Modelling predictive analytics

from network science to graph processing and stores

Genoveva Vargas-Solar
Senior Scientist, French Council of Scientific Research, LIG-LAFMIA
genoveva.vargas@imag.fr

http://vargas-solar.com/big-linked-data-keystone/

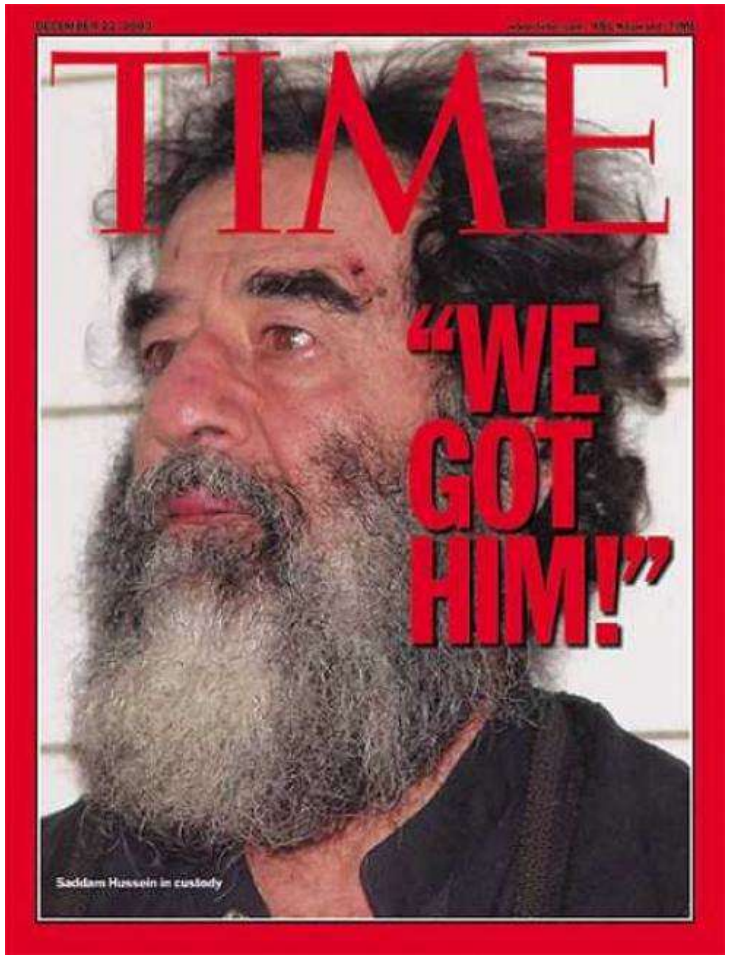
The study of network representations of physical, biological, and social phenomena leading to predictive models of these phenomena

