CMSC5719 Introductions to Social Computing

Irwin King

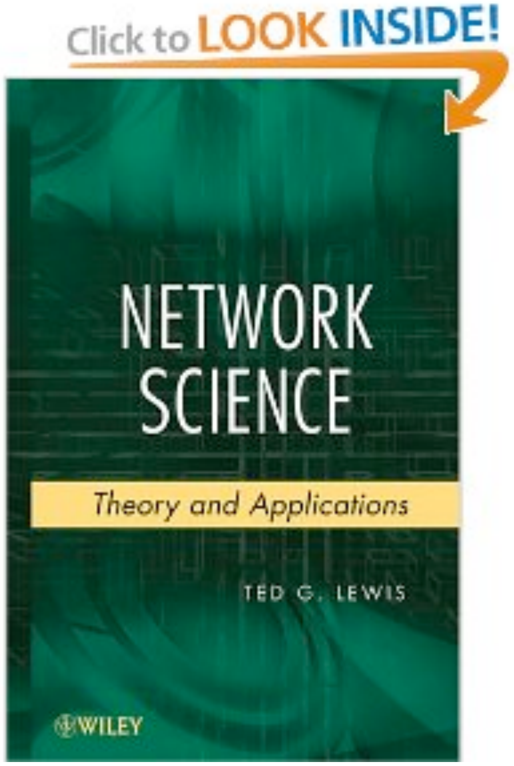
The Chinese University of Hong Kong

http://www.cse.cuhk.edu.hk/irwin.king/confs/cmsc5719
king@cse.cuhk.edu.hk

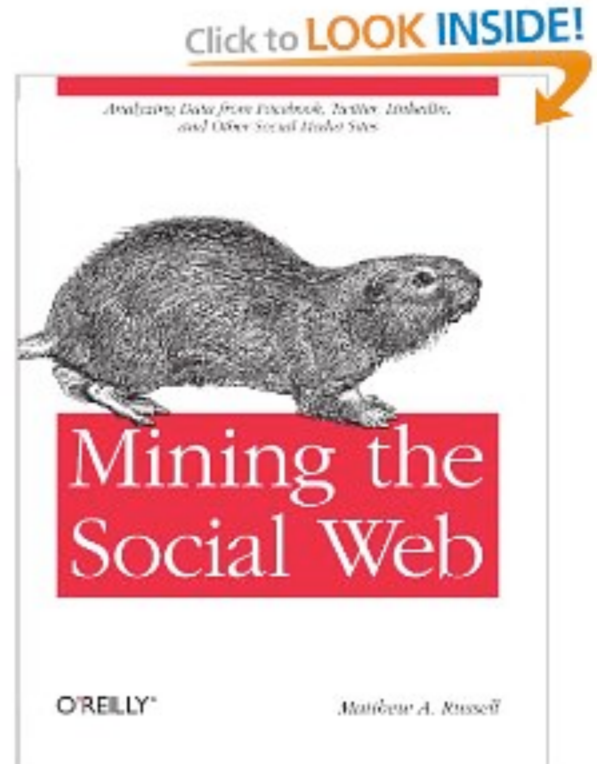
Course Information

- Irwin King
- 908 Ho Sin Hang Engineering Building
  - http://www.cse.cuhk.edu.hk/irwin.king
  - email: king@cse.cuhk.edu.hk
  - http://www.cse.cuhk.edu.hk/irwin.king/confs/cmsc5719
  - Office hours: M8 and T8 from 3:30 pm - 4:30 pm

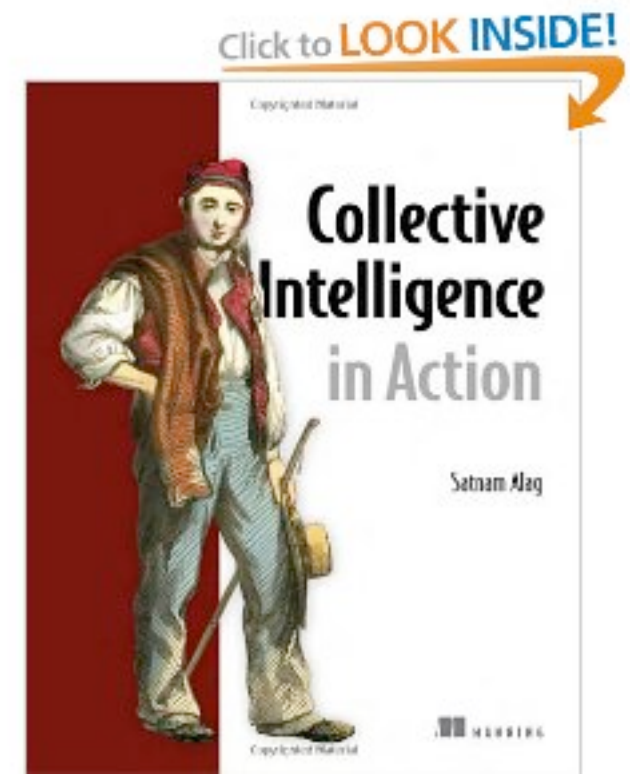
References



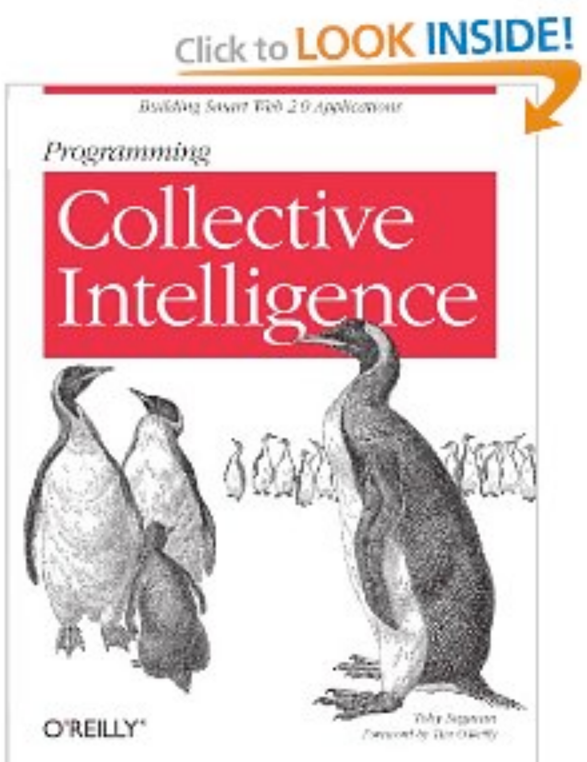
Network Science: Theory and Applications



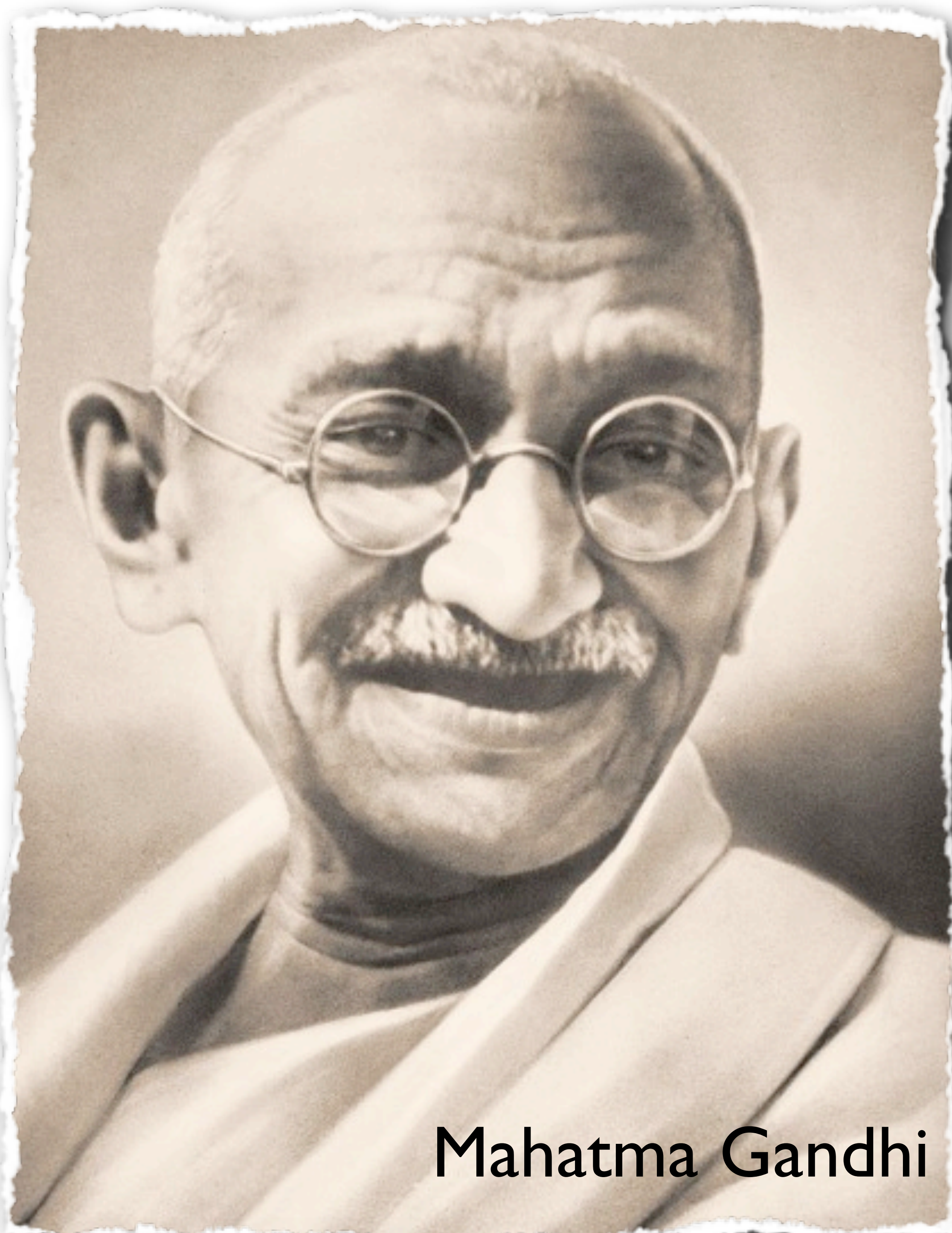
Mining the Social Web: Analyzing Data from Facebook, Twitter, LinkedIn, and Other Social Media Sites



Collective Intelligence in Action



Programming Collective Intelligence: Building Smart Web 2.0 Applications

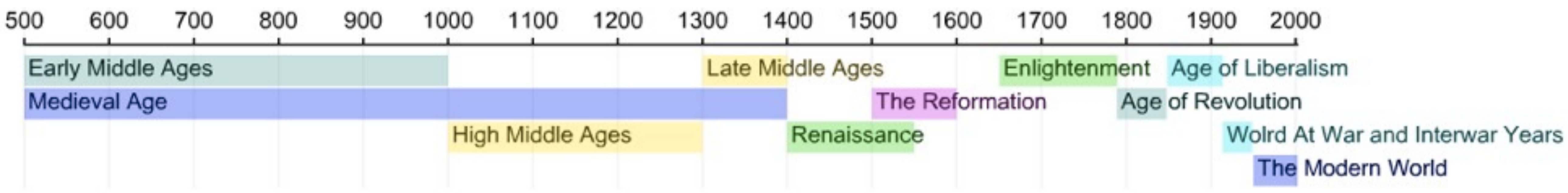


Mahatma Gandhi

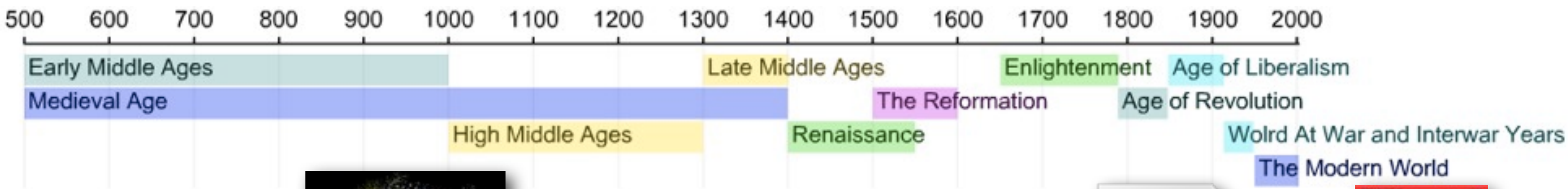
Interdependence is and ought to be as much the ideal of man as self-sufficiency.

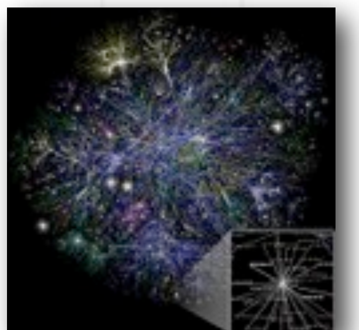
Man is a social being.

A Brief History of the World



A Brief History of the World

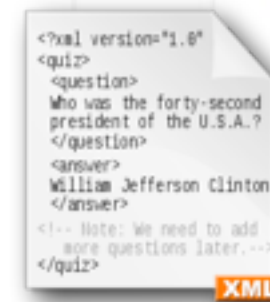




Birth of Internet

IBM Desktop PC

Apple Macintosh



Birth of XML

Time Magazine
Person of the Year

1750 1945 1969 1975 1981 1983 1984 1989 1996 2004 2006

Industrial Revolution

Information Age

Internet Age

WWW Age

Attention Age

ENIAC

The MITS Altair
Apple II

Time Magazine
Person of the Year

Birth of WWW

Birth of Web 2.0


intel.
revolution in
evolution
Highlights from the Journey to 1 Billion PCs
1,000,000,000
For more information, please visit http://www.intel.com

Social Networking

Billionaires' Shuffle

2007

Facebook in 2004.02

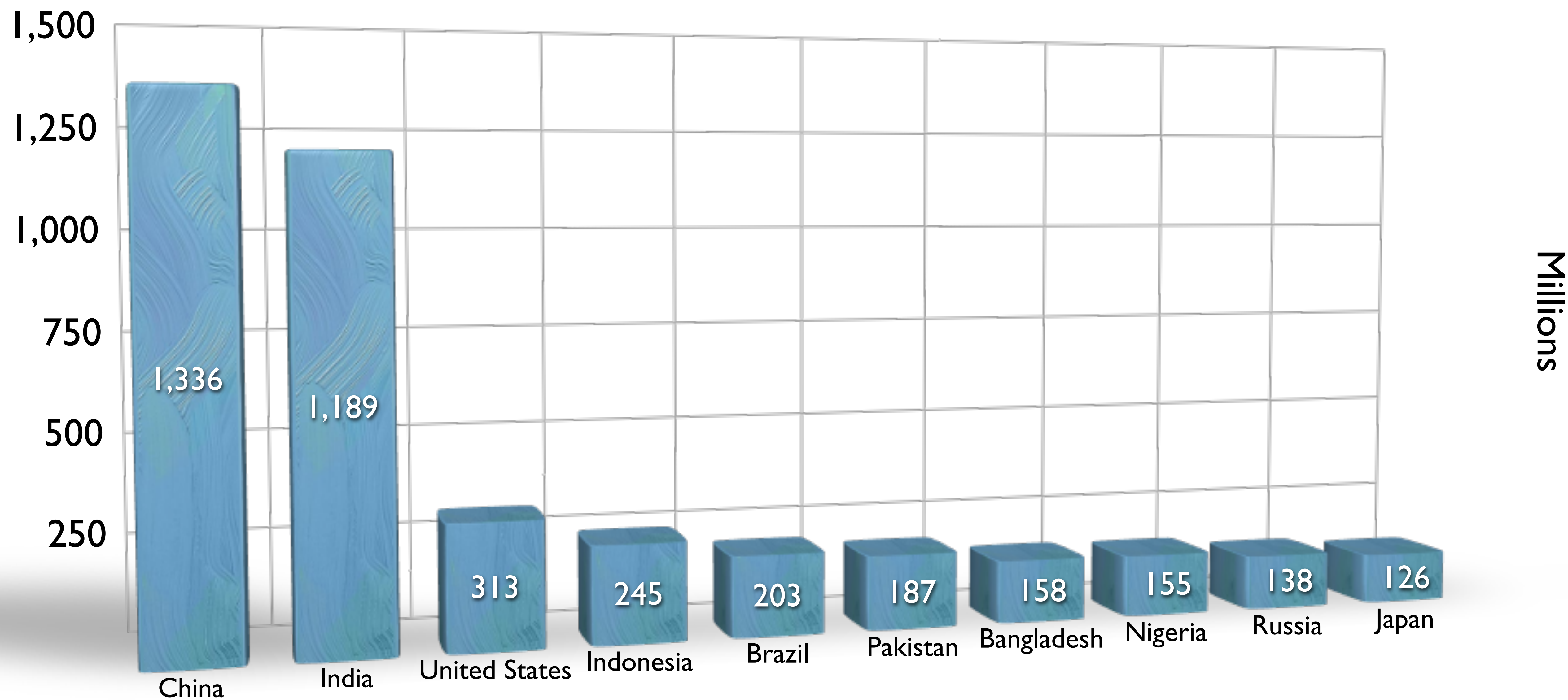
2008

at **23** and $**1.5** billion later...

2008

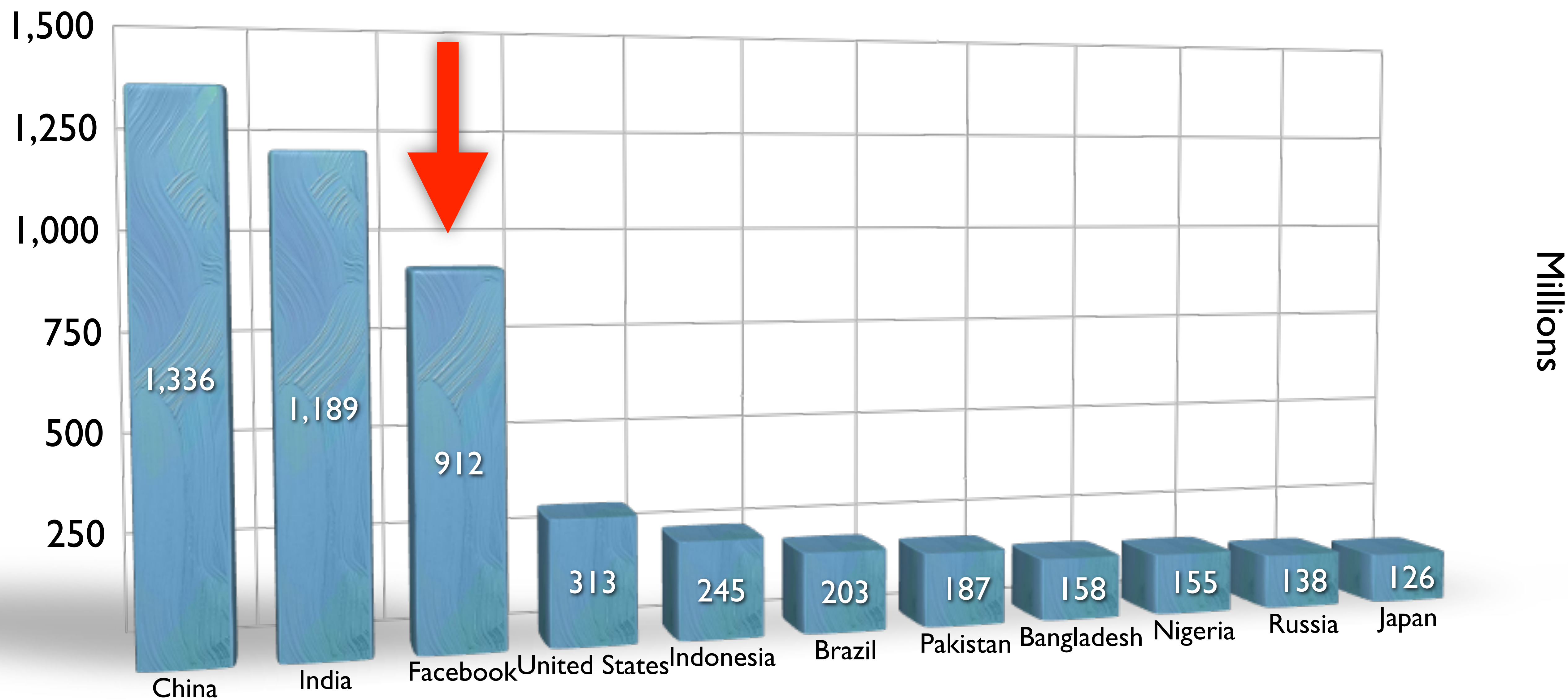
Top 10 Populations by Countries

as of December 31, 2011

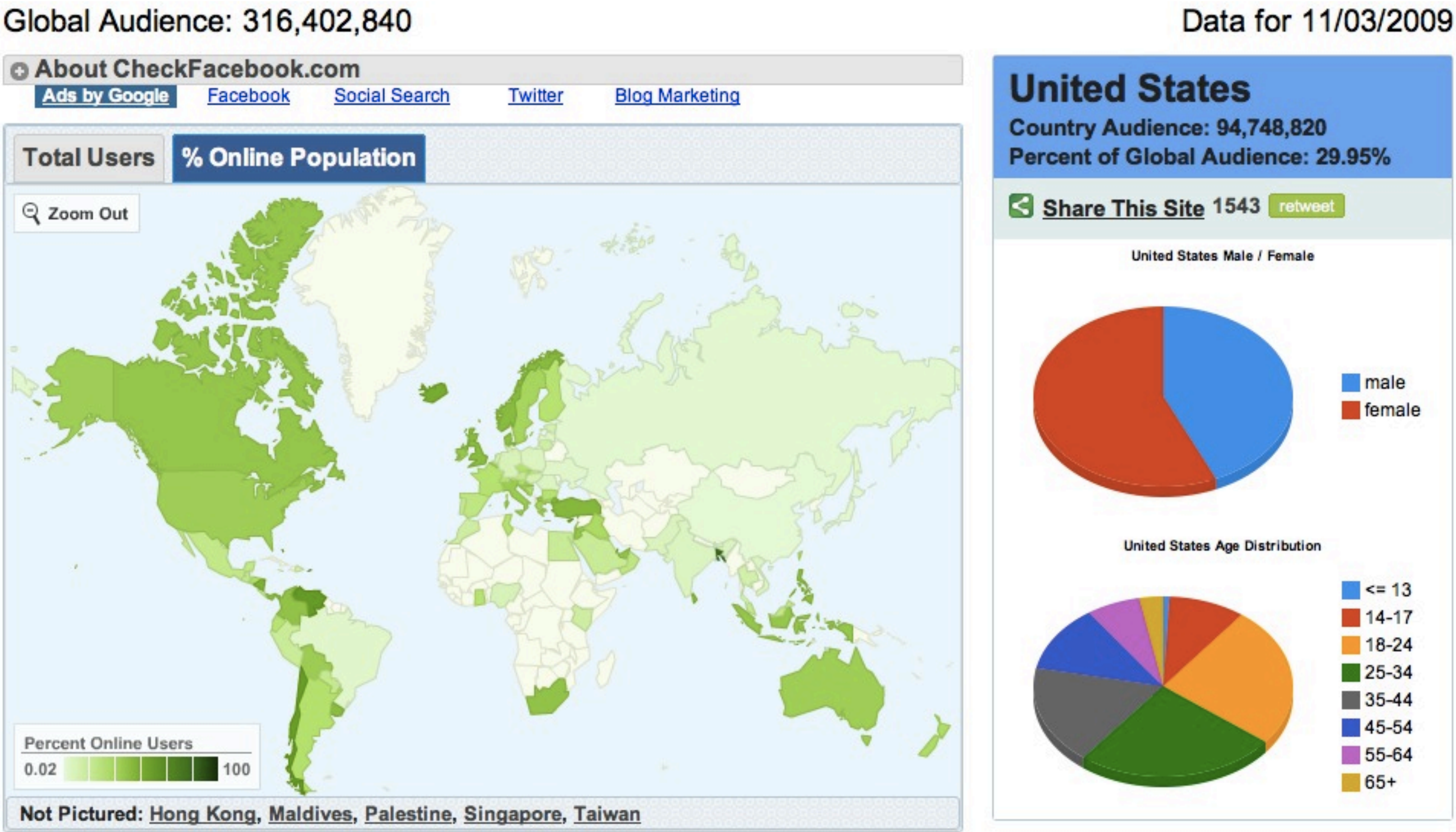


Top 10 Populations by Countries

as of September 8, 2012



Facebook's Global Audience



Facebook’s Global Audience

Facebook's Growth Stats

Statistics

Company Figures	More than 400 million active users
	50% of our active users log on to Facebook in any given day
	More than 35 million users update their status each day
	More than 60 million status updates posted each day
	More than 3 billion photos uploaded to the site each month
	More than 5 billion pieces of content (web links, news stories, blog posts, notes, photo albums, etc.) shared each week

10 Largest Countries		
1.	United States	94,748,820
2.	United Kingdom	22,261,080
3.	Turkey	14,215,880
4.	France	13,396,760
5.	Canada	13,228,380
6.	Italy	12,581,060
7.	Indonesia	11,759,980
8.	Spain	7,313,160
9.	Australia	7,176,640
10.	Philippines	6,991,040

10 Fastest Growing Over Past Week			
1.	Poland	12.46 %	137,900
2.	Thailand	10.96 %	161,300
3.	Portugal	9.81 %	80,040
4.	South Africa	9.25 %	189,080
5.	Taiwan	7.82 %	367,400
6.	Romania	7.65 %	28,060
7.	Germany	7.54 %	350,240
8.	Malaysia	7.43 %	236,840
9.	Indonesia	6.84 %	752,640
10.	Iraq	6.72 %	6,380

Facebook's Growth Stats

(as of September 2012)

Statistics

955 million monthly active users at the end of June 2012.

