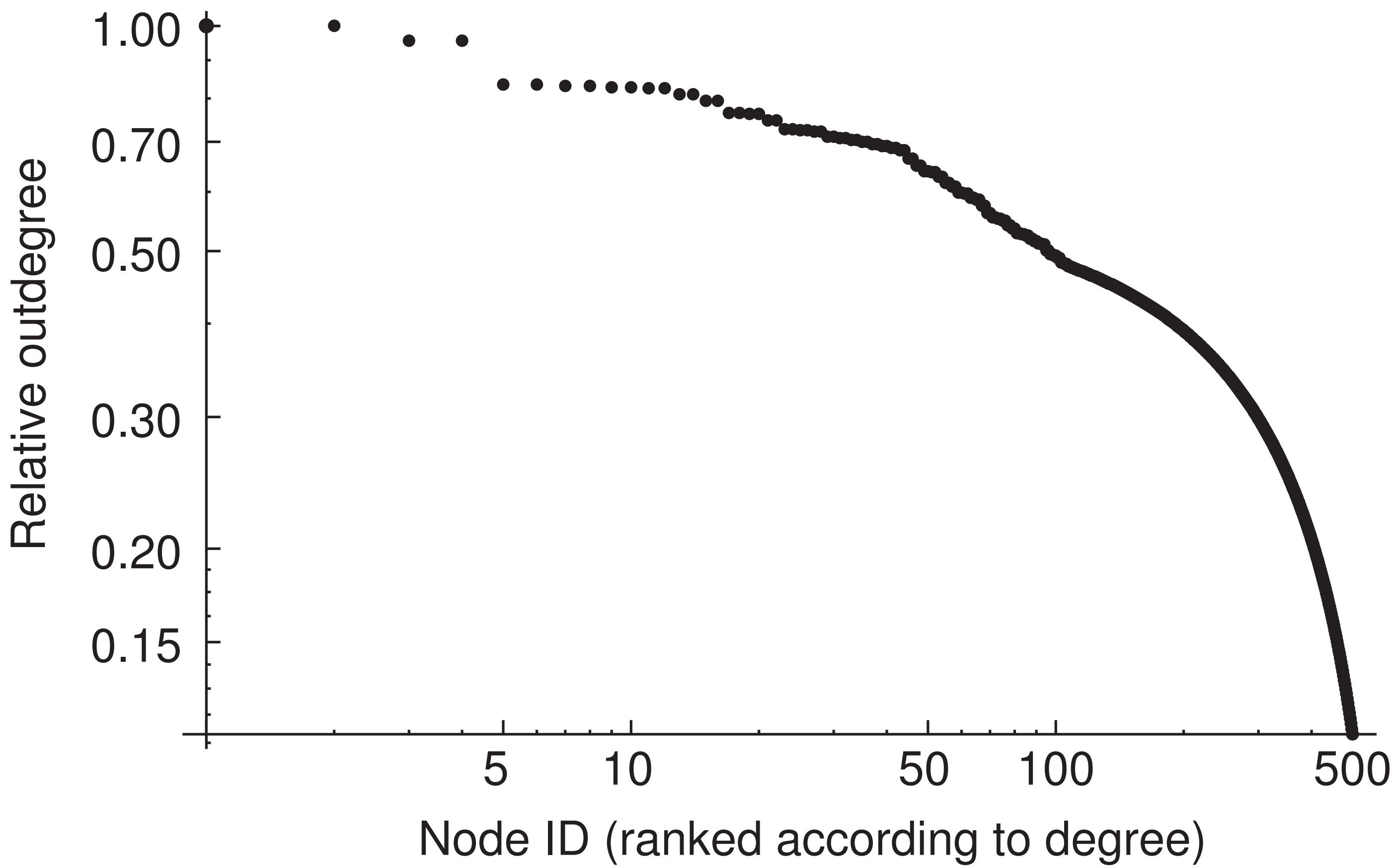
Observation

When we rank pages according to PageRank: $\mathbb{P}[rank = k] \propto \frac{1}{k^{2.1}}$

Observation

Characterizing and sampling the Web is again seen to be far from trivial.

Web graphs: oudegree distribution



Observation

To analyze the Web graph, we need to be very careful regarding measurements and conclusions.