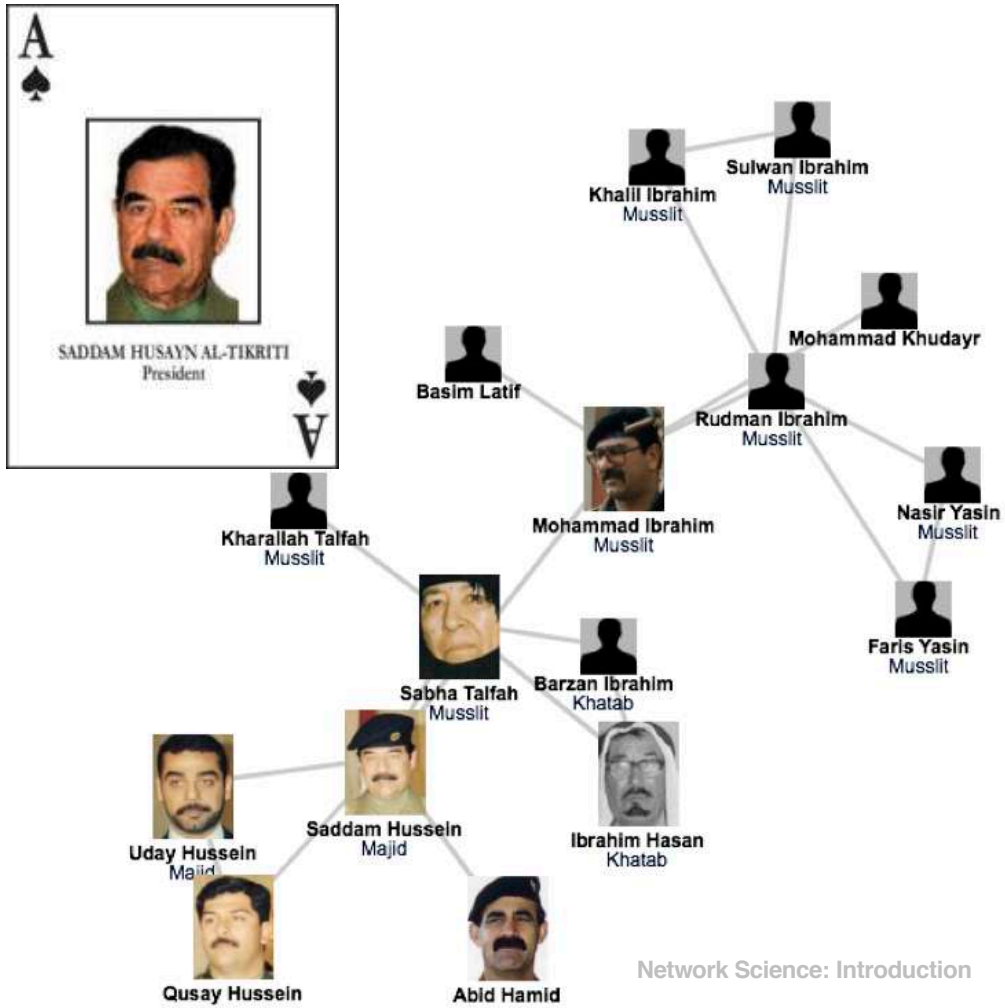
NETWORK SCIENCE FIELDS

The field draws on theories and methods including

- g**raph theory** from **mathematics**
- statistical mechanics from **physics**
- data mining and information visualization from **computer scien**ce
- inferential modelling from **statistics**
- **social** structure from sociology

THE FATE OF SADDAM AND NETWORK SCIENCE





THE FAITH OF SADDAMM HUSSEIN AND NETWORK SCIENCE

The capture of Saddam Hussein:

- Shows the strong **predictive power** of networks
- Underlies the need to obtain **accurate maps of the net**works; and the often heroic difficulties we encounter during the mapping process.
- demonstrates the **remarkable stability of these networks**:
  - the capture of Hussein was not based on fresh intelligence, but rather on his pre-invasion social links, unearthed from old photos stacked in his family album.
- Shows that the **choice of network** we focus on makes a huge difference:
  - the hierarchical tree, that captured the official organization of the Iraqi government, was of no use when it came to Saddam Hussein's whereabouts

Behind each complex system there is a network, that defines the interactions between the component

THE ROLE OF NETWORKS

Behind each system studied in complexity there is an intricate wiring diagram, or a **network**, that defines the interactions between the component

We will never understand complex system unless we map out and understand the networks behind them

THE HISTORY OF NETWORK ANALYSIS

- **Graph theory:** 1735, Euler
- **Social Network Research:** 1930s, Moreno
- **Communication networks/internet:** 1960s
- **Ecological Networks:** May, 1979