Approximately 81% of our monthly active users are outside the U.S. and Canada.

552 million daily active users on average in June 2012.

543 million monthly active users who used Facebook mobile products in June 2012.

10 Largest Countries		
1.	United States	163,358,340
2.	Brazil	56,804,900
3.	India	53,624,320
4.	Indonesia	44,156,440
5.	United Kingdom	40,036,380
6.	Mexico	37,542,740
7.	Turkey	31,108,760
8.	Philippines	29,136,040
9.	France	24,639,540
10.	Germany	24,300,340

10 Fastest Growing Over Past Week			
1.	Vietnam	100.09 %	3,598,480
2.	Brazil	0.18 %	100,060
3.	Thailand	0.21 %	34,780
4.	Colombia	0.19 %	32,060
5.	Romania	0.44 %	21,940
6.	Croatia	0.84 %	13,120
7.	Netherlands Antilles	12.76 %	9,720
8.	Canada	0.05 %	9,700
9.	Chile	0.10 %	9,560
10.	Jordan	0.37 %	8,980

Global Internet Traffic

Alexa as of August 2011	China	USA	Japan	India	Brazil	Global
1	Baidu	Google	Yahoo.jp	Google.in	Google.br	Google
2	**QQ**	**Facebook**	Google.jp	Google	Google	**Facebook**
3	Sina	Yahoo!	**FC2**	**Facebook**	**Facebook**	**YouTube**
4	Taobao	**YouTube**	**YouTube**	**YouTube**	**YouTube**	Yahoo!
5	Google.hk	Amazon	Google	Yahoo!	Universo Online	**Blogger**
6	163	**Wikipedia**	**Ameblo.jp**	**Blogger**	Windows Live	Baidu
7	**Weibo**	**Blogger**	rakuten	**Wikipedia**	Globo	**Wikipedia**
8	Google	**Twitter**	**livdoor**	**LinkedIn**	**Orkut.com.br**	Windows Live
9	ifeng	eBay	**Facebook**	**Twitter**	Yahoo!	**Twitter**
10	Yahoo	Craigslist	**Wikipedia**	Rediff	**Orkut.com**	**QQ**

Alexa as of May 2009	China	USA	Japan	India	Brazil	Global
1	Baidu	Google	Yahoo.jp	Google.in	Google	Google
2	**QQ**	Yahoo!	**FC2**	Google	**Orkut.br**	Yahoo!
3	Sina	**Facebook**	Google.jp	Yahoo	Windows Live	**YouTube**
4	Google.cn	**YouTube**	**YouTube**	**Orkut.in**	Universo Online	**Facebook**
5	Taobao	**Myspace**	Rakuten	**YouTube**	**YouTube**	Windows Live
6	163	MSN	Livedoor	**Blogger**	Globo	MSN
7	Google	Windows Live	**Ameblo.jp**	Rediff	MSN	**Wikipedia**
8	Sohu	**Wikipedia**	**mixi**	**Facebook**	Google	**Blogger**
9	Youku	Craigslist	**Wikipedia**	**Wikipedia**	Yahoo!	Baidu
10	Yahoo	EBay	Google	Windows Live	Terra	**Myspace**

Alexa as of August 2011	China	USA	Japan	India	Brazil	Global
1	Baidu	Google	Yahoo.jp	Google.in	Google.br	Google
2	**QQ**	**Facebook**	Google.jp	Google	Google	**Facebook**
3	Sina	Yahoo!	**FC2**	**Facebook**	**Facebook**	**YouTube**
4	Taobao	**YouTube**	**YouTube**	**YouTube**	**YouTube**	Yahoo!
5	Google.hk	Amazon	Google	Yahoo!	Universo Online	**Blogger**
6	163	**Wikipedia**	**Ameblo.jp**	**Blogger**	Windows Live	Baidu
7	**Weibo**	**Blogger**	rakuten	**Wikipedia**	Globo	**Wikipedia**
8	Google	**Twitter**	**livdoor**	**LinkedIn**	**Orkut.com.br**	Windows Live
9	ifeng	eBay	**Facebook**	**Twitter**	Yahoo!	**Twitter**
10	Yahoo	Craigslist	**Wikipedia**	Rediff	**Orkut.com**	**QQ**

The Brave New Words

博客

維基

AVATAR

頭像

tag cloud

推特

unfriend

tweet

blogsphere

twitterati

defriend

SEXTING

hashtags

Folksonomy

Politics

HOME PAGE | TODAY'S PAPER | VIDEO | MOST POPULAR | TIMES TOPICS

The New York Times
Friday, June 19, 2009

News

Search All NYTimes.com
Go

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION | ARTS | STYLE | TRAVEL | JOBS | REAL ESTATE | AUTOS

The Lede

The New York Times News Blog

June 2, 2009, 7:05 PM

China's Great Firewall Blocks Twitter

By ROBERT MACKEY

Catherine Henriette/Agence France-Presse — Getty Images

Search This Blog
Search

Previous Post:
Bloggers Ponder Last Message From Missing Jet's Computer

Next Post:
Punditry From Bin Laden and Zawahiri on Obama's Trip to the Middle East

Recent Posts

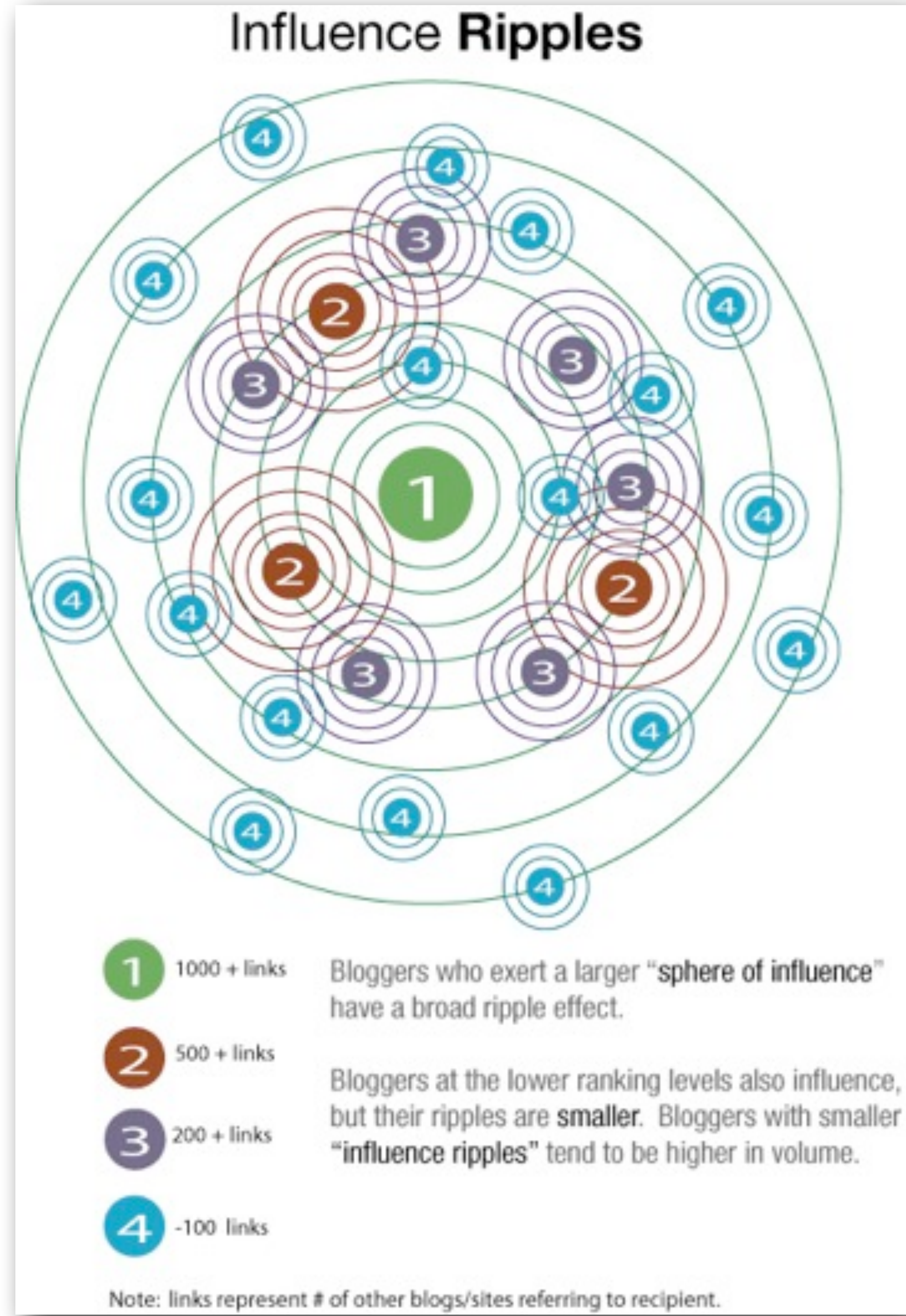
June 18
(38 comments)
Latest Updates on Iran's Disputed Election
To supplement reporting from New York Times correspondents inside Iran on Thursday, The Lede will continue to track the aftermath of Iran's disputed presidential election online.

June 17
(129 comments)
Wednesday: Latest Updates on Iran's Disputed Election
On Wednesday, The Lede will continue to track the aftermath of Iran's disputed presidential election online, to supplement reporting from New York Times correspondents inside Iran.

June 16
(198 comments)
Tuesday: Latest Updates on Iran's Disputed Election
To supplement reporting from New York Times correspondents inside Iran, The Lede

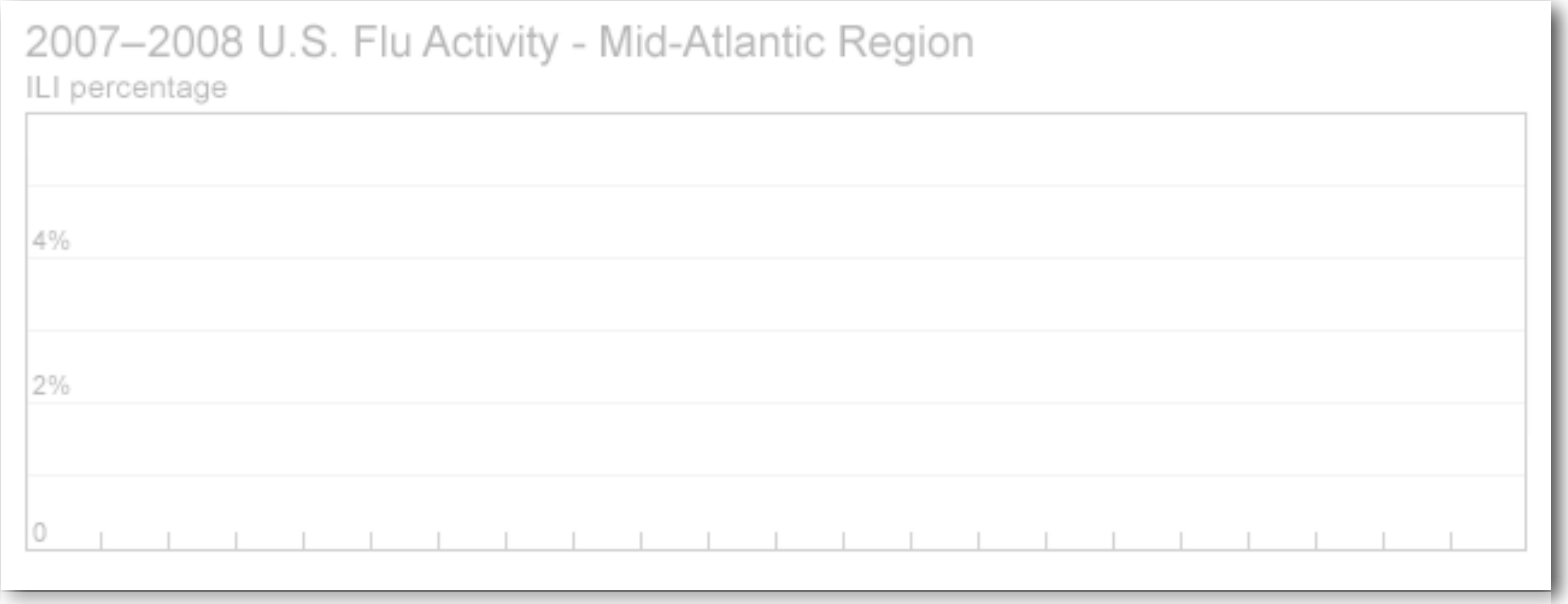
Commerce

- Social marketing
- Who are the brokers?
- Who can exert the most influence on buying/selling?
- How much should one advertise?



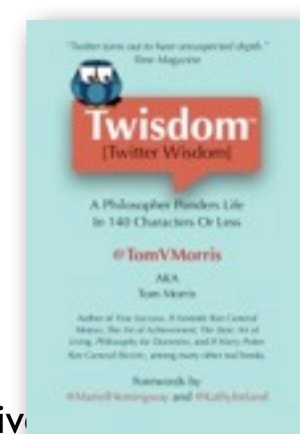
Public Health

- People's behavior can be monitored
- What is on people's mind translates to search queries
- Google predicts flu trends...



Pop Culture

- Twisdom: Twitter Wisdom
  - A Philosopher Ponders Life in 140 Characters or Less
    - "I don't know the key to success, but the key to failure is trying to please everybody." Bill Cosby Do what you know in your soul is right!
    - It is a miserable state of mind to have few things to desire, and many things to fear. – Francis Bacon
- The Longest Poem In the World-the awesome twitter poem! 956,644 verses this morning and ~4,000 a day!



The Social Media Generation

The Age of FaceBook

Outline

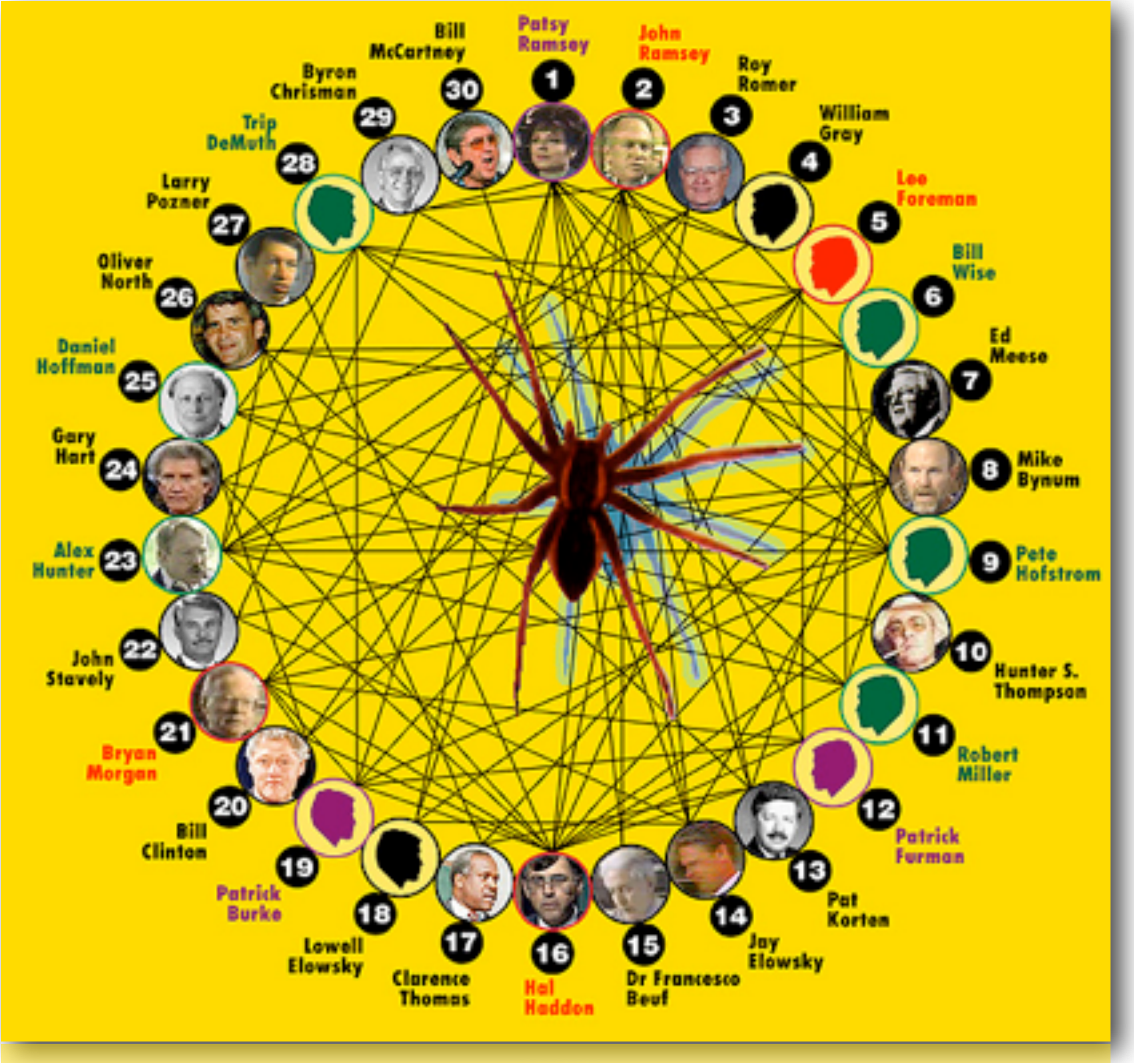
- Introduction to Social Computing
- Social Network Theory
- Graph mining
- Ranking and Link Analysis
- Recommender Systems
- Human Computation
- Opinion Mining/Sentiment Analysis
- Opinion mining and sentiment analysis
- Social Computing in Education
- Social Monetization
- and possibly more...

Web 2.0

- Web as a medium vs. **Web as a platform**
- Read-Only Web vs. **Read-and-Write Web**
- Static vs. **Dynamic**
- Restrictive vs. **Freedom & Empowerment**
- Technology-centric vs. **User-centric**
- Limited vs. **Rich User Experience**
- Individualistic vs. **Group/Collective Behavior**
- Consumer vs. **Producer**
- Transactional vs. **Relational**
- Top-down vs. **Bottom-up**
- People-to-Machine vs. **People-to-People**
- Search & browse vs. **Publish & Subscribe**
- Closed application vs. **Service-oriented Services**
- Functionality vs. **Utility**
- Data vs. **Value**

Social Networks

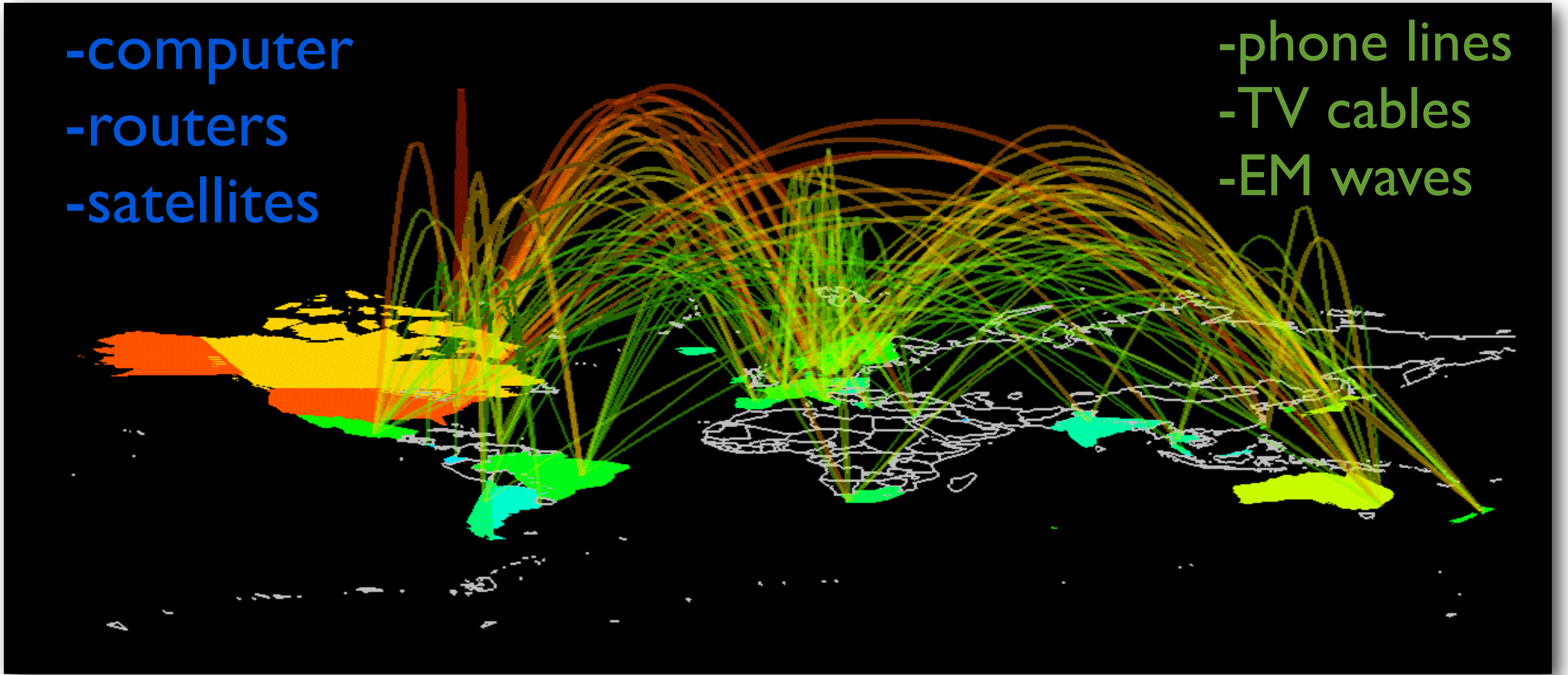
Society:
Nodes: individuals
Links: social relationship (family/work/friendship/etc.)



S. Milgram and John Guare: Six Degree of Separation.
Social networks: Many individuals with diverse social interactions between them.

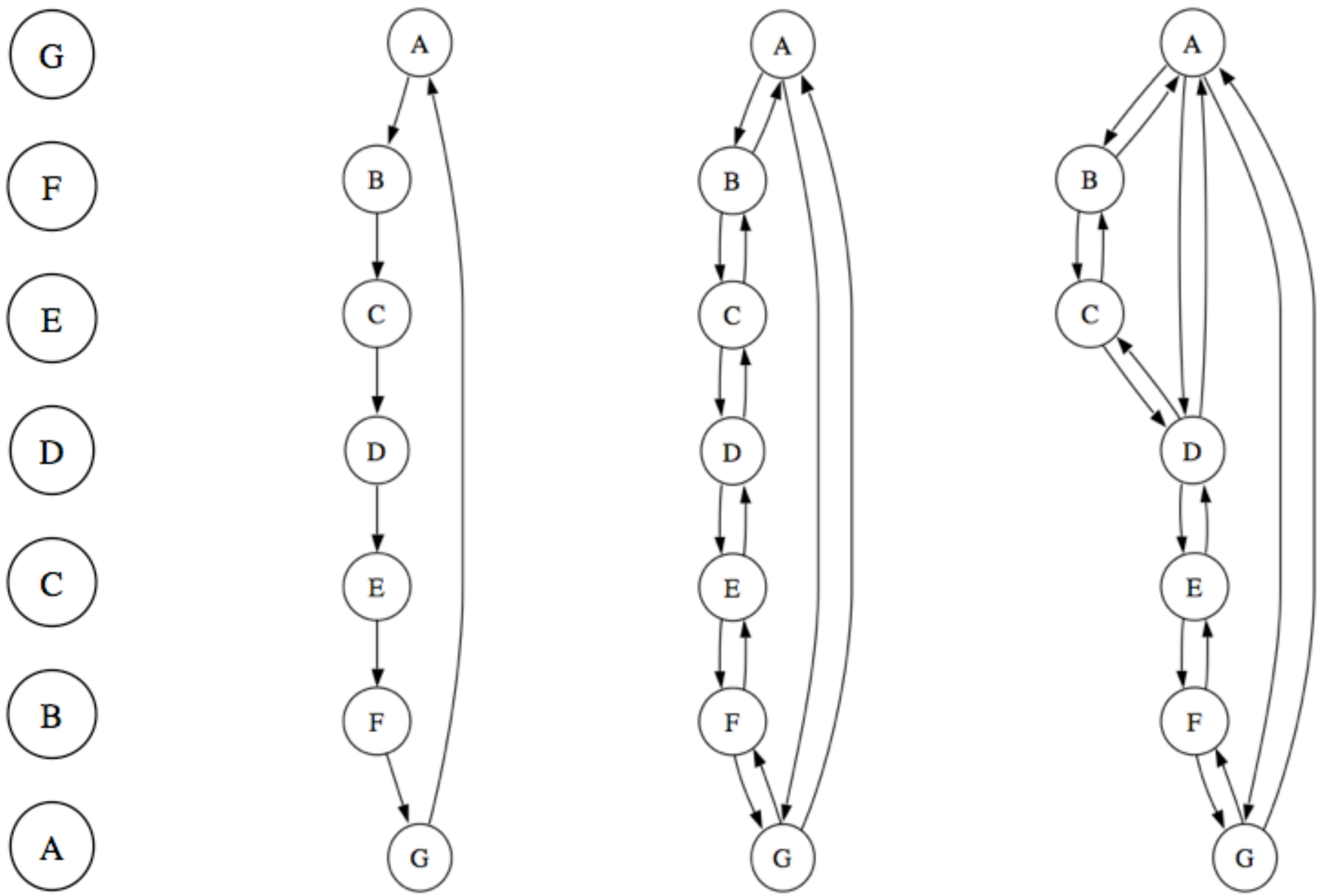
Social Networks

- The Earth is developing an electronic nervous system, a network with diverse nodes and links.

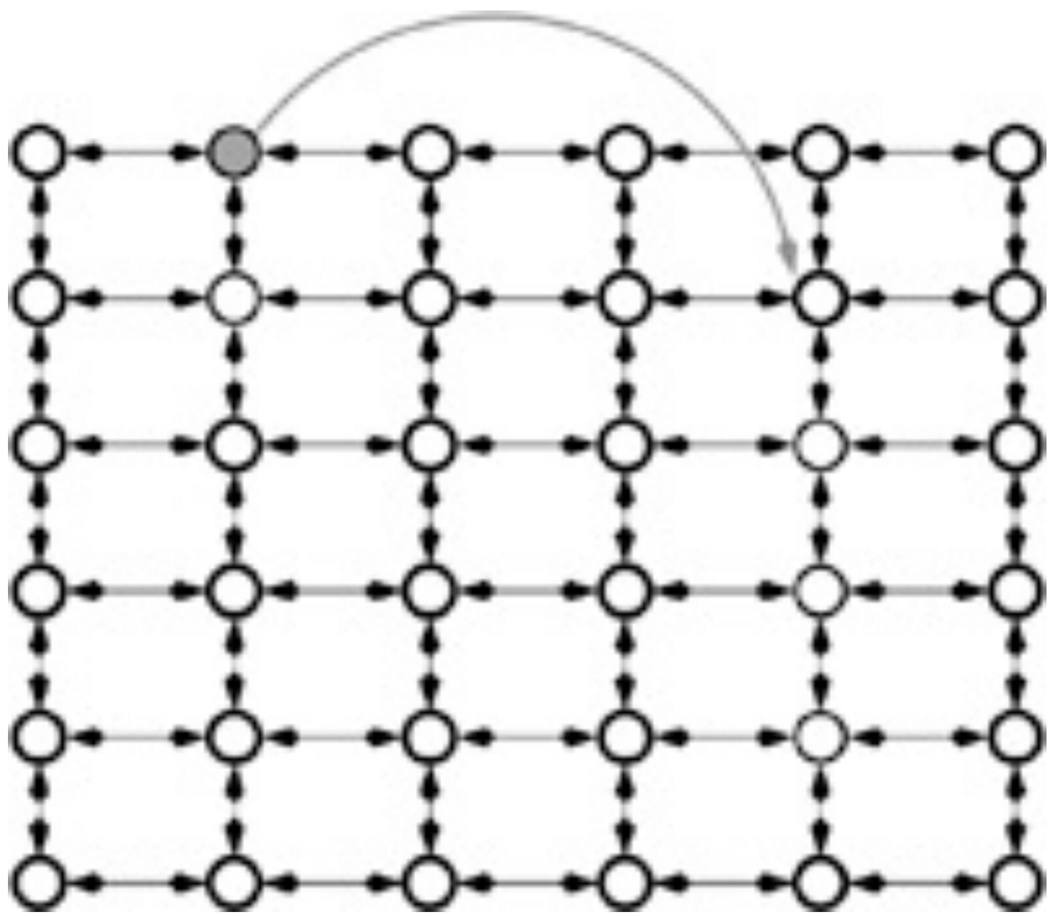


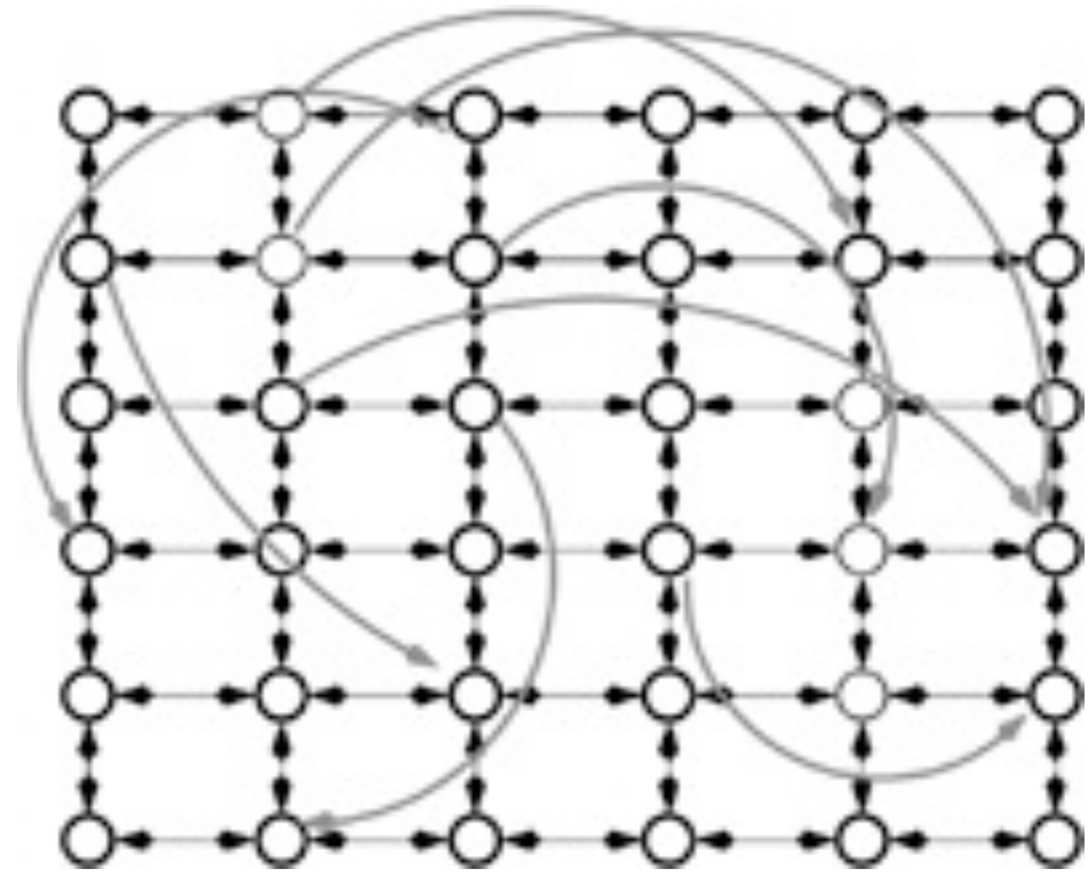
Communication networks: many non-identical components with diverse connections between them.

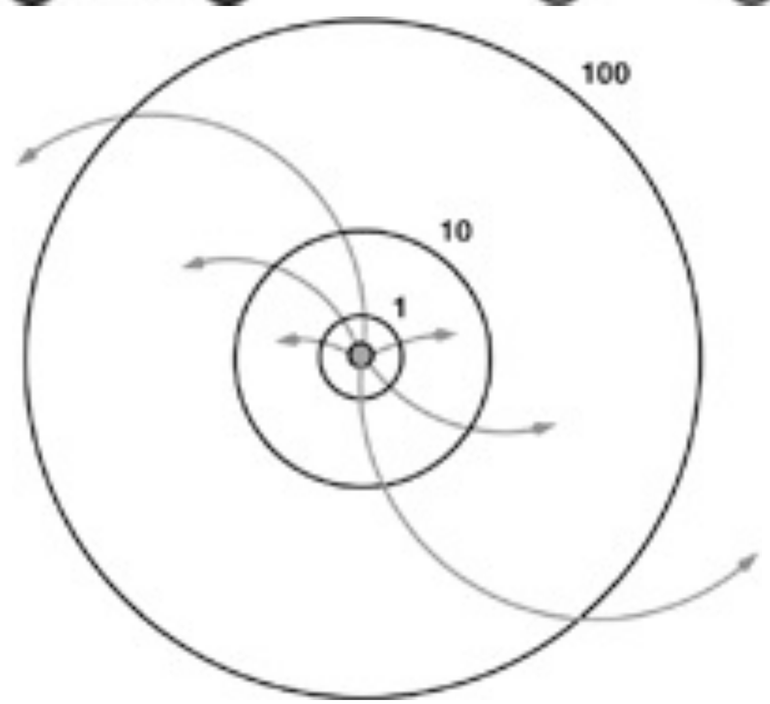
The Flow of Information



Examples

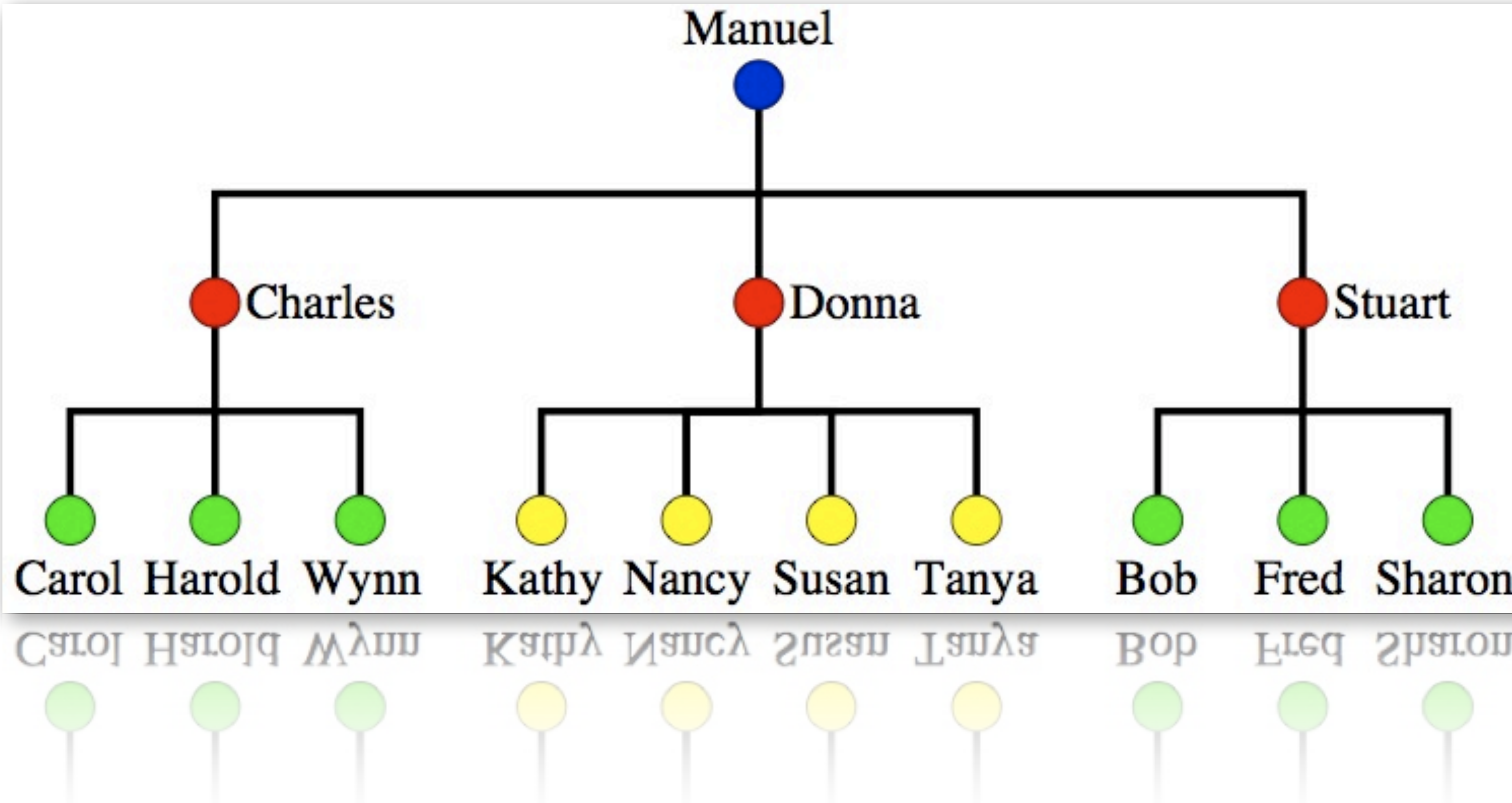






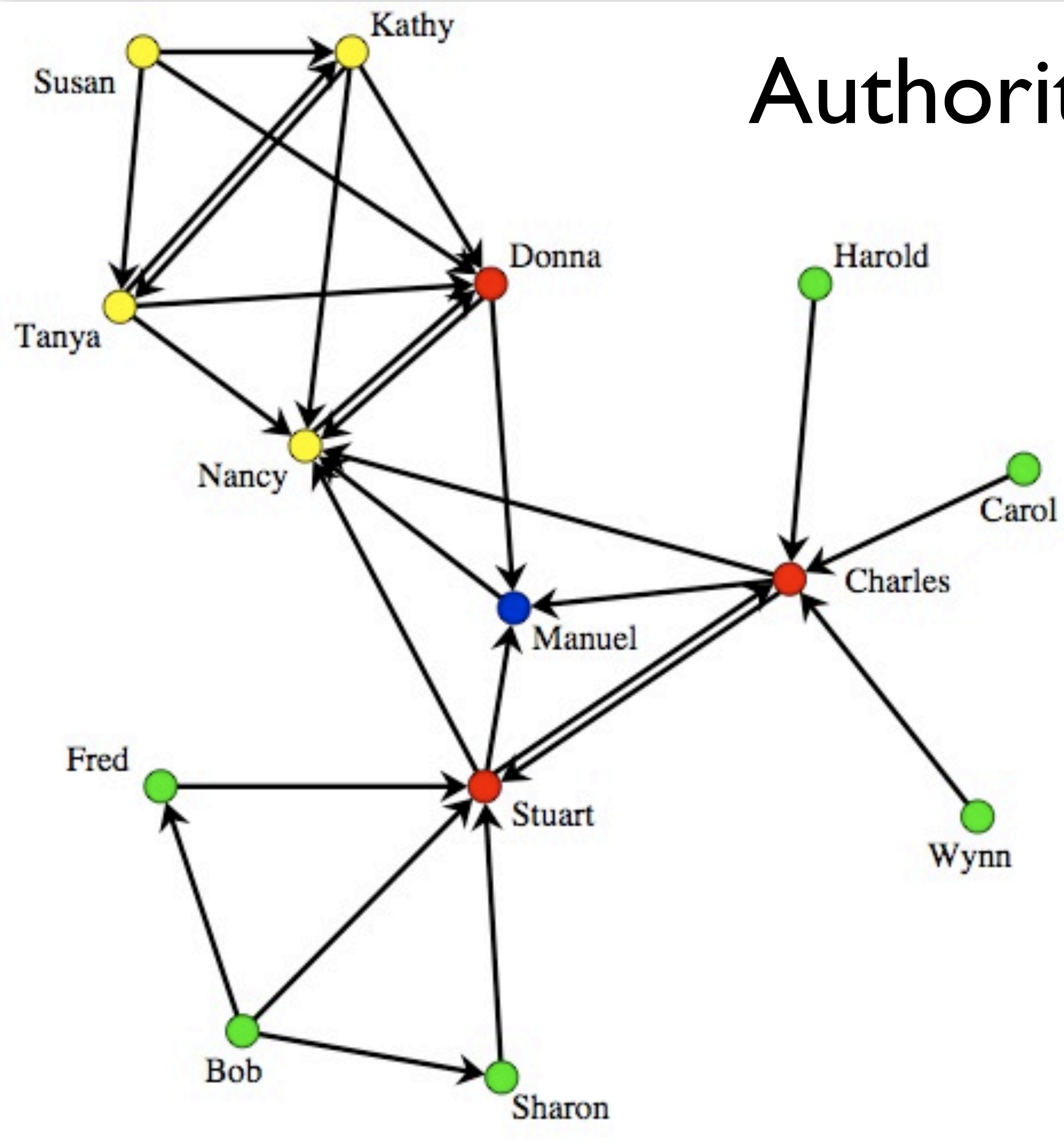
[Kleinberg 1999]

Organizational Chart



Social Network Chart

Authority vs. Importance



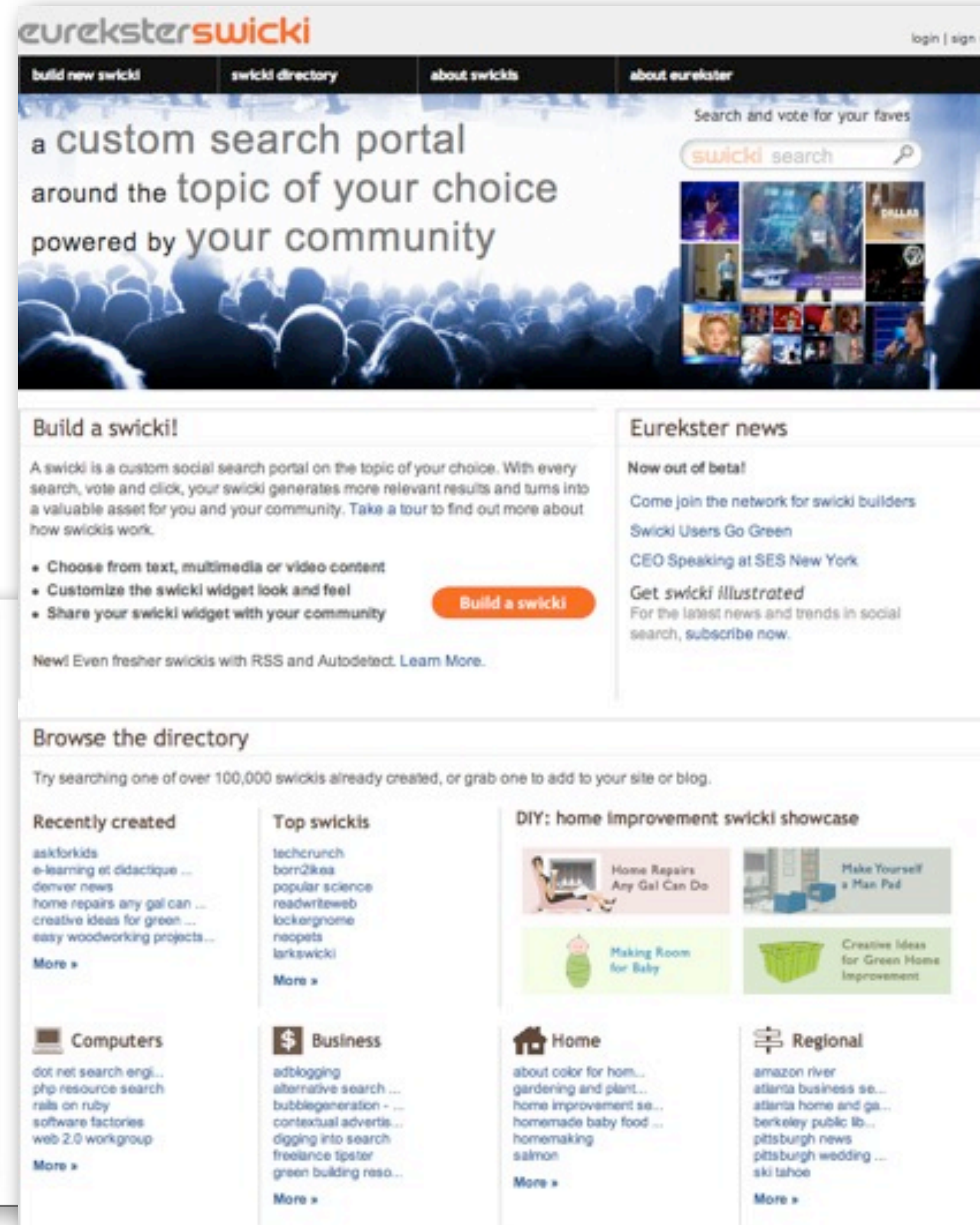
1.00	Nancy	(secretary)
0.66	Donna	(supervisor)
0.57	Manuel	(manager)
0.19	Stuart	(supervisor)
0.17	Charles	(supervisor)
0.08	Kathy	(secretary)
	Tanya	(secretary)
0.02	Fred	(auditor)
	Sharon	(auditor)
0.00	Bob	(auditor)
	Carol	(auditor)
	Harold	(auditor)
	Wynn	(auditor)
	Susan	(secretary)

Social Networking Sites

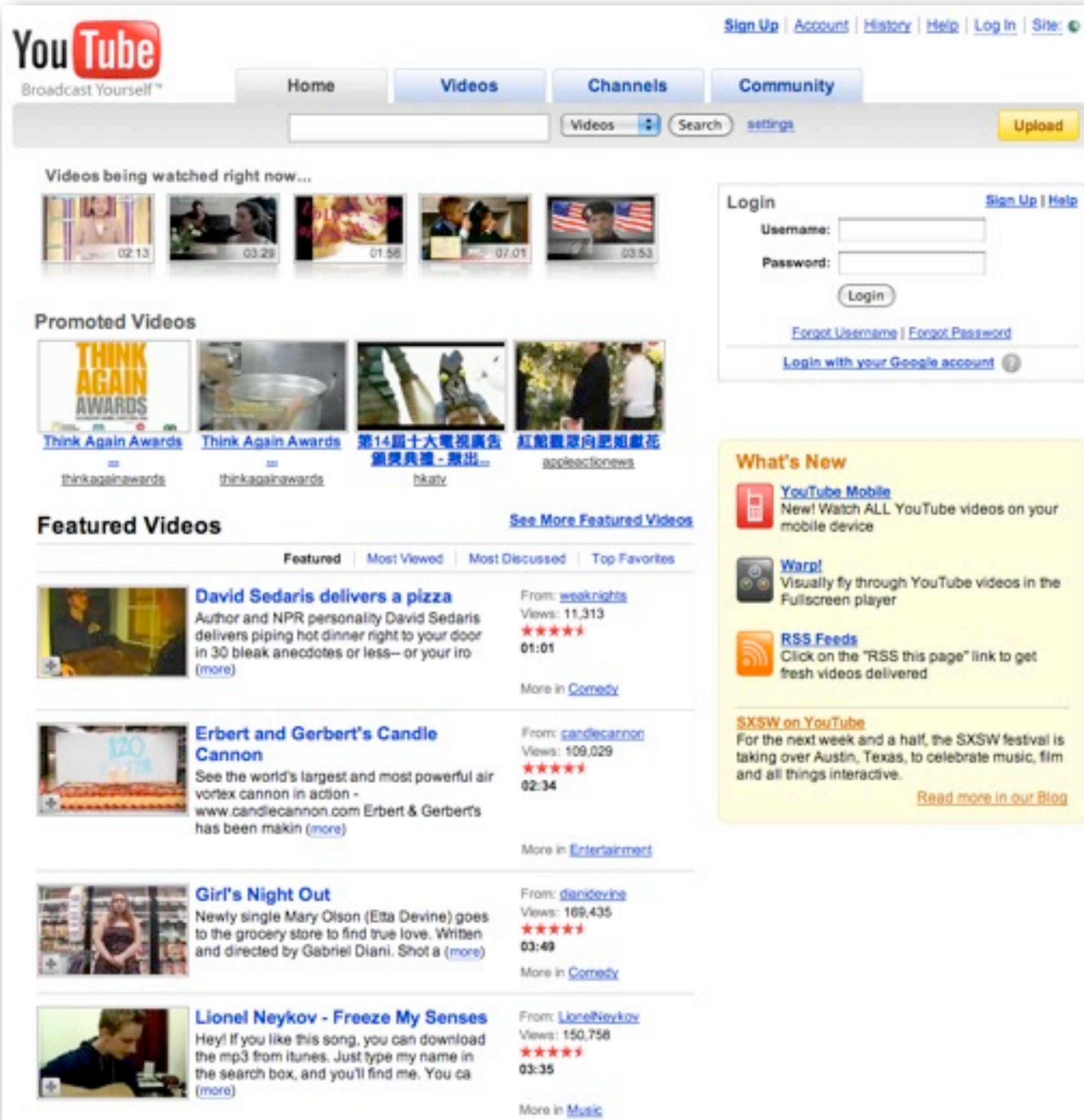
- Example of Social Networking Sites: FaceBook, MySpace, Blogger, QQ, etc.