CHARACTERISTICS OF NETWORK SCIENCE

Interdisciplinary

Empirical, data driven

Quantitative and Mathematical

Computational

CHARACTERISTICS OF NETWORK SCIENCE

Interdisciplinary

Empirical, data driven

Quantitative and Mathematical

Computational

CHARACTERISTICS OF NETWORK SCIENCE

Interdisciplinary

Empirical

Quantitative and Mathematical

Computational

CHARACTERISTICS OF NETWORK SCIENCE

Interdisciplinary

Empirical

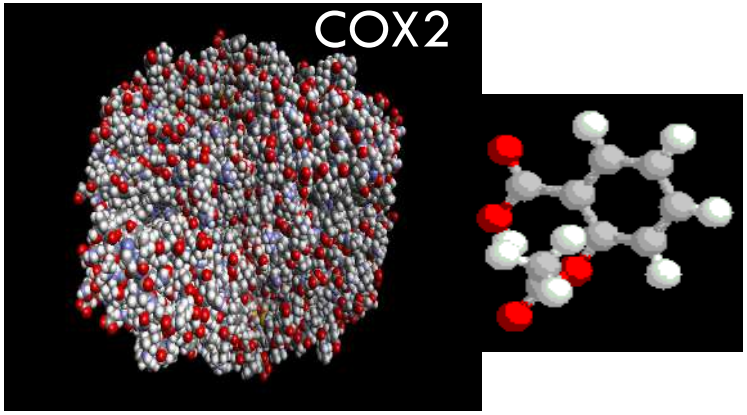
Quantitative and Mathematical

Computational

DRUG DESIGN, METABOLIC ENGINEERING

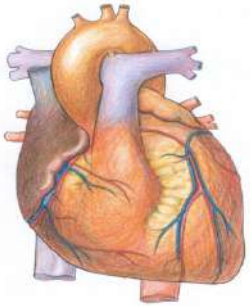
Reduces
Inflammation
Fever
Pain

Reduces the risk of
Alzheimer's Disease



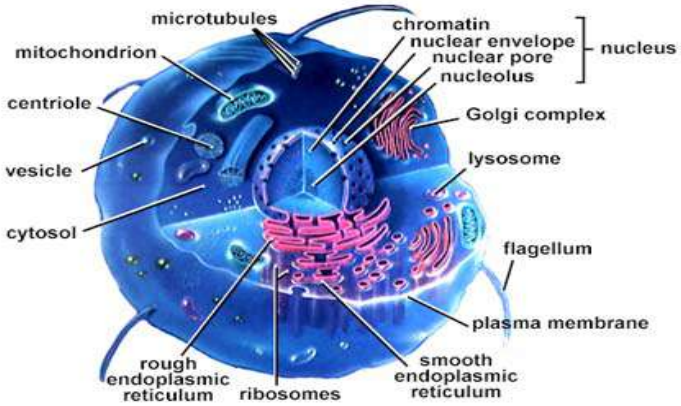
Reduces the risk of
breast cancer
ovarian cancers
colorectal cancer