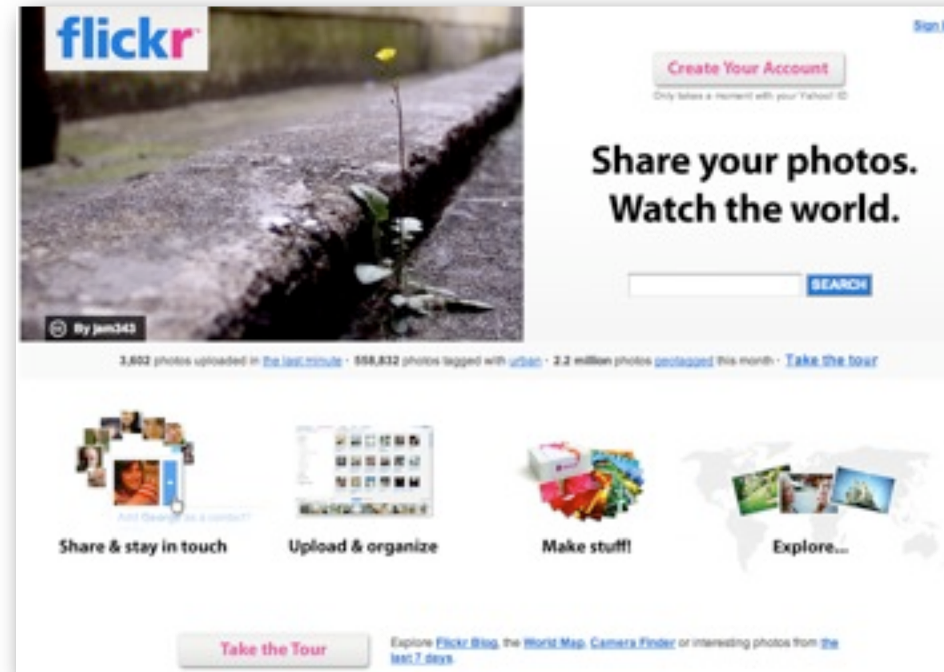
Social Search

- Social Search Engine
- Leveraging your social networks for searching



Social Media





Social News/Mash Up

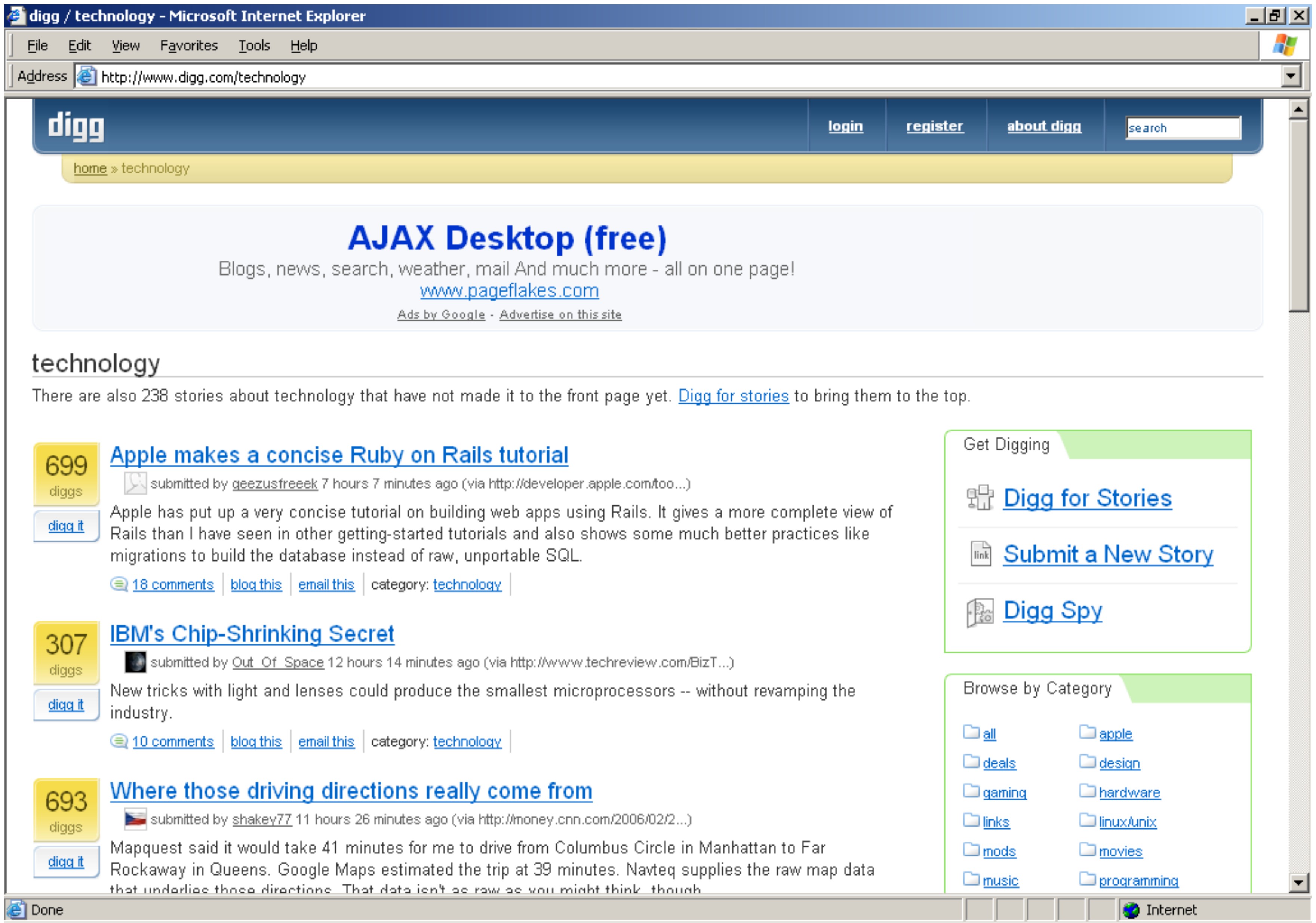
Social Knowledge Sharing

Social News and Knowledge Sharing

- Social news refers to websites where users can submit their owe information. Users can also vote on news or other links to determine which links are presented

Wikis allow collaborative publishing.
Anyone can create new articles or edit existing articles.
All versions are kept.

Social News and Knowledge Sharing



- Digg.com members "vote" for stories to appear on the home page

Social News and Knowledge Sharing

- The notion that each individual contributes to a collective pool of knowledge is further realized in AnswerBus, Webclopedia, Yahoo's babelfish, etc.

Social News and Knowledge Sharing

- Question and answering

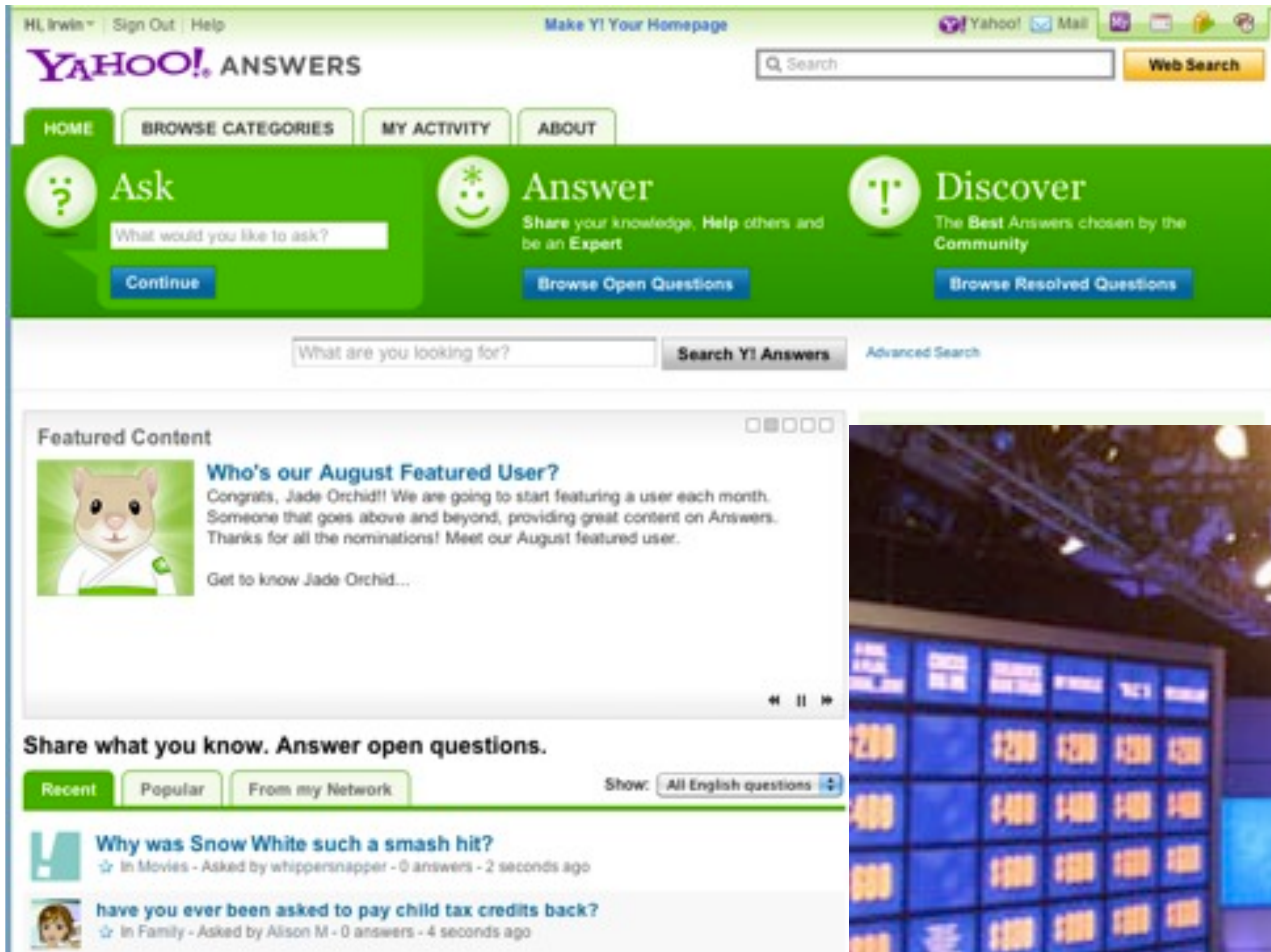


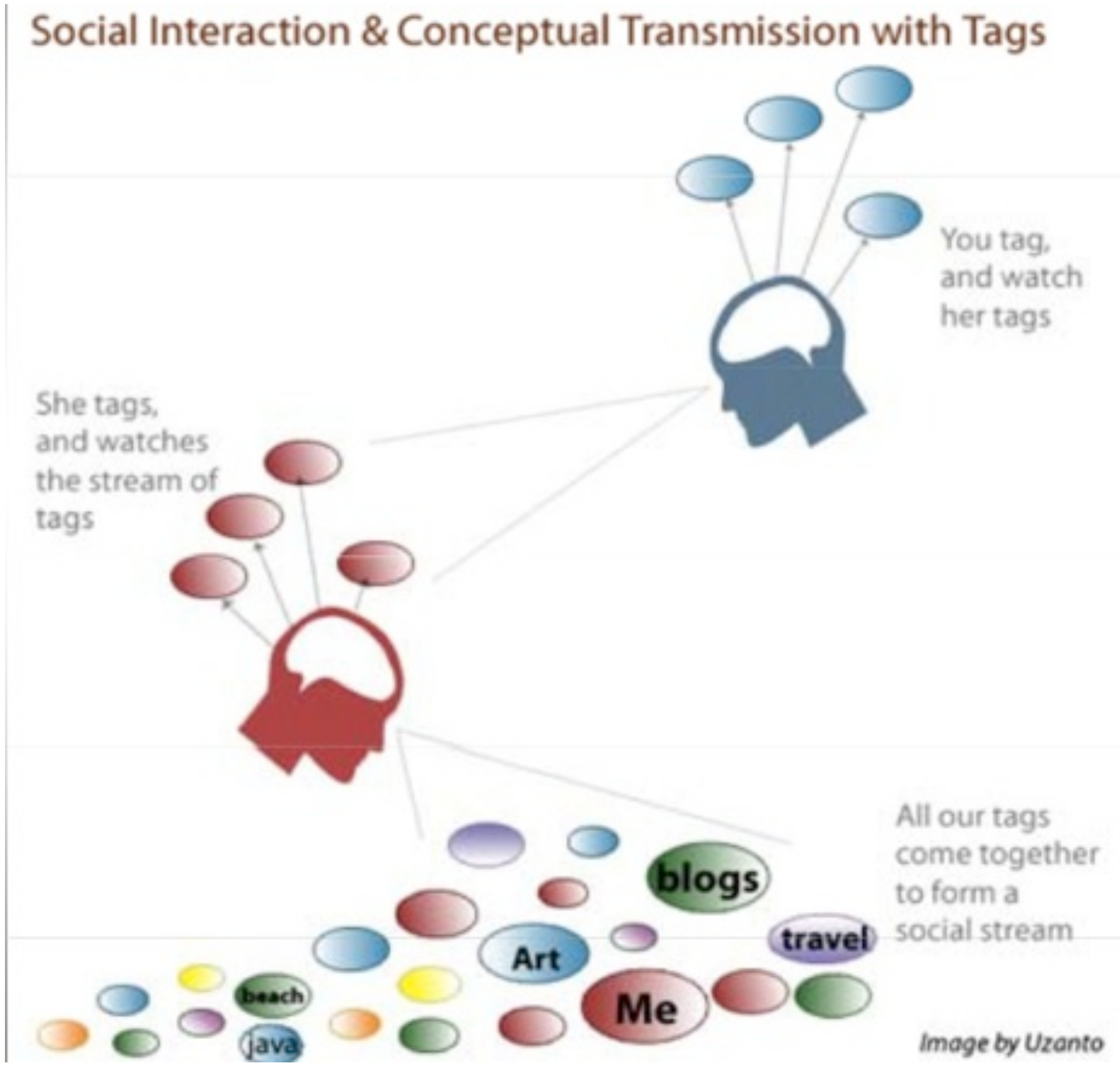
PHOTO: JULIANNE PEPITONE/CNNMONEY

Social Bookmarking

- What is a tag?
  - Descriptive metadata
  - A keyword or term associated with or assigned to a piece of information
  - User defined, created and shared
  - Many web users do it every day, with very little conscious awareness that they are "cataloging"
- What gets tagged?
  - Pictures, blog posts, video clips, catalog entries, just about anything...

Social Bookmarking

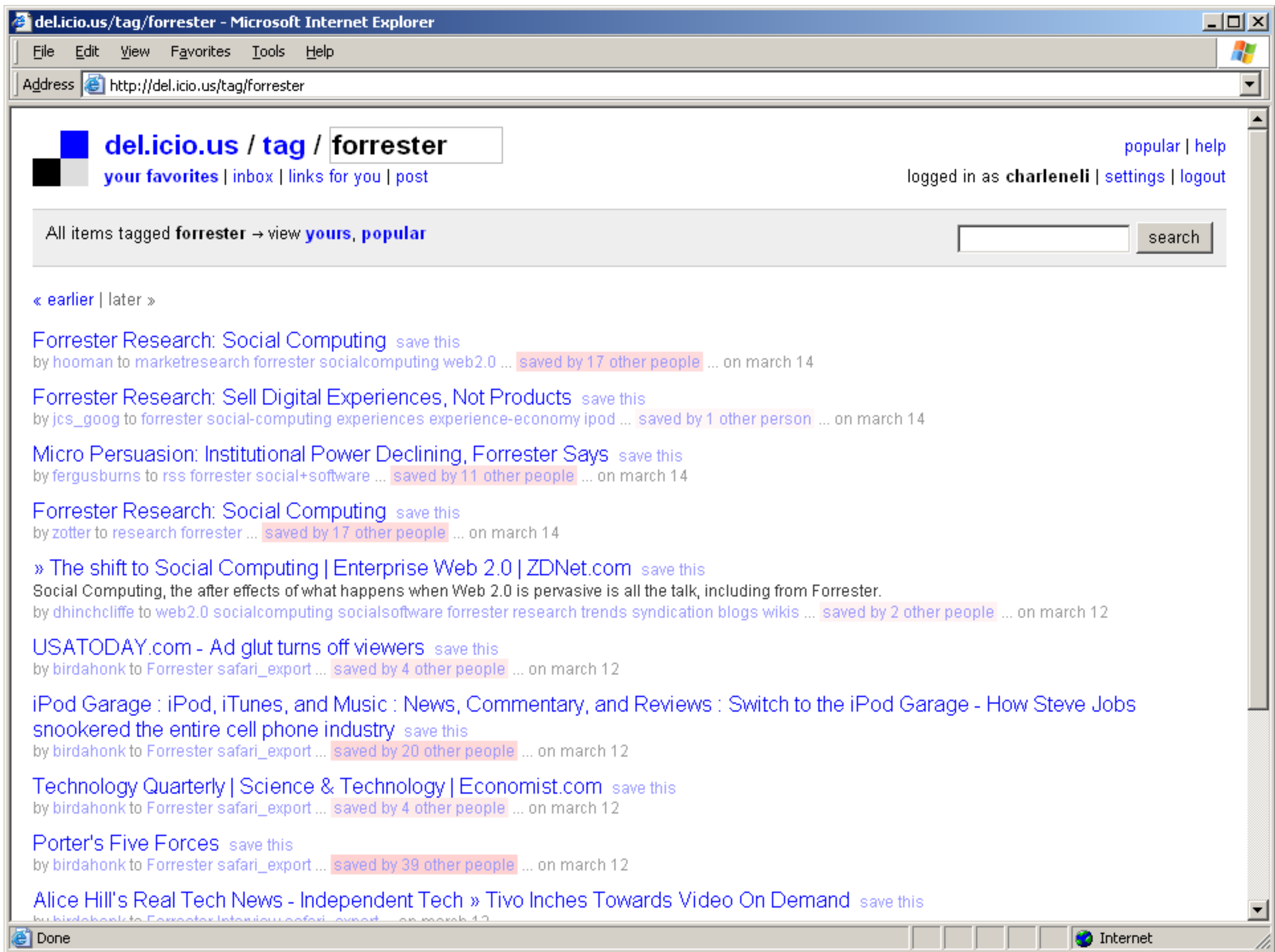
- Share one's tags
- Make the individual browsing experience a social one



Why users tag?

- Tagging means something specific to the user
- It is easy -- anyone can do it
- Finding things on the Internet
- Serendipitous discovery
- It is social
- New ways to share and discover

Social Bookmarking in del.icio.us



Social Bookmarking in StumbleUpon

- StumbleUpon allows users to discover and rate web pages, photos, and videos. It chooses which web page to display based on the user's ratings of previous pages, ratings by his/her friends, and by the ratings of users with similar interests.

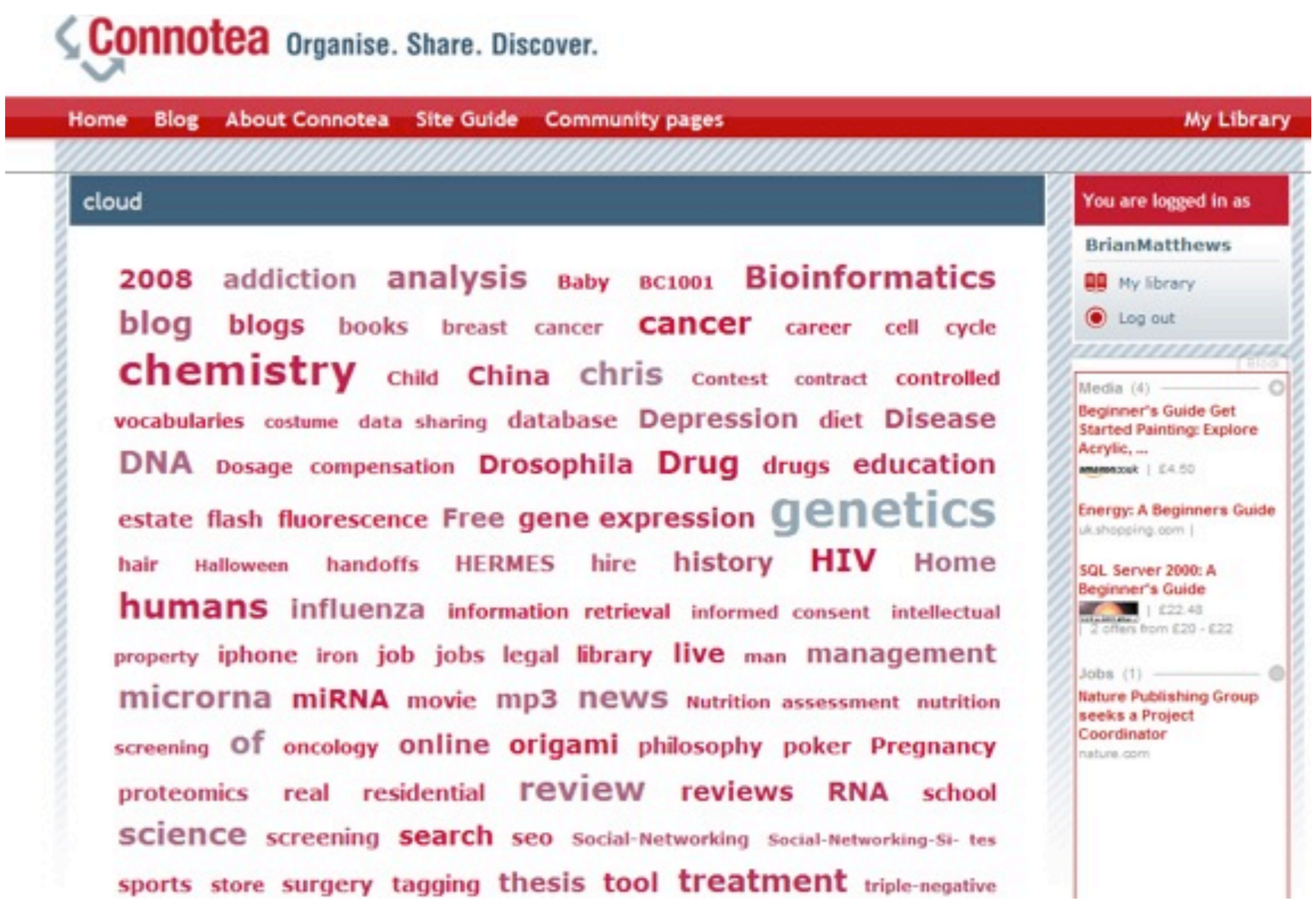
Tagging is Everywhere

Folksonomies

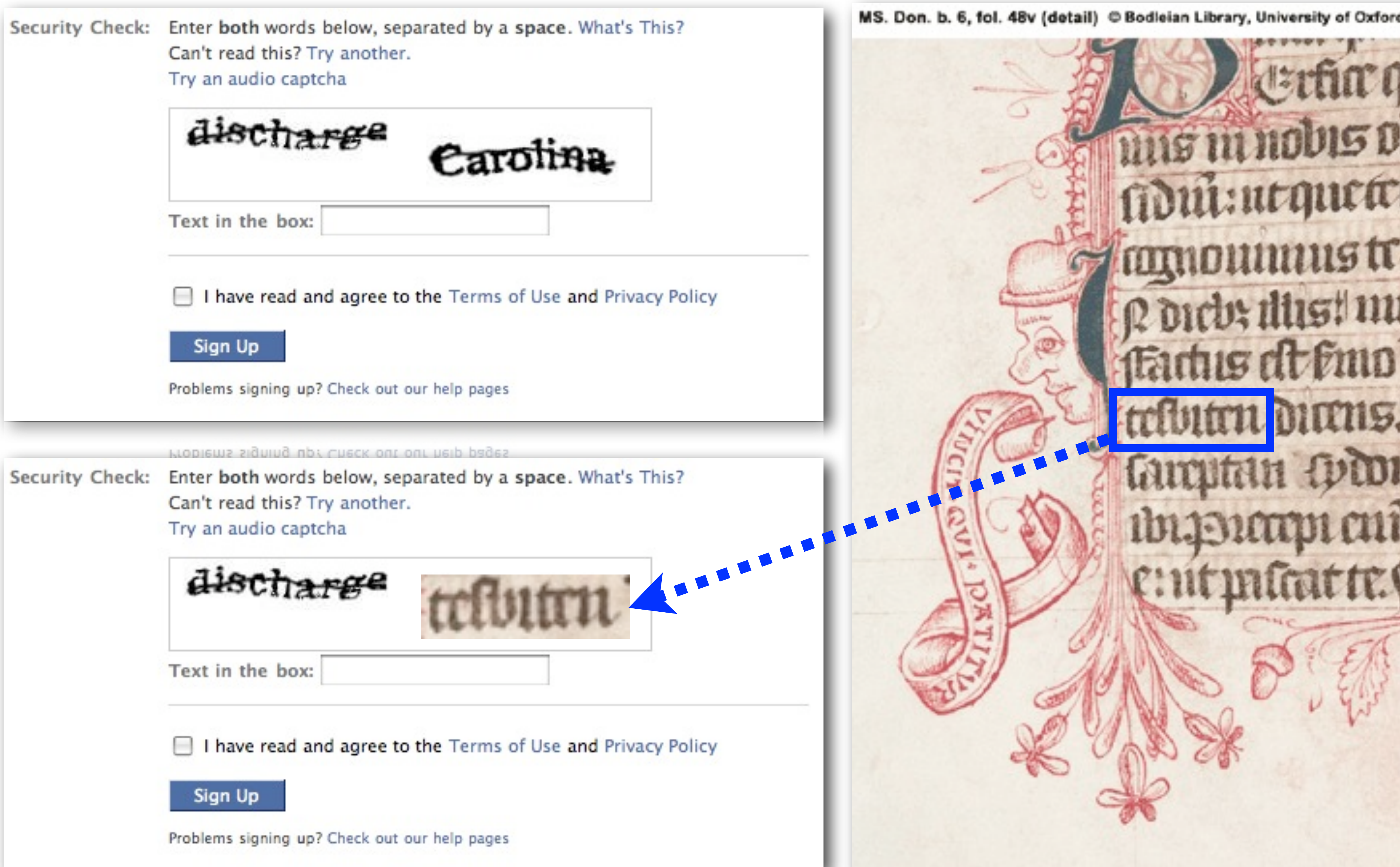
- Folksonomies are the actual output result of collaborative tagging
- Literally, it is taxonomy by "folks"
- Grass-roots
- Community based
- Inclusive -- everyone can get involved
- Scalability

Tag Clouds

- Visualization of tags
- Weighted value -- based on size, frequency of use of tag

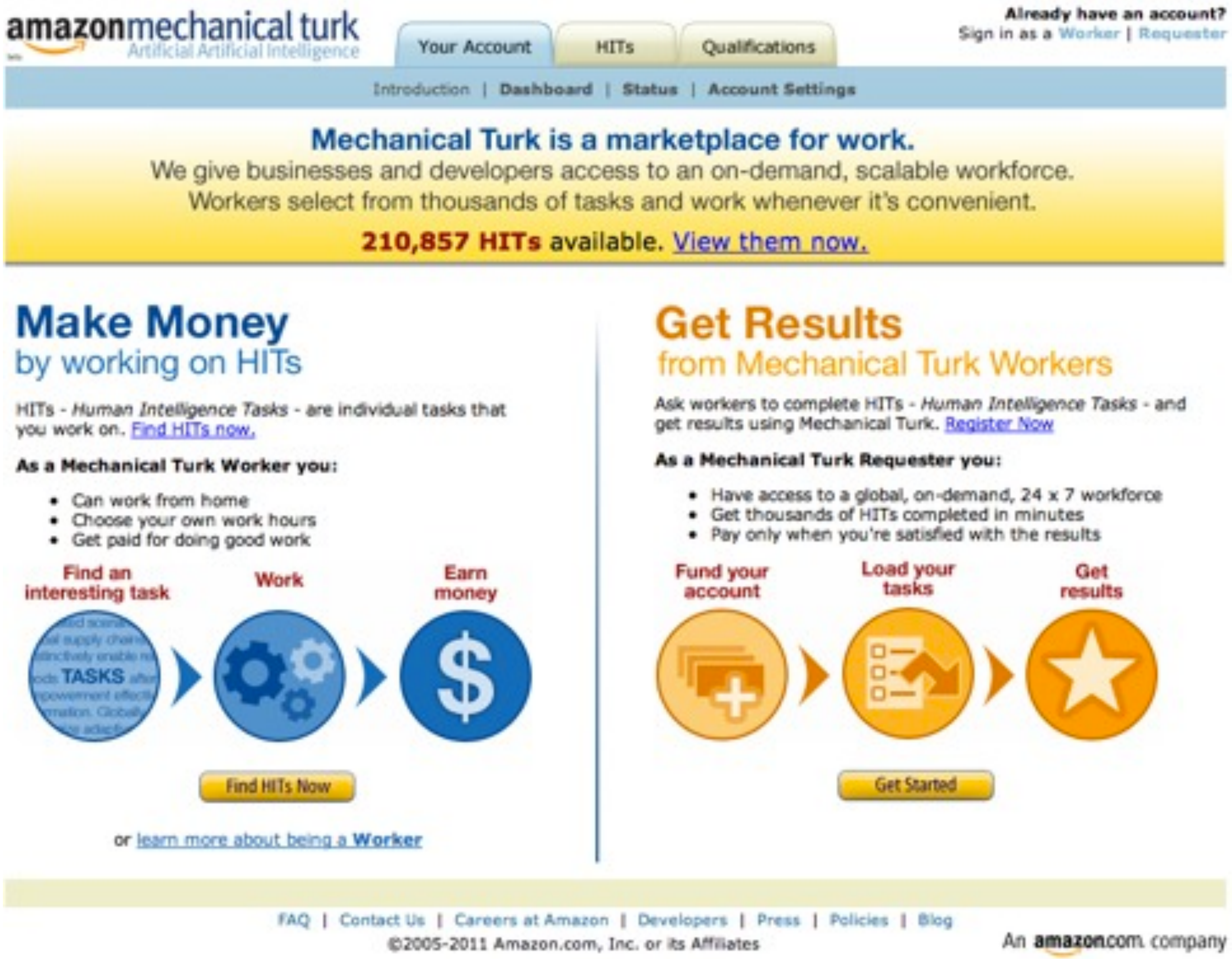


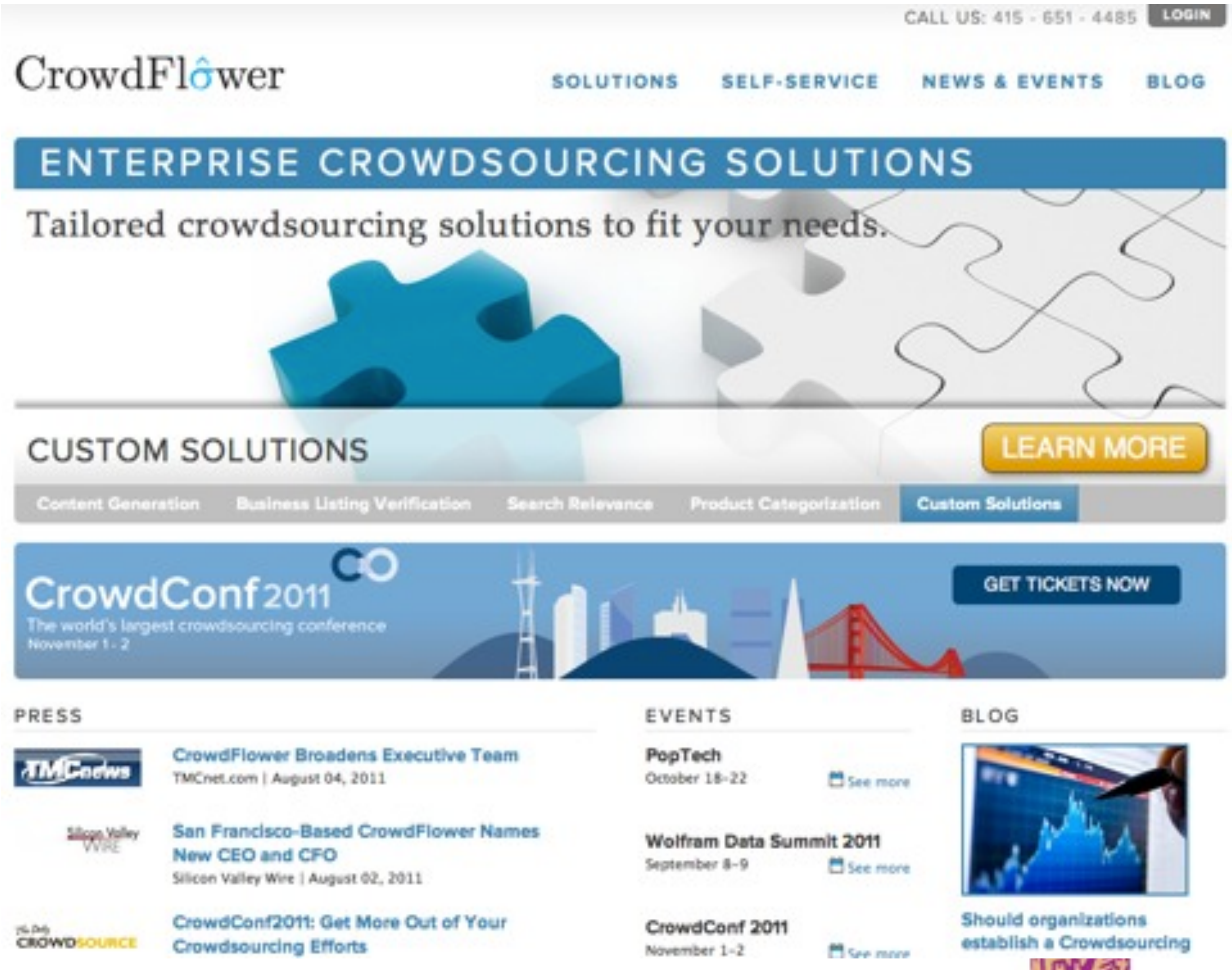
Social/Human Computation



Human Computation

Crowdsourcing





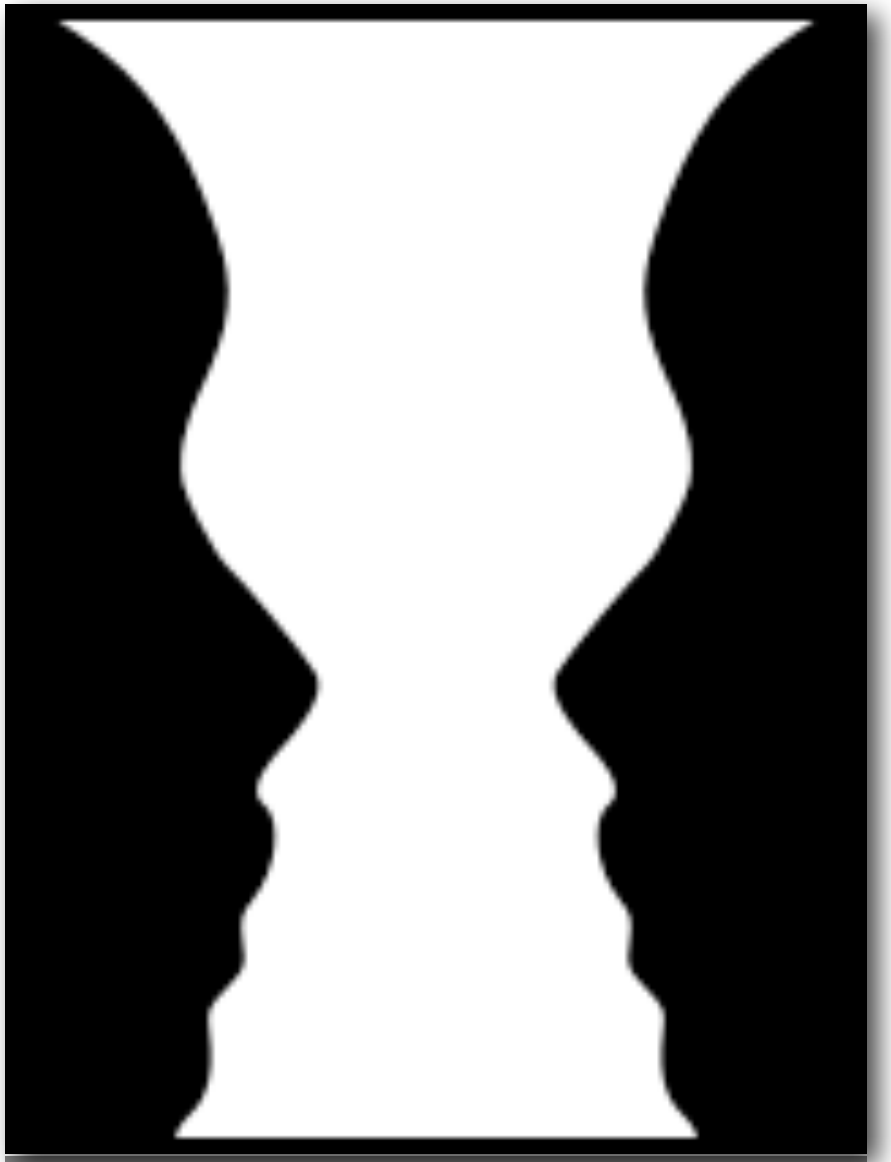
Web 2.0 Revolution

- Glocalization-think globally and act locally!
- Weblication-Web is the application!
- Three C's

  Connectivity

  Collaboration

  Communities



Social Relations

presence

crew

identity

binary

teams

social role

populations

squad

cardinal

reputation

organizations

expertise

integer

trust

cohorts

markets

ownership

real

communities

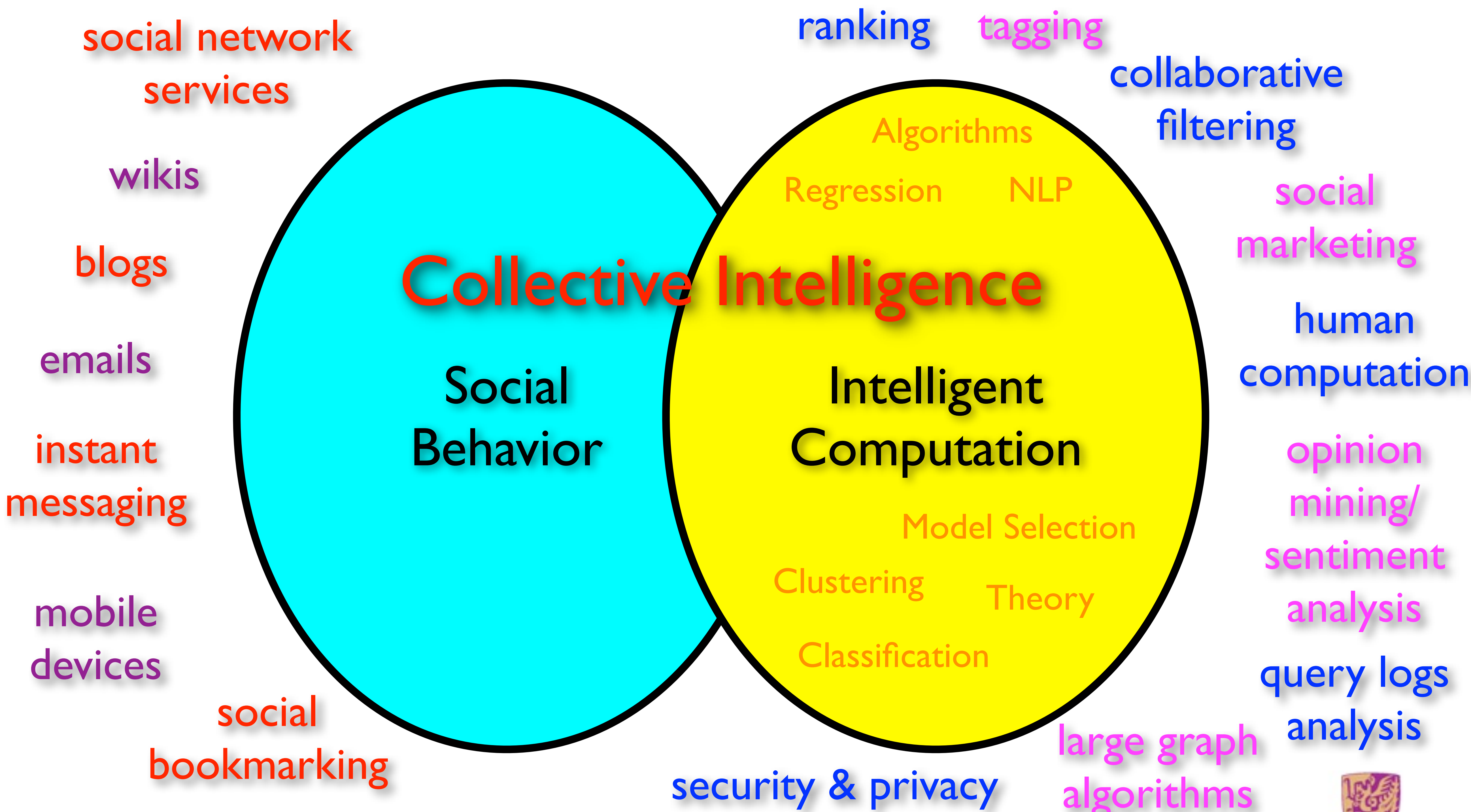
accountability

partners

knowledge

groups

Social Computing



Definition of Social Computing

[wiki]

- Any Computer-mediated communication and interaction
- In the weaker sense: supporting any sort of social behavior
  - blogs, email, instant messaging, wiki, social network services, social bookmarking
- In the stronger sense: supporting "computations" that are carried out by a group of people
  - collaborative filtering, online auctions, prediction markets, reputation systems, tagging, verification games

Emerging Issues

- Theory and models
- Seach, mining, and ranking of existing information, e.g., spatial (relations) and temporal (time) domains
  - Dealing with partial and incomplete information, e.g., collaborative filtering, ranking, tagging, etc.
- Scalability and algorithmic issues
- Security and privacy issues
- Monetization of social interactions

Social Network Theory

- Consider many kinds of networks:
    - social, technological, business, economic, content, ...
- These networks tend to share certain informal properties:
    - large scale; continual growth
    - distributed, organic growth: vertices "decide" who to link to
    - interaction restricted to links
    - mixture of local and long-distance connections
    - abstract notions of distance: geographical, content, social,…

Six Degree of Separation

- "Six degrees of separation between us and everyone else on this planet" [John Guare, 1990]
- What is the probability of two strangers having a mutual friend?
- What is the chain of intermediaries between two strangers?

Small World Networks

- A network that most nodes can be reached from every other node by a small number of steps
- $L \propto \log N$, where *L* is the steps and *N* is the network size
- Examples: road, power grid, online social networks, email network, neural networks, WWW, etc.

Dunbar's Number

- It is theoretical cognitive limit to the number of people with whom one can maintain stable social relationships
- It is assumed to be between 100 to 230 with 150 as the norm in various studies
- Allen curve--the exponential drop of frequency of communication as the distance between them increases

The Tipping Point

- It is "the moment of critical mass, the threshold, the boiling point"
- Three rules
  - The Law of the Few
  - The Stickiness Factor
    - ...the specific content of a message that renders its impact memorable, i.e., Apple's 1984 Super Bowl commercial
  - The Power of Context
    - ...are sensitive to the conditions and circumstances of the times and places in which they occur

Pareto Principle

- Also known as the "80-20 Rule" or "The Law of the Vital Few"
- Roughly 80% of the effects come from 20% of the causes
  - In 1906, the observation that 80% of the land in Italy was owned by 20% of the population
  - 80% of your profits come from 20% of your customers
  - 80% of your complaints come from 20% of your customers
  - 80% of your sales are made by 20% of your sales staff
  - 80% of the work will be done by 20% of the participants

Social Network Theory

- Do these networks share more quantitative universals?
- What would these "universals" be?
- How can we make them precise and measure them?
- How can we explain their universality?
- This is the domain of social network theory

Some Interesting Quantities

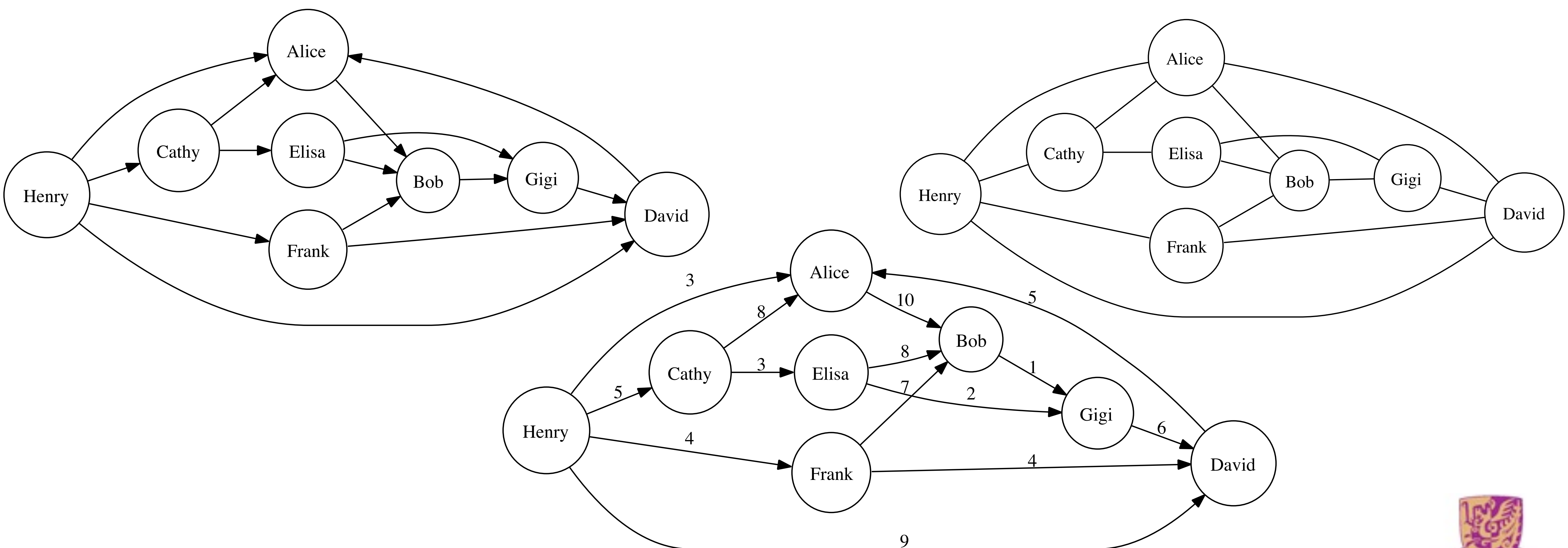
- Connected components
  - how many, and how large?
- Network diameter
  - maximum (worst-case) or average?
  - exclude infinite distances? (disconnected components)
  - the small-world phenomenon
- Clustering
  - to what extent that links tend to cluster "locally"?
  - what is the balance between local and long-distance connections?
  - what roles do the two types of links play?
- Degree distribution
  - what is the typical degree in the network?
  - what is the overall distribution?

Types of Relations

- **Kinship**–mother of, wife of
- **Other role-based**–boss of, teacher of, friend of, brother of, father of, sister of, enemy of, lover of
- **Cognitive/perceptual**–knows, aware of what they know, is familiar with
- **Affective**–likes, loves, hates, admires, trusts
- **Interactive**–give advice, talks to, fights with, sex/drugs with, buys from, sells to
- **Affiliations**–belong to same clubs, is physically near
- **Derived**–has subscription to the same magazine as, is taller than, distance between
- **Flows**–moves to, flows to

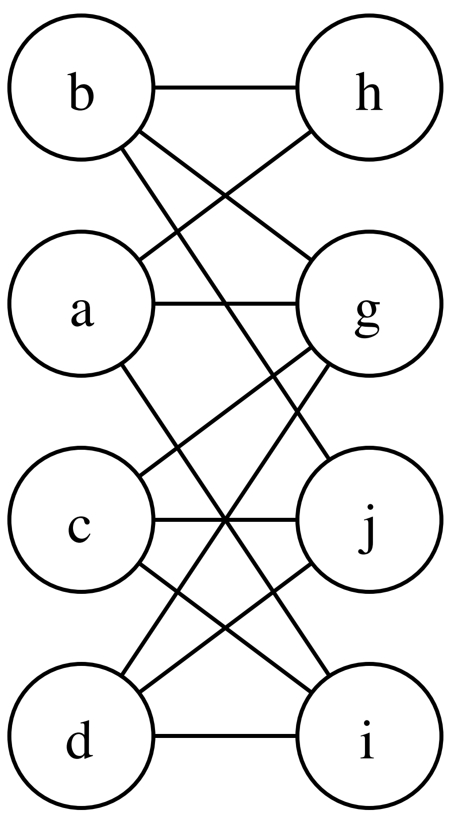
Define Graphs

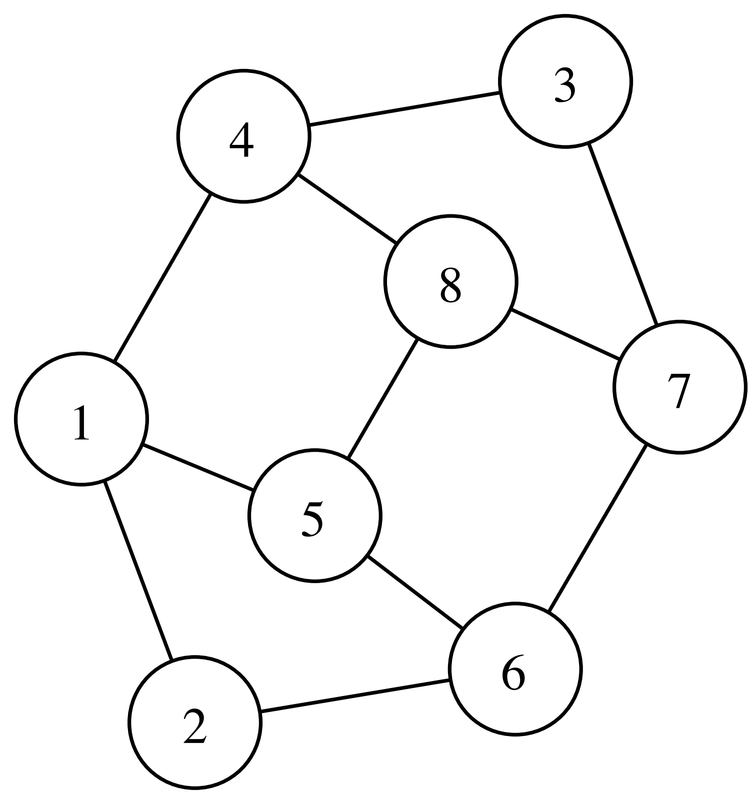
- A graph $G = (V, E)$ consists of a set of vertices, V, and a set of edges, E.
- Each edge is a pair (v, w), where $v, w \in V$. It is said to join the vertices v and w.
- If the edge $e = (v, w) \in E$, then u and v are both said to be *incident* with e and *adjacent* to each other.
- If the pair is ordered, then the graph is directed (digraphs).
- One can associate an attribute to the edge which is called *weight*.