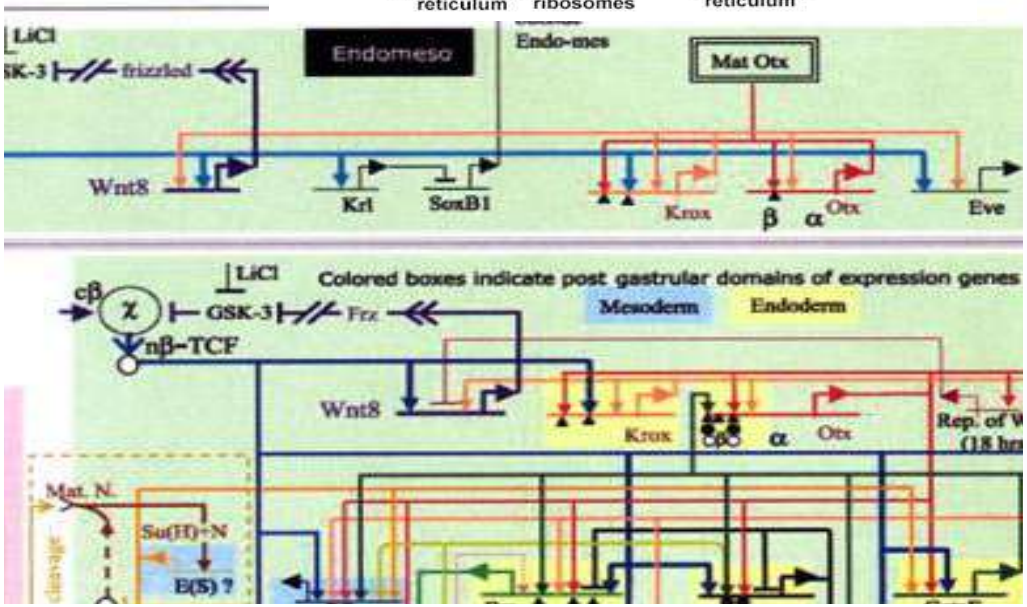
Prevents
Heart attack
Stroke

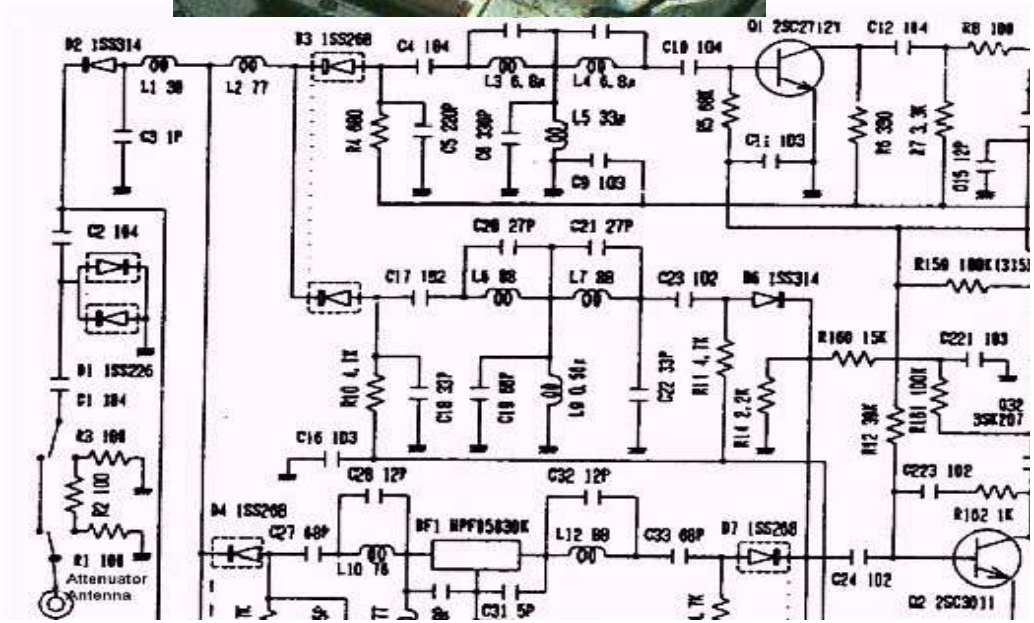


Causes
Bleeding
Ulcer

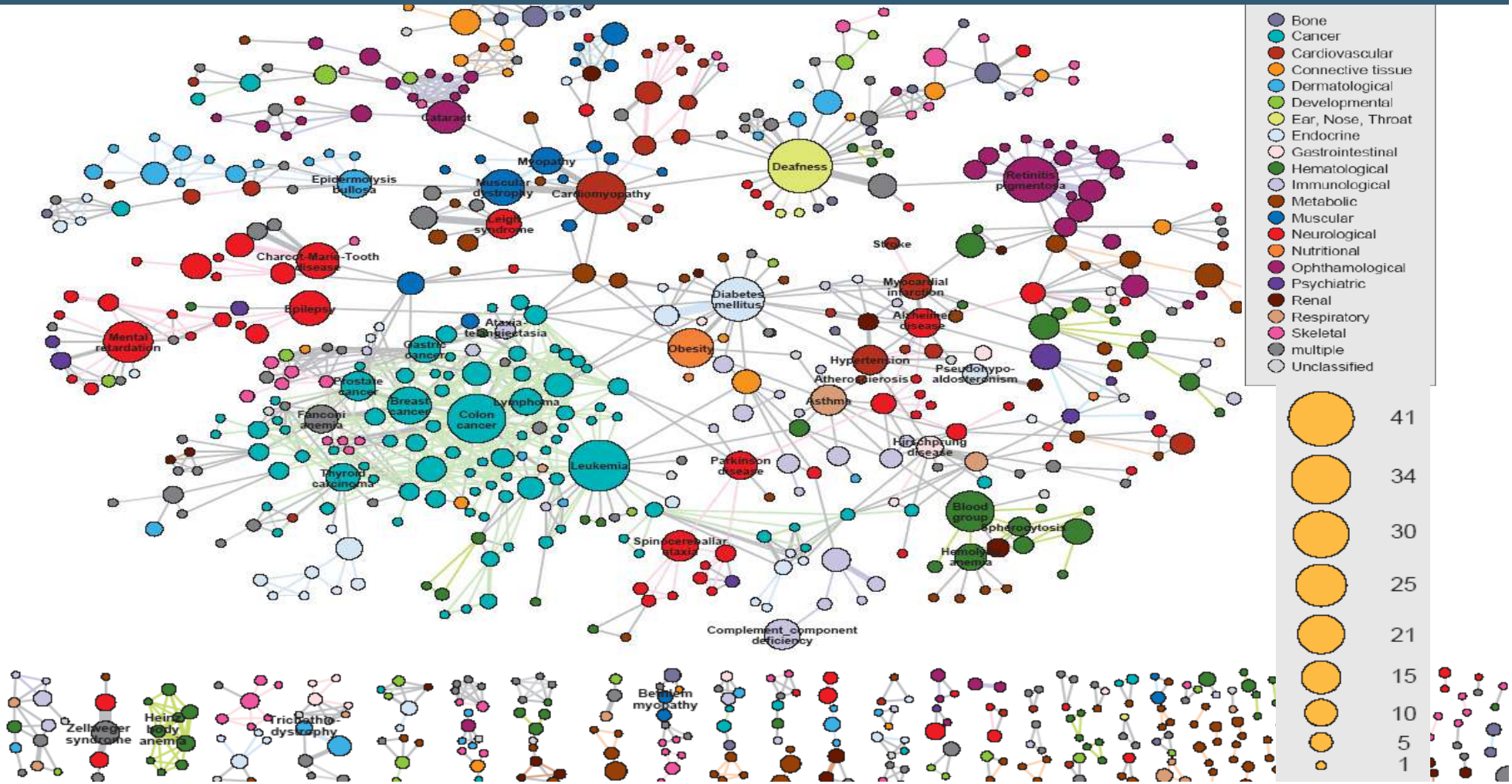
DRUG DESIGN, METABOLIC ENGINEERING