Define Graph Isomorphism

- Two graphs G and H are said to be *isomorphic*, denoted by $G \sim H$, if there is a one-to-one correspondence, called an *isomorphism*, between the vertices of the graph such that two vertices are adjacent in G if and only if their corresponding vertices are adjacent in H.
- Likewise, a graph G is said to be *homomorphic* to a graph H if there is a mapping, called a homomorphism, from $V(G)$ to $V(H)$ such that if two vertices are adjacent in G then their corresponding vertices are adjacent in H.





a	1
b	6
c	8
d	3
g	5
h	2
i	4
j	7

Define Adjacency Matrix

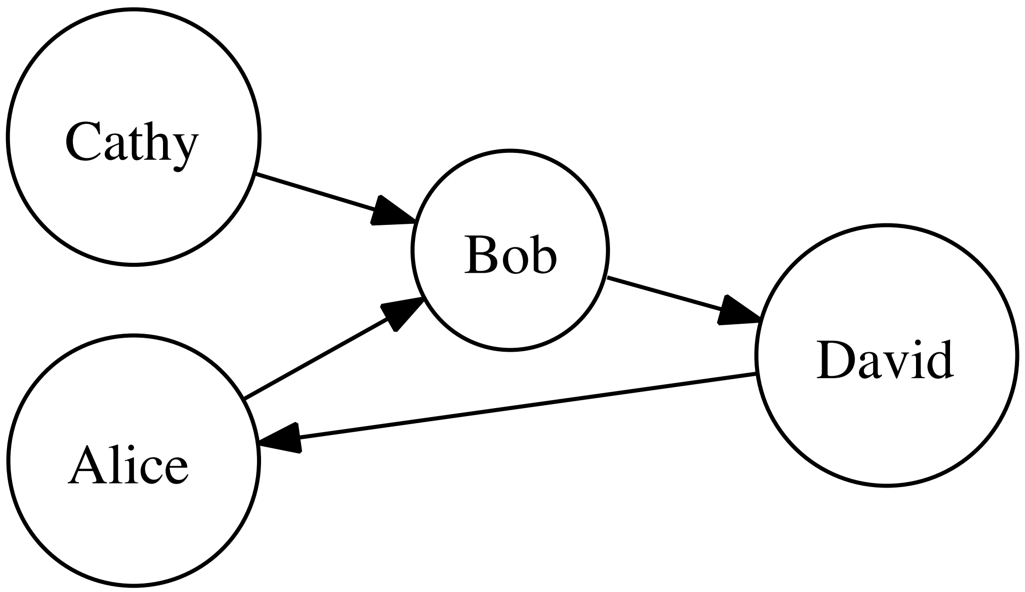
A graph G with n nodes can be represented by an n-by-n matrix. Given $V = \{v_1, v_2, \cdots, v_n\}$. Then the adjacency matrix A is an n-by-n matrix whose entry A_{ij} is defined to be:

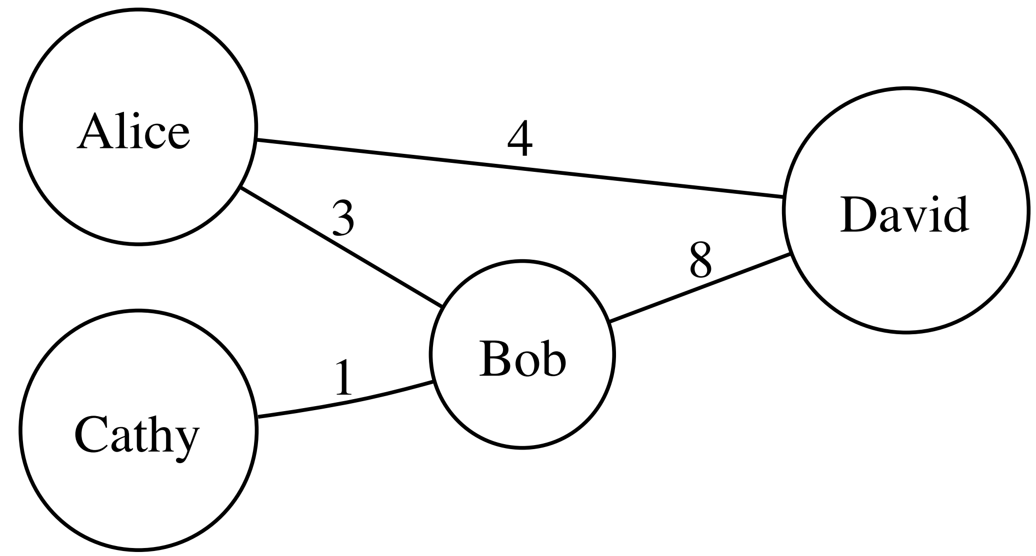
$$A_{ij} = \begin{cases} 1, & \text{if there is an edge from } v_i \text{ to } v_j \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Note that for the an undirected graph, $A_{ij} = A_{ji}$ so the adjacency matrix is a symmetric matrix. Moreover, we can put value attributes to the edges and define A_{ij} to be:

$$A_{ij} = \begin{cases} \text{w}, & \text{if there is an edge from } v_i \text{ to } v_j \text{ and } w \text{ is its weight} \\ 0, & \text{if there is no edge from } v_i \text{ to } v_j. \end{cases} \tag{2}$$

Examples of Adjacency Matrix





	Alice	*Bob*	*Cathy*	*David*
Alice	–	1	0	0
Bob	0	–	0	1
Cathy	0	1	–	0
David	1	0	0	–

	Alice	*Bob*	*Cathy*	*David*
Alice	–	3	0	4
Bob	3	–	1	8
Cathy	0	1	–	0
David	4	8	0	–

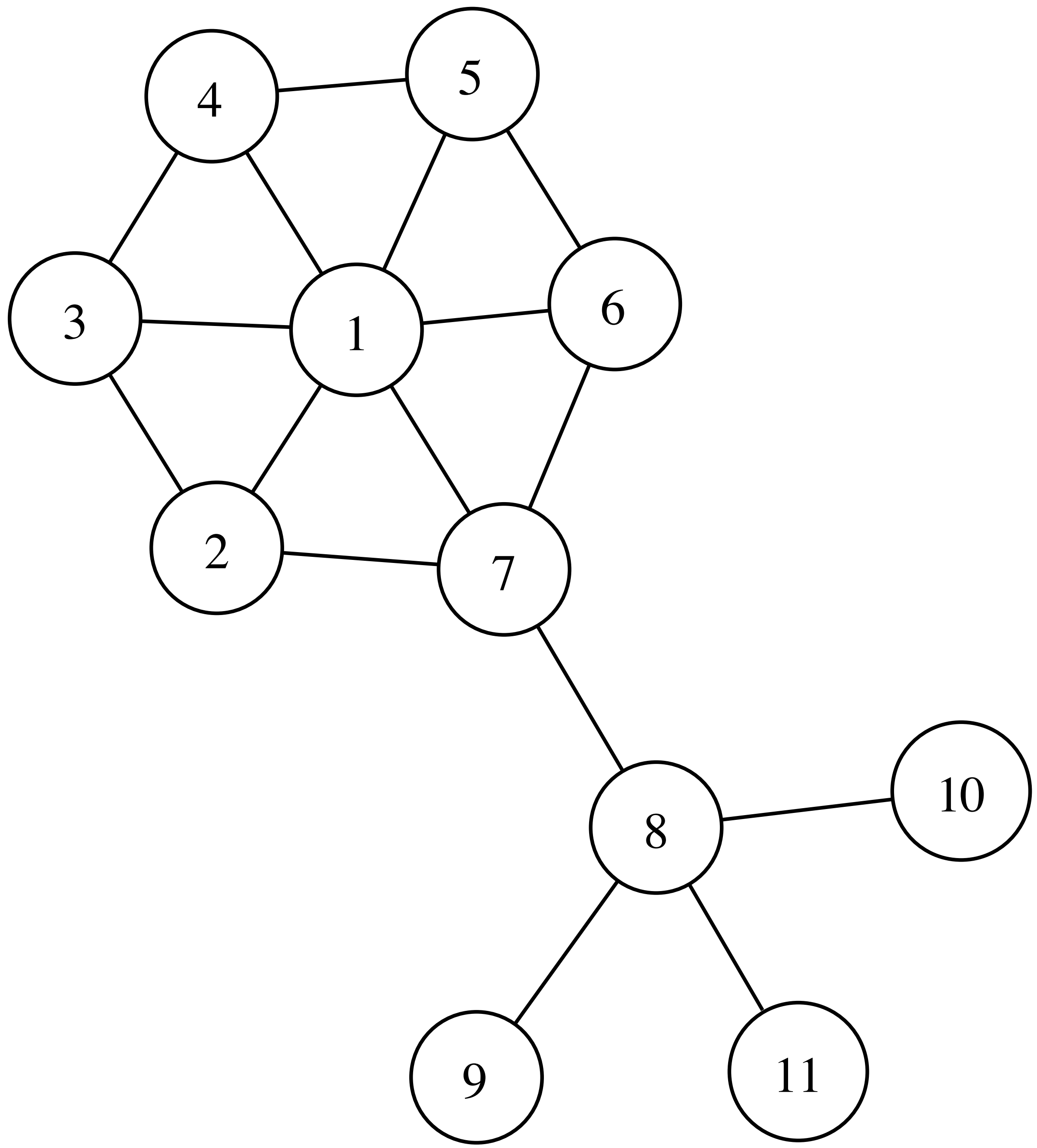
Define Length, Path, and Cycle

- A **path** p in G is a sequence of vertices $w_1, w_2, w_3, \cdots, w_N$ such that $(w_i, w_{i+1}) \in E$ for $1 \leq i \leq N$ and that $w_i \neq w_j$ with $i \neq j$.
- The length of p is the number of edges on the path, which is equal to $N - 1$.
- The length can be zero for the case of a single vertex.
- The distance between two nodes is the length of shortest path.
- A path with no repeated vertices is called a *simple path*.
- A cycle with no repeated vertices aside from the starting and ending vertex is a simple cycle. A simple cycle that includes every vertex of the graph is known as a Hamiltonian cycle.
- Two paths are independent if they do not have any internal vertex in common.
- For a weighted graph, the weight of a path is the sum of the weights of the traversed edges.

Define Trail and Walk

- A **trail** t in G is a sequence of vertices $w_1, w_2, w_3, \cdots, w_N$ such that $(w_i, w_{i+1}) \in E$ for $1 \leq i \leq N$ and that $e_i \neq e_j$ with $i \neq j$.
- A **walk** is an alternating sequence of vertices and edges, beginning and ending with a vertex, where each vertex is incident to both the edge that precedes it and the edge that follows it in the sequence, and where the vertices that precede and follow an edge are the end vertices of that edge.
- A walk is *closed* if its first and last vertices are the same, and *open* if they are different.

Example of Path, Trial, and Walk



Walk is the most general!
Walk: {1,4,5,1,6,5,4,3}

Trail is a special type of walk with no repeated edges.
Trail: {2,7,6,1,7,8,9}

Path is a walk with no repeated vertices.
Path: {1,4,5,6,7,8,9}

A walk is closed if the starting and ending nodes are the same.

A cycle is a closed trail. A cycle of length *k* is called a *k*-cycle.

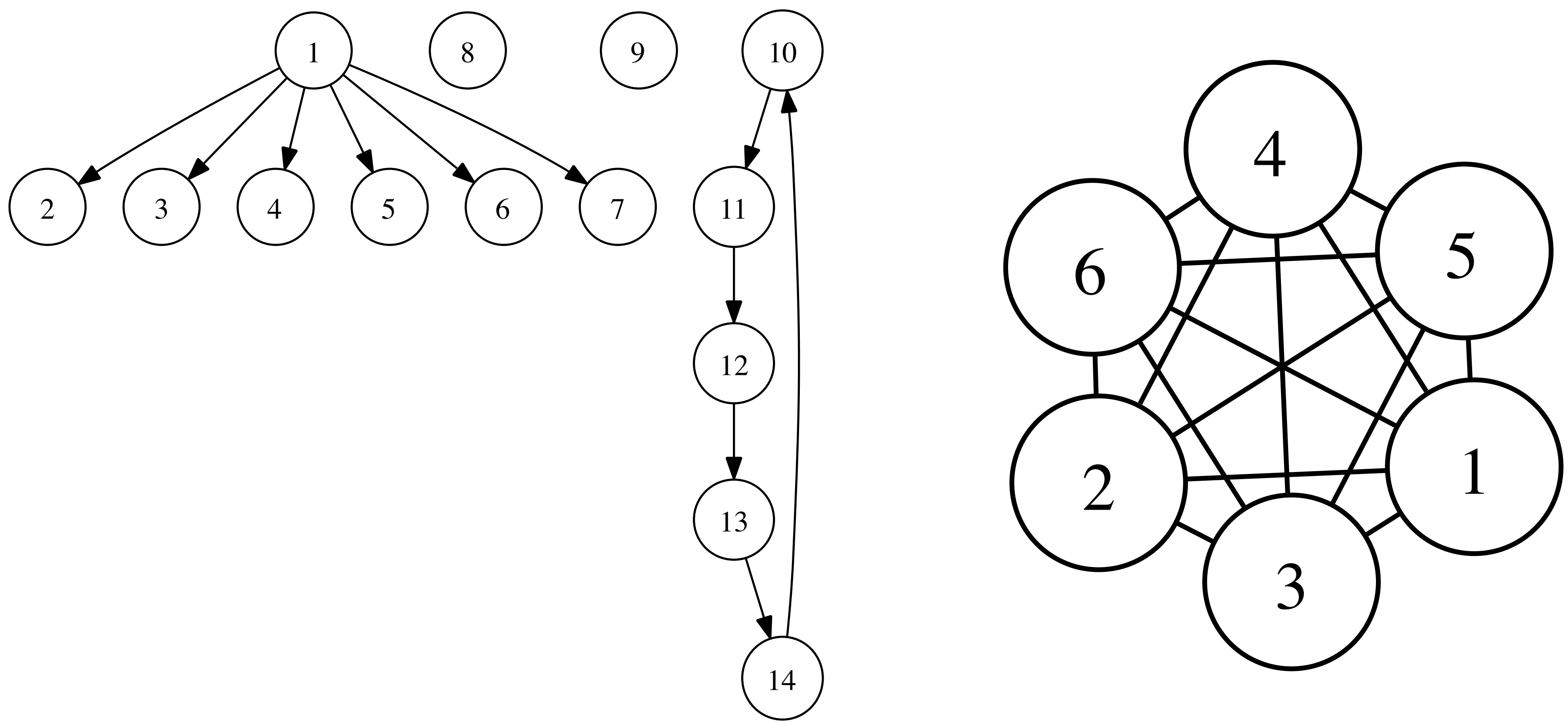
Define Some Graph Properties

- The **eccentricity** ϵ of a vertex v is the greatest distance between v and any other vertex.
- The **radius** of a graph is the minimum eccentricity of any vertex.
- The **diameter** of a graph is the maximum eccentricity of any vertex in the graph, i.e., it is the greatest distance between any two vertices.
- A **peripheral vertex** in a graph of diameter d is one that is distance d from some other vertex, i.e., a vertex that achieves the diameter.
- A **pseudo-peripheral vertex** v has the property that for any vertex u, if v is as far away from u as possible, then u is as far away from v as possible.
- The **girth** of a graph is the length of a shortest cycle contained in the graph. If the graph does not contain any cycles, its girth is defined to be infinity.

Define Connected Component

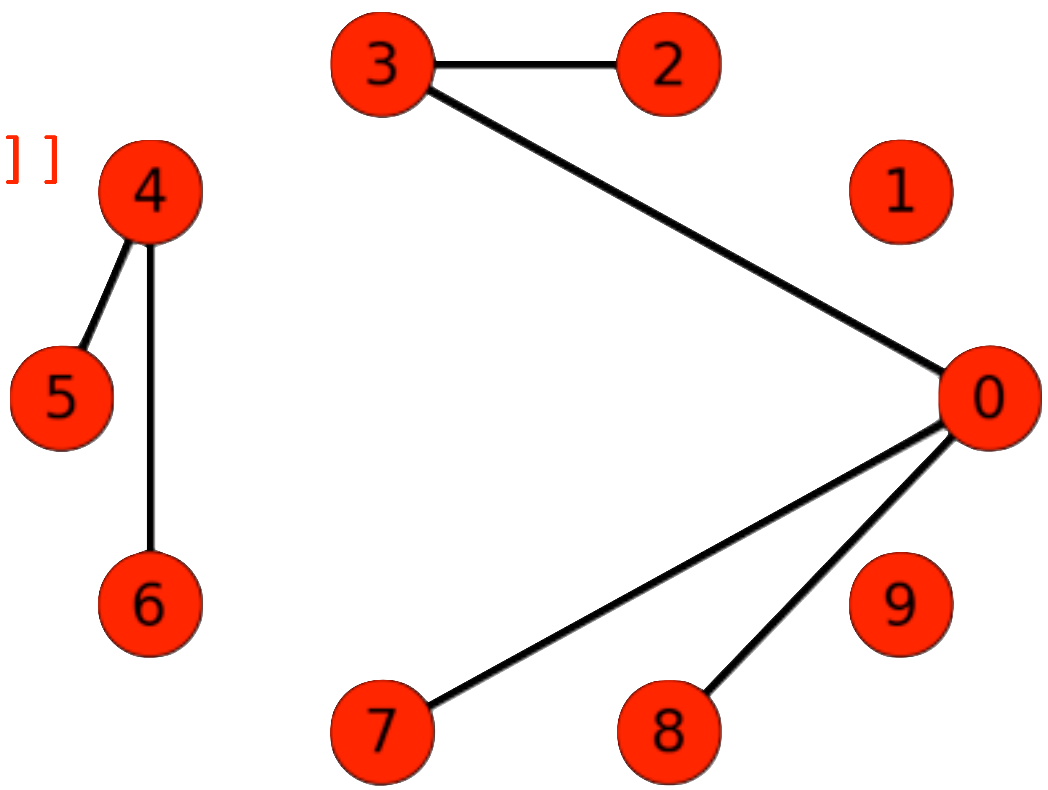
- A **connected component** of an undirected graph is a subgraph in which any two vertices are connected to each other by paths, and to which no more vertices or edges can be added while preserving its connectivity. That is, it is a maximal connected subgraph.
- An undirected graph is connected if there is a path from every vertex to every other vertex.
- A directed graph with this property is called **strongly connected**.
- A **weakly connected graph** is a a directed graph which is not strongly connected, but the underlying graph (without direction to the edges) is connected.
- A **complete graph** is a graph in which there is an edge between every pair of vertices.

Example of Components



Connected Components

```
>>> G =
nx.generators.random_graphs.gnp_random_graph(10,0.15)
>>>
>>> nx.is_connected(G)
False
>>> nx.number_connected_components(G)
4
>>> nx.connected_components(G)
[[0, 8, 2, 3, 7], [4, 5, 6], [1], [9]]
```



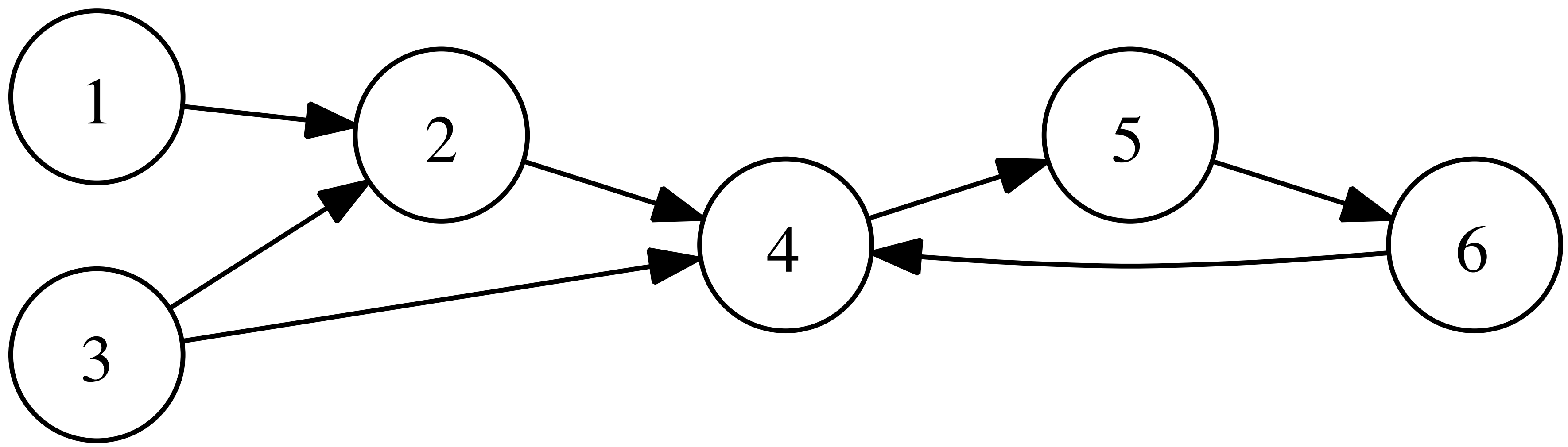
Define Cutpoint

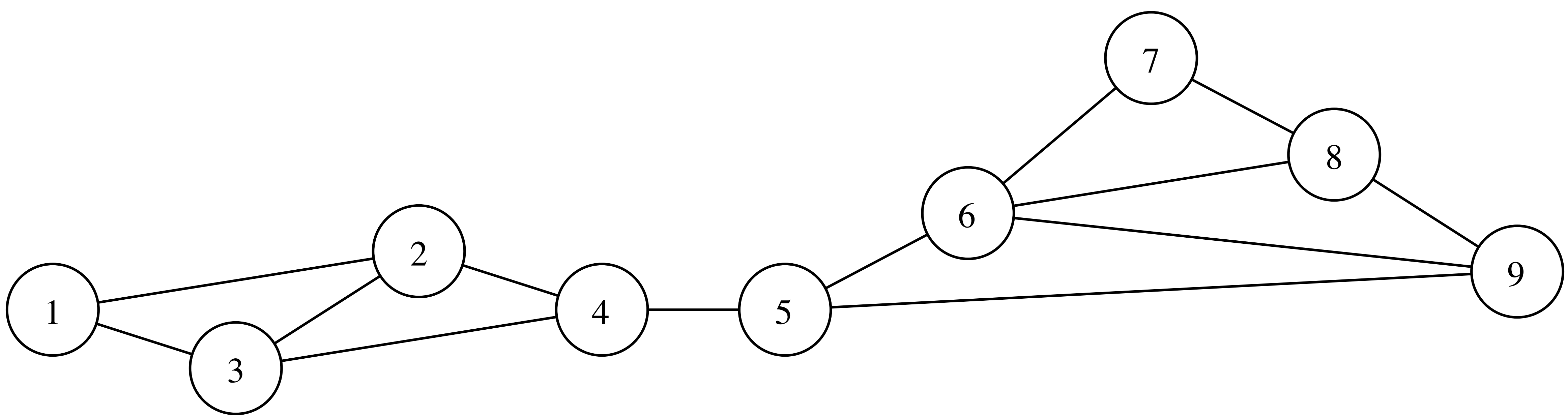
- A **cutpoint** is a vertex whose removal from the graph increases the number of components. That is, it makes some points unreachable from some others. It disconnects the graph.
- A **cutset** is a collection of points whose removal increases the number of components in a graph.
- A **minimum weight cutset** consists of the smallest set of points that must be removed to disconnect a graph. The number of points in a minimum weight cutset is called the **point connectivity** of a graph.
- If a graph has a cutpoint, the connectivity of the graph is 1.
- The minimum number of points separating two nonadjacent points s and t is also the maximum number of point-disjoint paths between s and t.

Define Cutpoint

- A **bridge** is an edge whose removal from a graph increases the number of components (disconnects the graph).
- An **edge cutset** is a collection of edges whose removal disconnects a graph.
- A local bridge of degree k is an edge whose removal causes the distance between the endpoints of the edge to be at least k.
- The **edge-connectivity** of a graph is the minimum number of lines whose removal would disconnect the graph. The minimum number of edges separating two nonadjacent points s and t is also the maximum number of edge-disjoint paths between s and t.