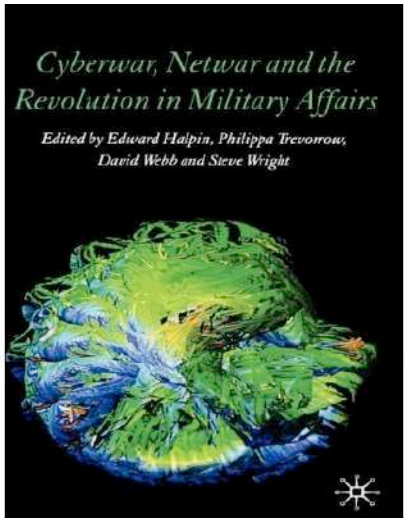


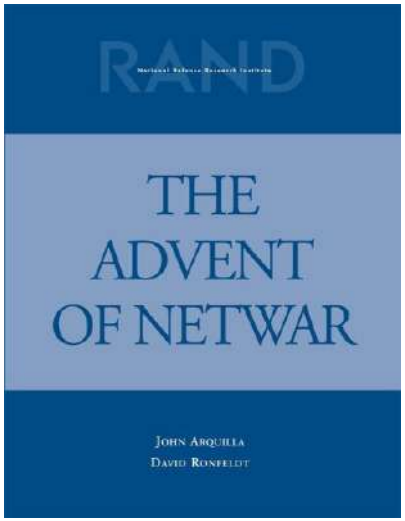


HUMAN DISEASE NETWORK



FIGHTING TERRORISM AND MILITARY

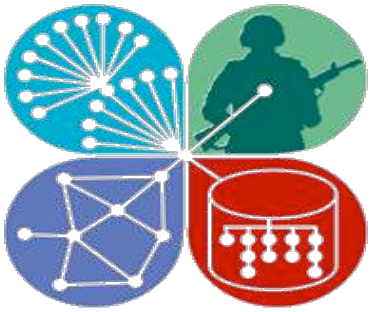




http://www.slate.com/id/2245232