Example of a Cutpoint and Bridge



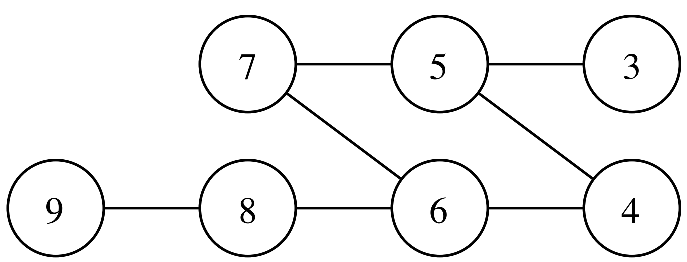


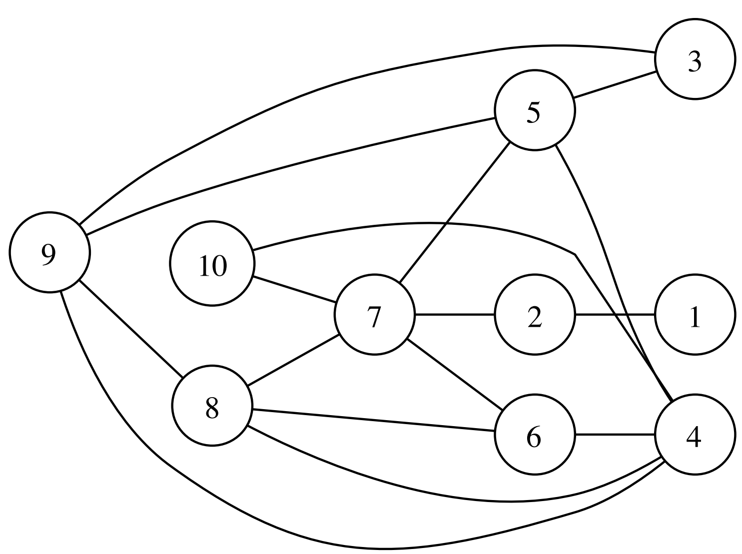
Define Graph Density

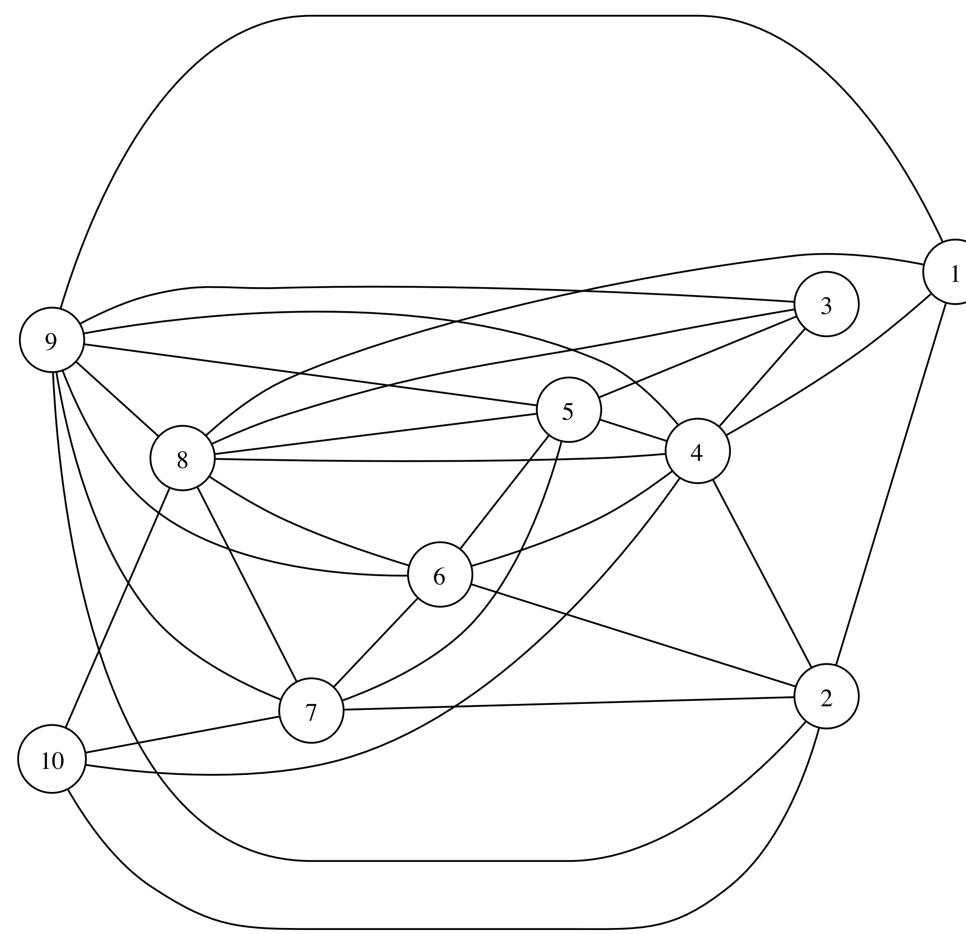
For undirected simple graphs, the **graph density** is defined as:

$$D = \frac{2|E|}{|V|(|V|-1)}, \tag{1}$$

where $|E|$ denotes the number of edges and $|V|$ denotes the number of vertices. The maximum number of edges is $\frac{1}{2}|V|(|V|-1)$, so the maximal density is 1 (for complete graphs) and the minimal density is 0.



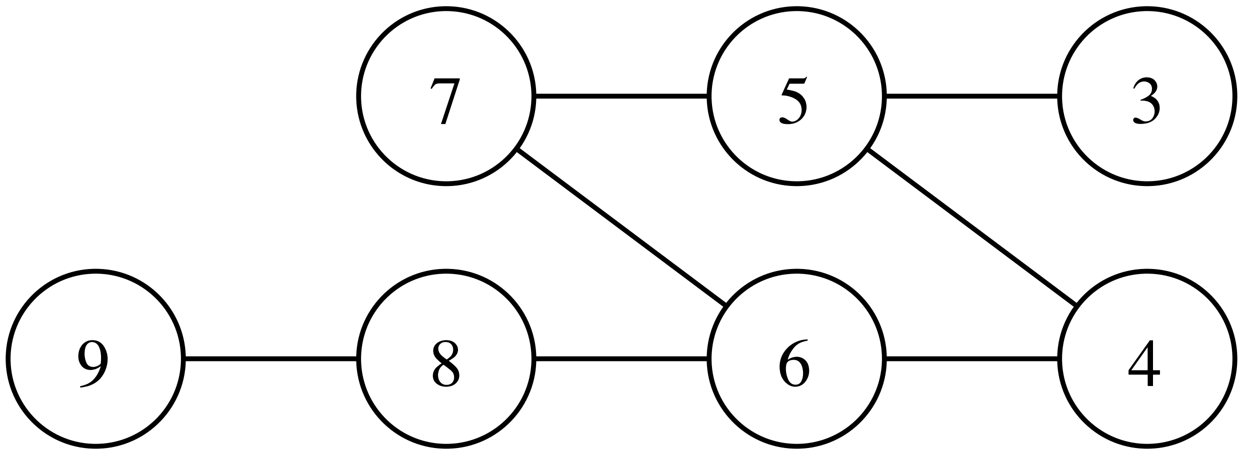




Define Graph Distance

- The **distance** $d_G(u, v)$ between two (not necessary distinct) vertices u and v in a graph G is the length of a shortest path between u and v.
- When u and v are identical, their distance is 0. When u and v are unreachable from each other, their distance is defined to be infinity ∞.
- The average distance is the summation of the distance between all pairs of reachable nodes divided by the number of nodes.

$$d_{av}(G) = \frac{\sum_{u,v}^{V} d_G(u, v)}{|V|}$$



Define Degree Centrality

Let $G = (V, E)$ with n vertices, the **Degree Centrality** $C_D(v)$ for a vertex v is defined as

$$C_D(v) = \frac{\deg(v)}{n-1} \tag{1}$$

For directed graphs, the above can be decomposed to include indegree and outdegree as

$$C_{Din}(v) = \frac{\mathrm{indeg}(v)}{n-1} \tag{2}$$

$$C_{Dout}(v) = \frac{\mathrm{outdeg}(v)}{n-1} \tag{3}$$

Define Group Degree Centralization

The **Group Degree Centralization** is defined by Freeman as

$$C_D(G) = \frac{\sum_v (\Delta_G - C_D(v))}{\max_H \sum_{v \in H} (\Delta_H - C_D(v))}, \tag{1}$$

where Δ_G is the maximum degree of any node in G, $C_D(v)$ is the degree of node v in G and the maximum is taken over all possible graph of the same order (the same number of nodes).

Define Group Degree Centralization

Let n and m denote the numbers of nodes and edges, respectively. We have

$$C_D(G) = \frac{n\Delta_G - \sum_v (C_D(v))}{(n-1)(n-2)}. \tag{1}$$

For an undirected graph,

$$C_D(G) = \frac{n\Delta_G - 2m}{(n-1)(n-2)} \tag{2}$$

For a directed graph,

$$C_D^{in}(G) = \frac{n\Delta_G^{in} - m}{(n-1)(n-2)} \tag{3}$$

and

$$C_D^{out}(G) = \frac{n\Delta_G^{out} - m}{(n-1)(n-2)}. \tag{4}$$

Define Degree Centrality for G

Let V^* be the node with the highest degree centrality in G. Let $G' = (V', E')$ be the n node connected graph that maximizes the following quantity

$$H = \sum_{j=1}^{|V'|} C_D(v'^*) - C_D(v'_j) \tag{1}$$

Then the degree centrality of the graph G is defined as

$$C_D(G) = \frac{\sum_{i=1}^{|V|}[C_D(v^*) - C_D(v_i)]}{H} \tag{2}$$

H is maximized when the graph G' contains one node that is connected to all other nodes are connected only to this one central node (a star graph). In this case

$$H = (n-1)(1 - \frac{1}{n-1}) = n - 1 \tag{3}$$

so the degree centrality of G reduces to

$$C_D(G) = \frac{\sum_{i=1}^{|V|}[C_D(v^*) - C_D(v_i)]}{n-2} \tag{4}$$

Define Closeness Centrality

Let $d(u, v)$ denote the distance from u to v and $D(v) = \sum_u d(v, u)$ be the total distance from v to all other nodes. The **Closeness Centrality** of v is measured by $1/D(v)$ and normalized to $C_C(v) = (n-1)/D(v)$ since the minimum $D(v)$ is $n-1$, which happens at the center of a star graph. Freeman defines the group centrality as follows,

$$C_C(G) = \frac{\sum_v (C_C(v^*) - C_C(v))}{\max_H \sum_{v \in H} (C_C(v^*) - C_C(v))}, \tag{1}$$

where v^* is the node of maximum closeness.

Define Between Centrality

The **Between Centrality** is a measure of a vertex within a graph (this can also be extended to edge as well). Vertices that occur on many shortest paths between other vertices have higher betweenness than those that do not. Hence, Betweenness Centrality of a node counts the number of times that a node lies along the shortest path between two others vertices in the graph. It is defined as

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{g\sigma_{st}}. \tag{1}$$

where σ_{st} is the number of shortest paths from s to t and $\sigma_{st}(v)$ is the number of shortest paths from s to t that pass through a vertex v.

The normalized betweenness of undirected graphs is given by

$$C'_B(v) = \frac{C_B(v)}{(n-1)(n-2)/2}. \tag{2}$$

Define Group Between Centrality

The normalized betweenness of directed graphs is given by

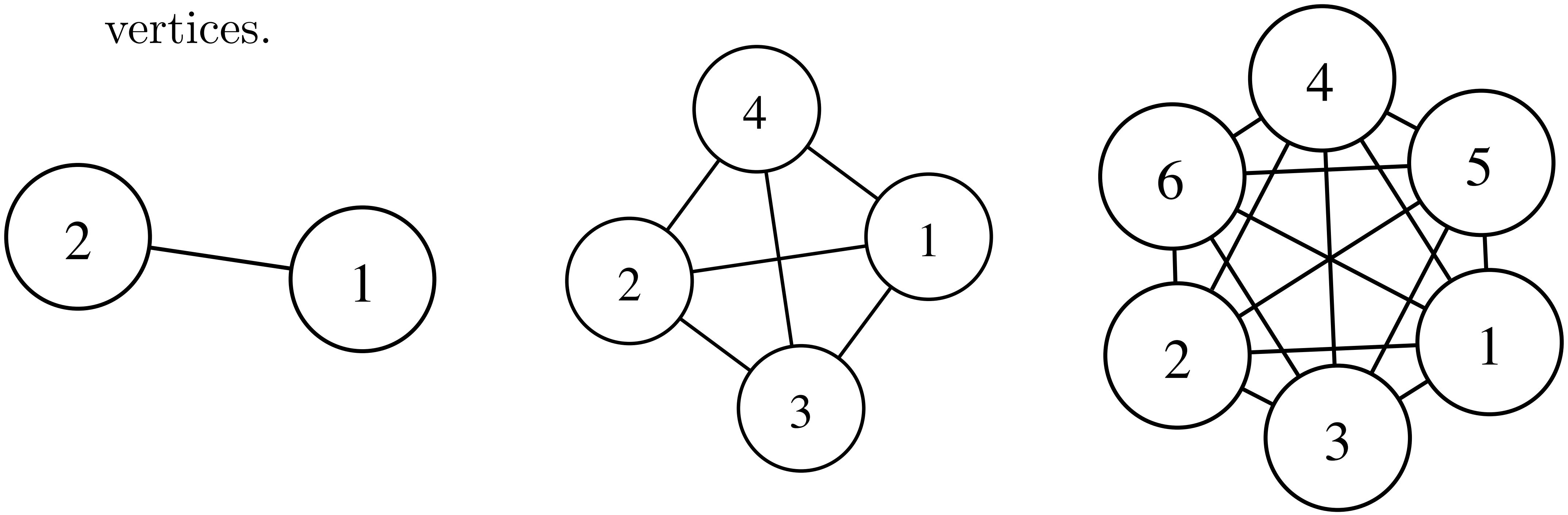
$$C'_B(v) = \frac{C_B(v)}{(n-1)(n-2)}. \tag{1}$$

To compute the group betweenness centrality, we compute the one of a star at first. For a star, the center has betweenness $(n-1)(n-2)/2$ and it is zero for all the others. The group centrality of a star is then $(n-1)^2(n-2)/2$. Then we have

$$\begin{aligned} C_B(G) &= \frac{\sum_v (C_B(v^*) - C_B(v))}{(n-1)^2(n-2)/2} \\ &= \frac{\sum_v (C'_B(v^*) - C'_B(v))}{n-1} \end{aligned} \tag{2}$$

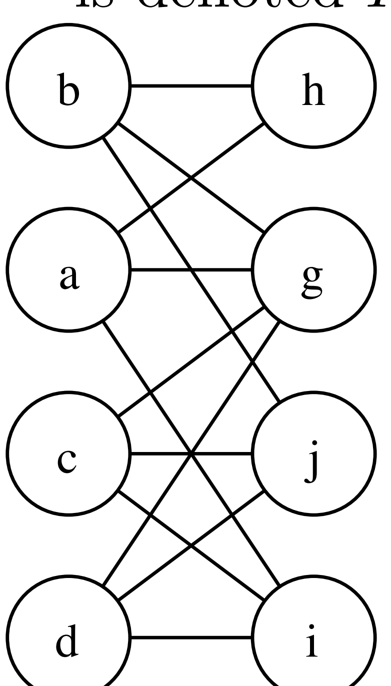
Complete Graph

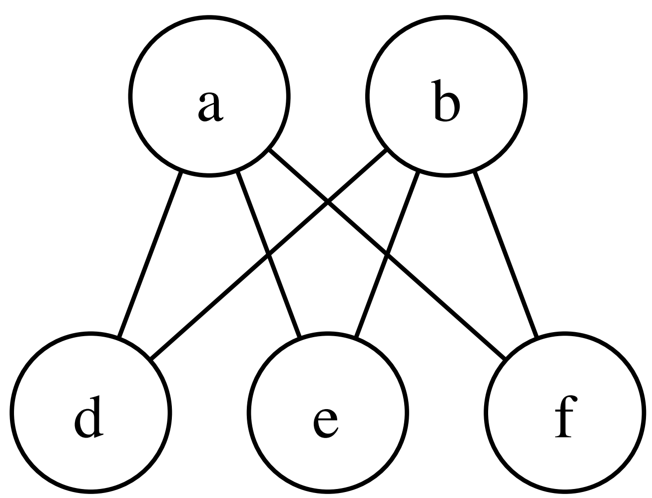
- A **complete graph** is a simple graph in which every pair of distinct vertices is connected by an edge.
- The complete graph on n vertices has n vertices and $n(n-1)/2$ edges, and is denoted by K_n.
- It is a regular graph of degree $n-1$.
- All complete graphs are their own cliques. They are maximally connected as the only vertex cut which disconnects the graph is the complete set of vertices.

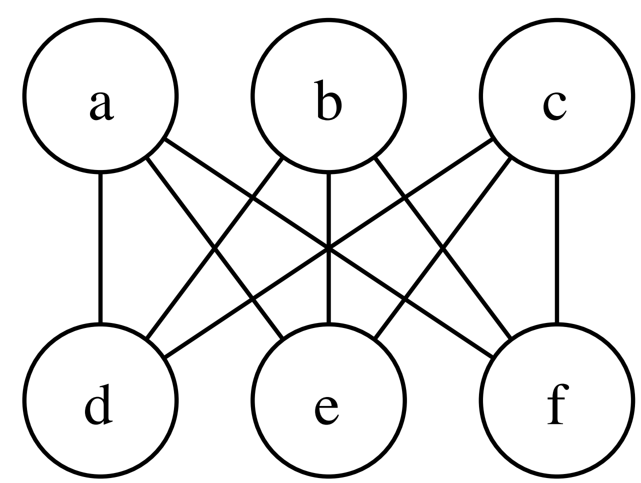


Bipartite Graph

- A **bipartite graph** (or bigraph) is a graph whose vertices can be divided into two disjoint sets U and V such that every edge connects a vertex in U to one in V, i.e., U and V are independent sets.
- Equivalently, a bipartite graph is a graph that does not contain any odd-length cycles.
- A **balanced bipartite graph** is a bipartite graph that satisfy the condition $|U| = |V|$.
- A **complete bipartite graph** $G = (U + V, E)$ is bipartite such that for any two vertices $u \in U$ and $v \in V$ that (u, v) is an edge in G.
- The complete bipartite graph with partitions of size $|U| = m$ and $|V| = n$, is denoted $K_{m,n}$.







Properties of Bipartite Graphs

- A graph is bipartite if and only if it does not contain an odd cycle. Therefore, a bipartite graph cannot contain a clique of size 3 or more.
- A graph is bipartite if and only if it is 2-colorable.
- The size of minimum vertex cover is equal to the size of the maximum matching.
- The size of the maximum independent set plus the size of the maximum matching is equal to the number of vertices.

Basic Functions

`is_bipartite(G)`

- Returns True if graph G is bipartite, False if not. is_bipartite_node_set(G, nodes) Returns True if nodes and G/nodes are a bipartition of G.

`sets(G)`

- Returns bipartite node sets of graph G.

`color(G)`

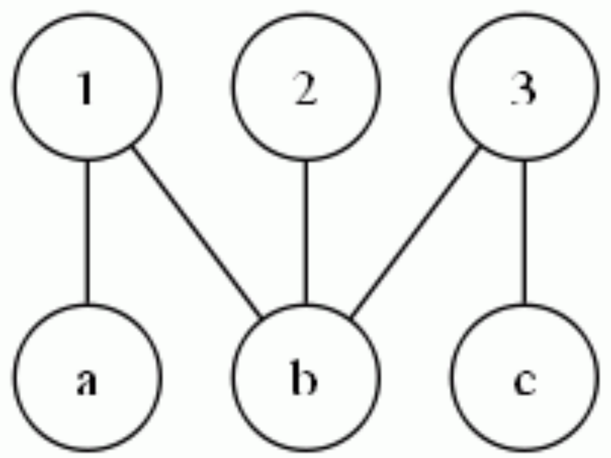
- Returns a two-coloring of the graph.

`density(B, nodes)`

- Return density of bipartite graph B. degrees(B, nodes[, weighted]) Return the degrees of the two node sets in the bipartite graph B.

Bipartite Module in NetworkX

- This module provides functions and operations for bipartite graphs. Bipartite graphs G(X,Y, E) have two node sets X,Y and edges in E that only connect nodes from opposite sets.
- For example:




```
>>> import networkx as nx
>>> top_nodes=[1,1,2,3,3]
>>> bottom_nodes=['a','b','b','b','c']
>>> edges=zip(top_nodes,bottom_nodes) # create 2-tuples of
edges
>>> B=nx.Graph(edges)
>>> print(B.edges())
```

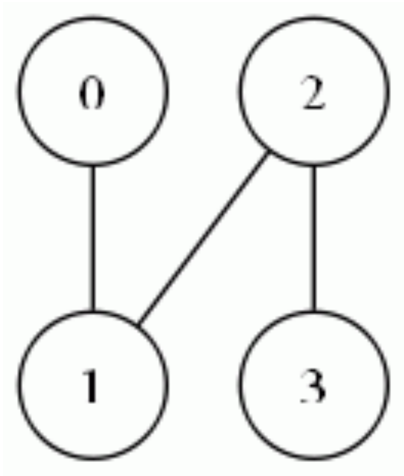
- The bipartite algorithms are not imported into the networkx (version 1.5) namespace at the top level so you need to do:

```
>>> from networkx.algorithms import bipartite
```

Examples of Basic Functions

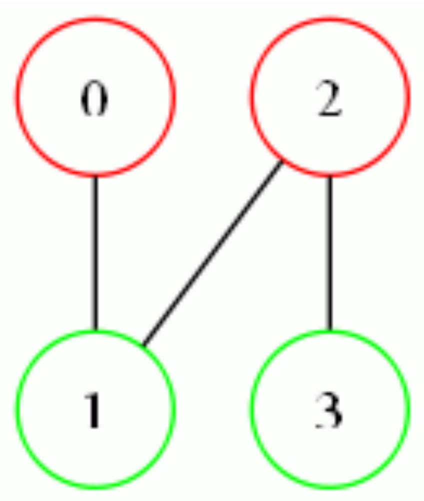
- networkx.algorithms.bipartite.basic.sets

```
>>> from networkx.algorithms import bipartite
>>> G = nx.path_graph(4)
>>> X, Y = bipartite.sets(G)
>>> list(X)
[0, 2]
>>> list(Y)
[1, 3]
```



- networkx.algorithms.bipartite.basic.color

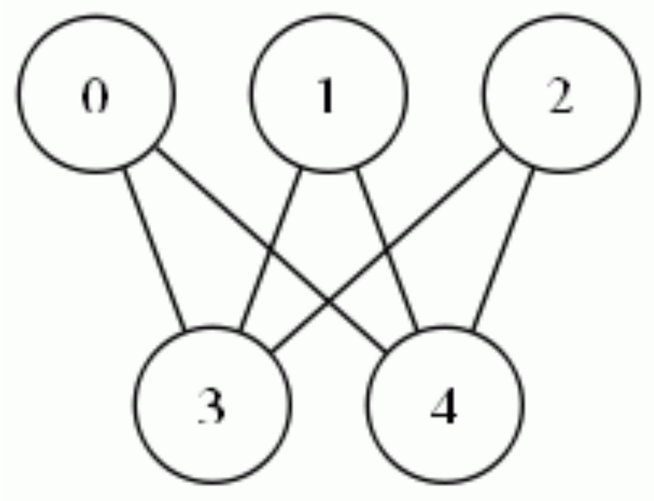
```
>>> from networkx.algorithms import bipartite
>>> G = nx.path_graph(4)
>>> c = bipartite.color(G)
>>> print(c)
{0: 1, 1: 0, 2: 1, 3: 0}
```



More Examples

- networkx.algorithms.bipartite.basic.density

```
>>> from networkx.algorithms import bipartite
>>> G = nx.complete_bipartite_graph(3,2)
>>> X=set([0,1,2])
>>> bipartite.density(G,X)
1.0
>>> Y=set([3,4])
>>> bipartite.density(G,Y)
1.0
```



- networkx.algorithms.bipartite.basic.degrees

```
>>> from networkx.algorithms import bipartite
>>> G = nx.complete_bipartite_graph(3,2)
>>> Y=set([3,4])
>>> degX,degY=bipartite.degrees(G,Y)
>>> degX
{0: 2, 1: 2, 2: 2}
```

Other Functions

- Spectral

  `spectral_bipartivity(G[, nodes, weight])`

  - Returns the spectral bipartivity.

- Clustering

  `clustering(G[, nodes, mode])`

  - Compute a bipartite clustering coefficient for nodes.

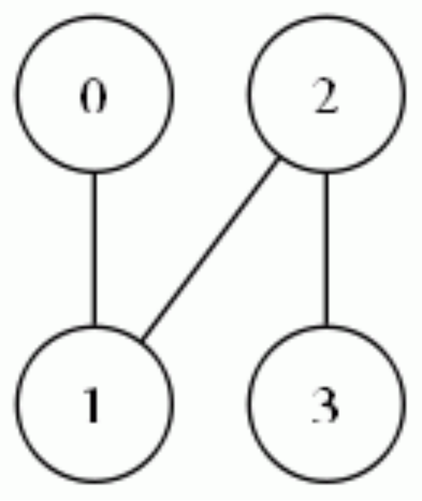
  `average_clustering(G[, nodes, mode])`

  - Compute the average bipartite clustering coefficient.

Examples of Clustering

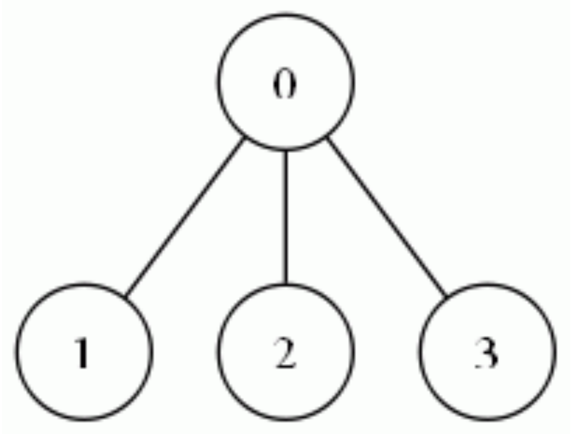
- networkx.algorithms.bipartite.cluster.clustering

```
>>> from networkx.algorithms import bipartite
>>> G=nx.path_graph(4) # path is bipartite
>>> c=bipartite.clustering(G)
>>> c[0]
0.5
>>> c=bipartite.clustering(G,mode='min')
>>> c[0]
1.0
```



- networkx.algorithms.bipartite.cluster.average_clustering

```
>>> from networkx.algorithms import bipartite
>>> G=nx.star_graph(3) # path is bipartite
>>> bipartite.average_clustering(G)
0.75
>>> X,Y=bipartite.sets(G)
>>> bipartite.average_clustering(G,X)
0.0
>>> bipartite.average_clustering(G,Y)
1.0
```



Bipartite Cluster Clustering

- The bipartie clustering coefficient is a measure of local density of connections defined as

$$c_u = \frac{\sum_{v \in N(N(v))} c_{uv}}{N(N(u))}$$

where $N(N(u))$ are the second order neighbors of u in G excluding u, and c_{uv} is the pairwise clustering coefficient between nodes u and v.

- c_{uv} can be defined in three ways.
  -
$$c_{uv} = \frac{\|N(u) \cap N(v)\|}{\|N(u) \cup N(v)\|}$$
  - min:
$$c_{uv} = \frac{\|N(u) \cap N(v)\|}{\min(\|N(u) \cup N(v)\|)}$$
  - max:
$$c_{uv} = \frac{\|N(u) \cap N(v)\|}{\max(\|N(u) \cup N(v)\|)}$$

More Functions

- Redundancy

  `node_redundancy(G[, nodes])`

  - Compute bipartite node redundancy coefficient.

- Centrality

  `closeness_centrality(G, nodes[, normalized])`

  - Compute the closeness centrality for nodes in a bipartite network.

  `degree_centrality(G, nodes)`

  - Compute the degree centrality for nodes in a bipartite network.

  `betweenness_centrality(G, nodes)`

  - Compute betweenness centrality for nodes in a bipartite network.

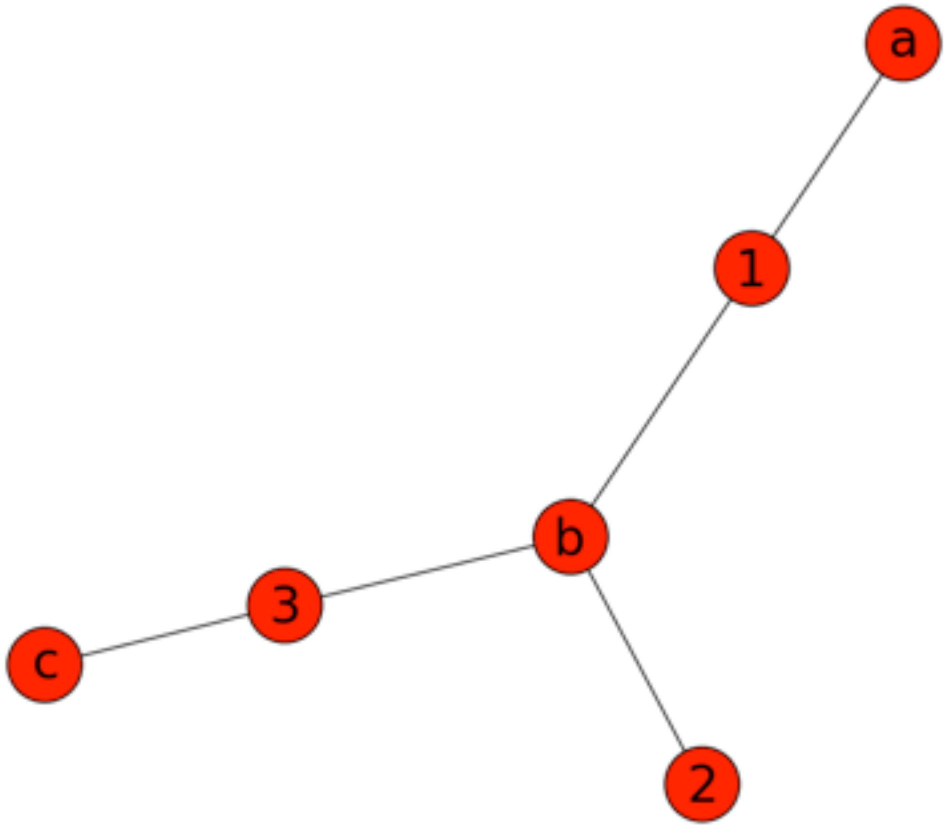
Examples of Redundancy

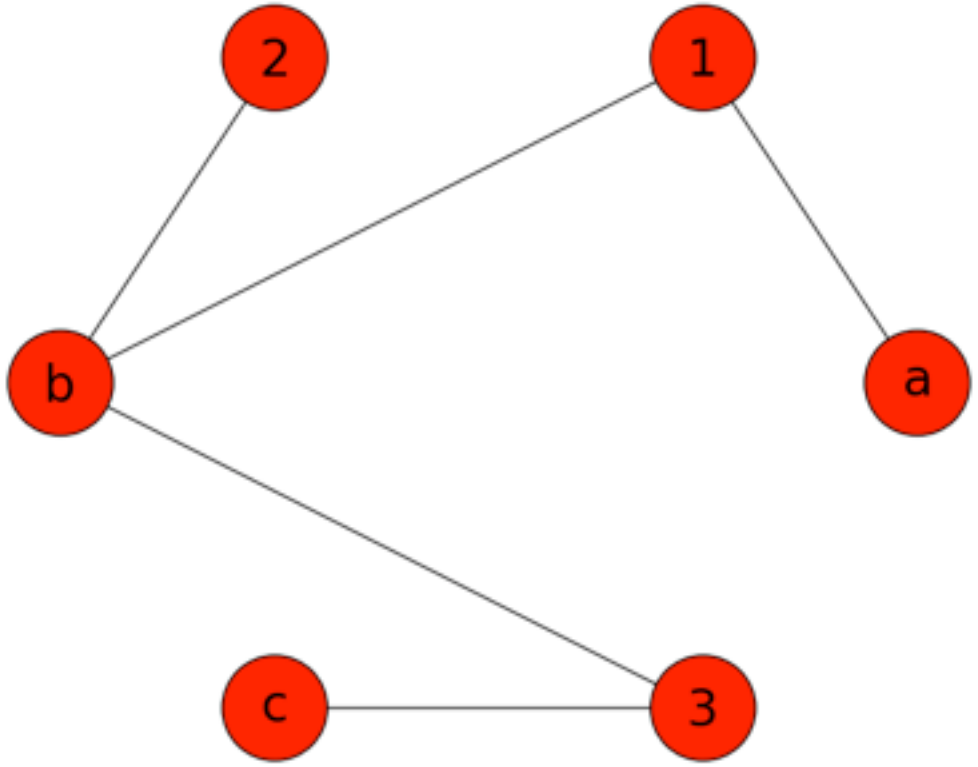
- networkx.algorithms.bipartite.redundancy.node_redundancy

```
>>> from networkx.algorithms import bipartite
>>> G = nx.cycle_graph(4)
>>> rc = bipartite.node_redundancy(G)
>>> rc[0]
1.0
```

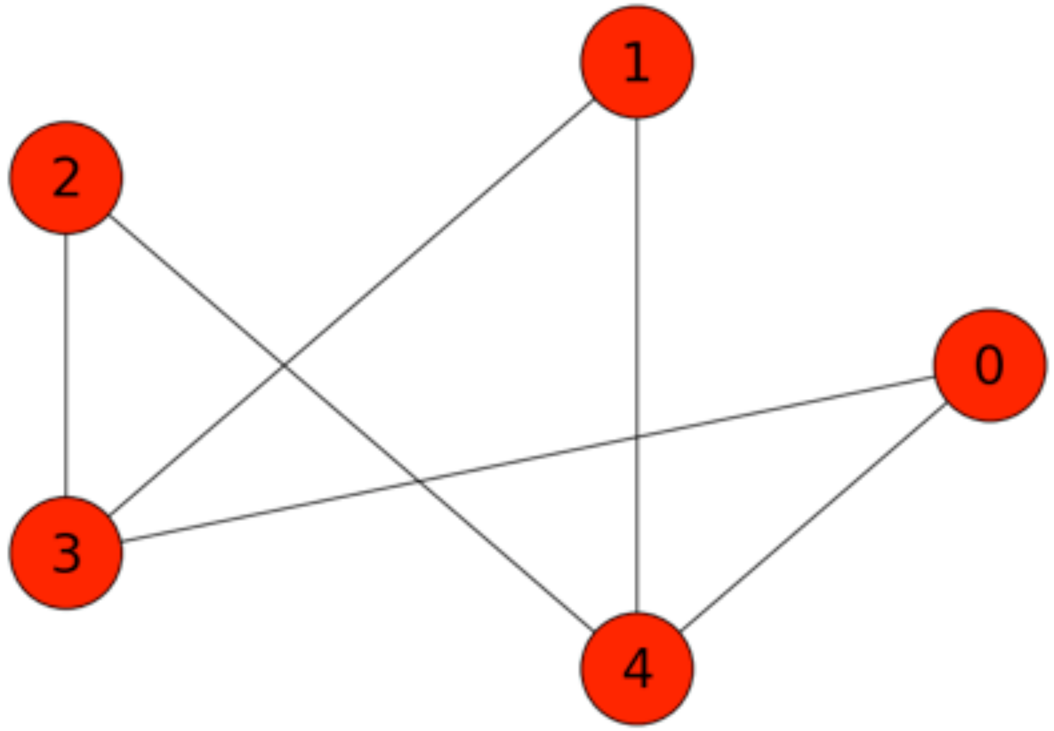
Example

```
import networkx as nx
import matplotlib.pyplot as plt
import pygraphviz

top_nodes=[1,1,2,3,3]
bottom_nodes=['a','b','b','b','c']
edges=zip(top_nodes,bottom_nodes) # create 2-tuples of
edges
G=nx.Graph(edges)
print(G.edges())

nx.draw(G)
plt.savefig("example.png")
plt.show()
```






```
>>> centrality.degree_centrality(G)
{'a': 0.20000000000000001, 1: 0.40000000000000002, 2:
0.20000000000000001, 'b': 0.60000000000000009, 'c':
0.20000000000000001, 3: 0.40000000000000002}
>>> centrality.betweenness_centrality(G)
{'a': 0.0, 1: 0.40000000000000002, 2: 0.0, 'b':
0.80000000000000004, 'c': 0.0, 3: 0.40000000000000002}
>>> centrality.closeness_centrality(G)
{'a': 0.38461538461538464, 1: 0.55555555555555558, 2:
0.45454545454545453, 'b': 0.7142857142857143, 'c':
0.38461538461538464, 3: 0.55555555555555558}
```




```
>>> from networkx.algorithms import bipartite
>>> G = nx.complete_bipartite_graph(3,2)
>>> X=set([0,1,2])
>>> bipartite.density(G,X)
>>> Y=set([3,4])

>>> centrality.degree_centrality(G)
{0: 0.5, 1: 0.5, 2: 0.5, 3: 0.75, 4: 0.75}
>>> centrality.betweenness_centrality(G)
{0: 0.055555555555555552, 1: 0.055555555555555552, 2:
0.055555555555555552, 3: 0.25, 4: 0.25}
>>> centrality.closeness_centrality(G)
{0: 0.66666666666666663, 1: 0.66666666666666663, 2:
0.66666666666666663, 3: 0.80000000000000004, 4: 0.80000000000000004}
```

Subgroup Cohesion

- A **Clique** in an undirected graph $G = (V, E)$ is a subset of the vertex set $C \subseteq V$, such that for every two vertices in C, there exists an edge connecting the two. This is equivalent to saying that the subgraph induced by C is complete.
- The size of the clique is the number of vertices it contains.
- The **clique number** $\omega(G)$ of a graph G is the order of a largest clique in G.
- An n**-clique** S of a graph is a maximal set of nodes in which for all $u, v \in S$, the graph-theoretic distance $d(u, v) \leq n$.
- In other words, an n-clique is a set of nodes in which every node can reach every other in n or fewer steps, and the set is maximal in the sense that no other node in the graph is distance n or less from every other node in the subgraph.
- A 1-clique is the same as an ordinary clique.

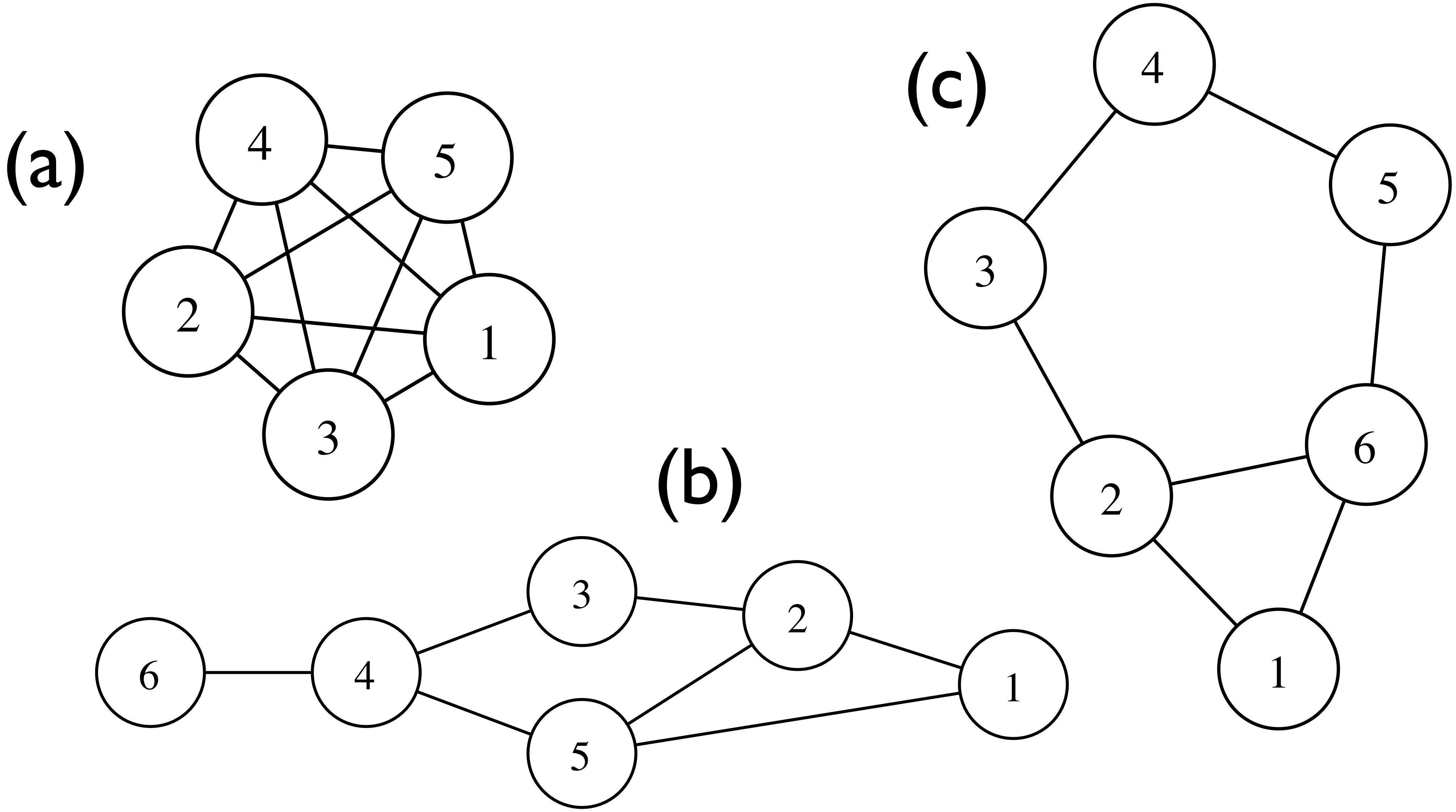
Clan

An n**-clan** is an n-clique in which the diameter of the subgraph G' induced by S is less than or equal to n.

An n**-club** is a subset S of nodes such that in the subgraph induced by S, the diameter is n or less. Every n-clan is both an n-club and an n-clique.

A k**-plex** is a subset S of nodes such that every member of the set is connected to $n - k$ others, where n is the size of S. The k**-plex** generalizes the clique by relaxing density.

Example of Cliques, Clans, Clubs, etc.



(a) A complete graph and also a clique of size 5. (b) An example of a clique of size 3. (c) An example of 2-clique with {1, 2, 3, 4, 5}. An example of 2-clan with {2, 3, 4, 5, 6}. An example of 2-club with {1, 2, 3, 6}.

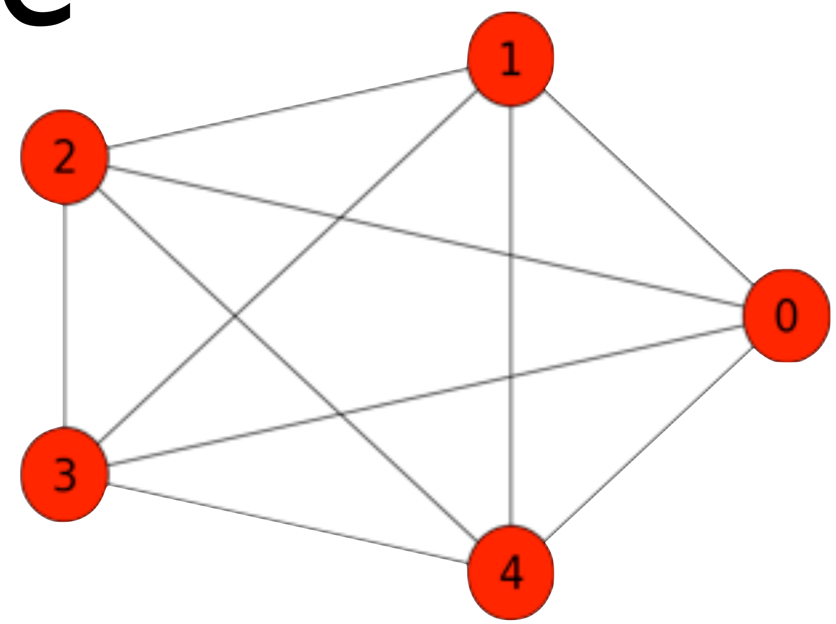
The Clique Problem

- There is a clique of size at least *k* iff there is an independent set of size last least k in the complement graph.
- Brute Force Algorithm
  - Examine each subgraph with at least *k* vertices and check to see if it forms a clique.
  - Polynomial if *k* is the number of vertices, or a constant

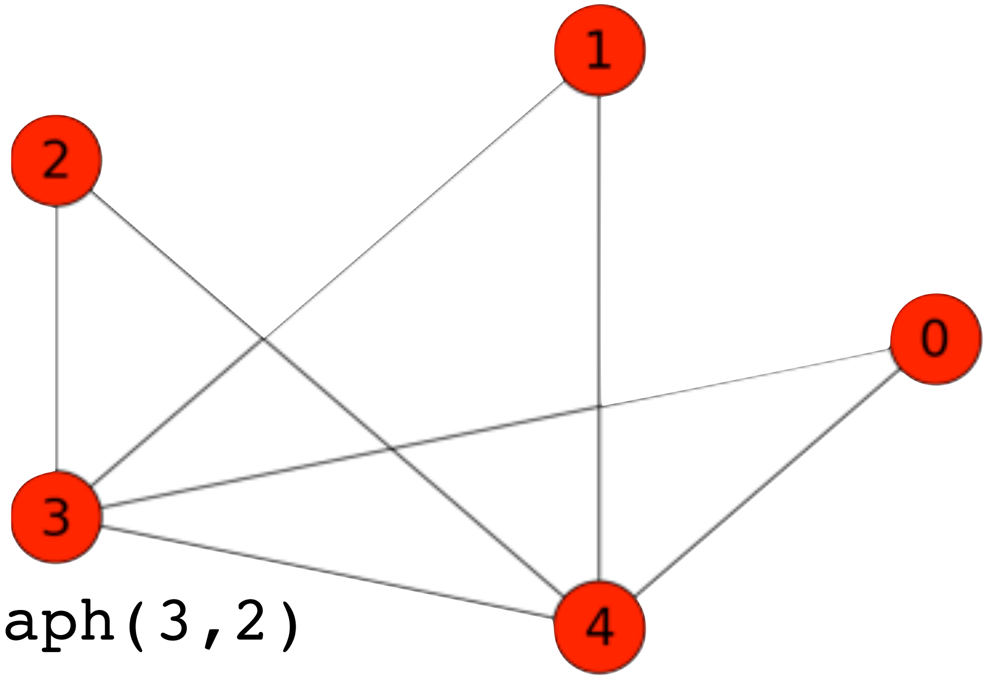
$$\left(\begin{array}{c} V \\ k \end{array} \right) = \frac{V!}{k!(V-k)!}$$

  - Consider each node to be a clique of size one, and to merge cliques into larger cliques
  - Linear time by the edges
  - Disjoint-set data structure

Clique Example



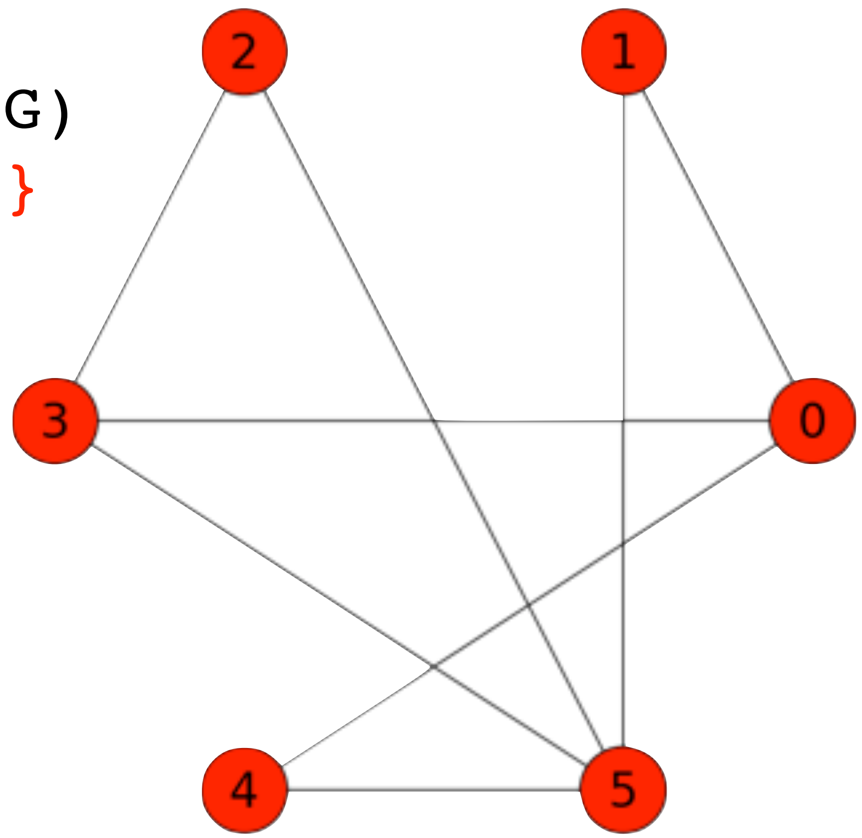

```
from networkx.algorithms import clique

G = nx.complete_graph(5)

clique.graph_clique_number(G) # Return the clique number
(size of the largest clique) for G
5
list(clique.find_cliques(G)) # Search for all maximal
cliques in a graph.
[[0, 1, 2, 3, 4]]
clique.cliques_containing_node(G) # Returns a list of
cliques containing the given node.
{0: [[0, 1, 2, 3, 4]], 1: [[0, 1, 2, 3, 4]], 2: [[0, 1, 2,
3, 4]], 3: [[0, 1, 2, 3, 4]], 4: [[0, 1, 2, 3, 4]]}
```




```
>>> G = nx.complete_bipartite_graph(3,2)
>>> G.add_edge(3,4)
>>>
>>> clique.graph_clique_number(G) # Return the clique
number (size of the largest clique) for G
3
>>> list(clique.find_cliques(G)) # Search for all maximal
cliques in a graph.
[[3, 4, 0], [3, 4, 1], [3, 4, 2]]
>>> clique.cliques_containing_node(G) # Returns a list of
cliques containing the given node.
{0: [[3, 4, 0]], 1: [[3, 4, 1]], 2: [[3, 4, 2]], 3: [[3, 4,
0], [3, 4, 1], [3, 4, 2]], 4: [[3, 4, 0], [3, 4, 1], [3, 4,
2]]}
```

Graph Measures

```
>>> from networkx.algorithms import generators
>>> from networkx.algorithms import distance_measures
>>> G = nx.generators.random_graphs.gnp_random_graph(6,0.5)

>>> distance_measures.diameter(G)
2
>>> distance_measures.eccentricity(G)
{0: 2, 1: 2, 2: 2, 3: 2, 4: 2, 5: 2}
>>> distance_measures.center(G)
[0, 1, 2, 3, 4, 5]
>>> distance_measures.periphery(G)
[0, 1, 2, 3, 4, 5]
>>> distance_measures.radius(G)
2
```



References

- NetworkX, http://networkx.lanl.gov/
- D. J. Cook and L. B. Holder, *Mining Graph Data*, 1st ed. Wiley-Interscience, 2006
- T. G. Lewis, *Network Science: Theory and Applications*, 1st ed. Wiley, 2009.
- M. Gladwell, *The Tipping Point: How Little Things Can Make a Big Difference*. Back Bay Books, 2002.