FIGHTING TERRORISM AND MILITARY

Network Science Center
West Point

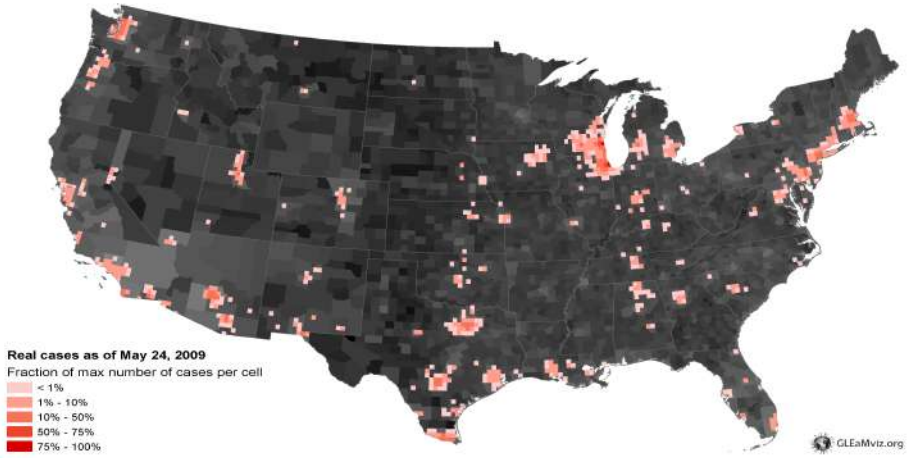


http://www.ns-cta.org/ns-cta-blog/

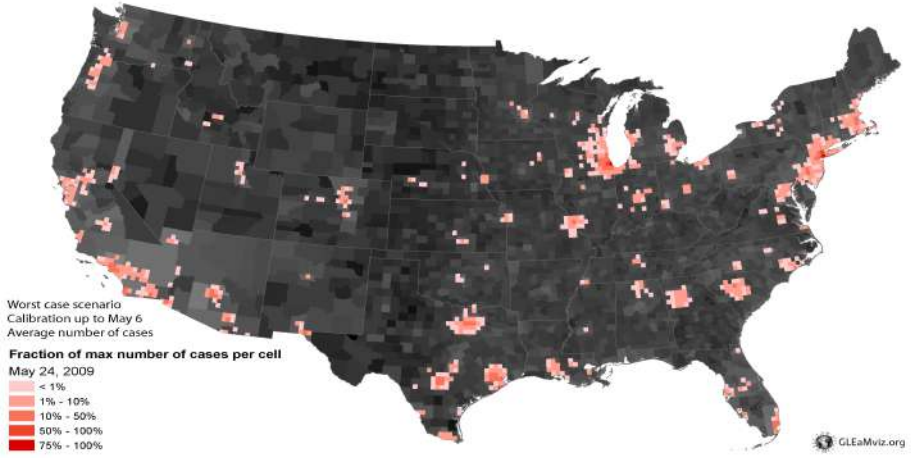
PREDICTING THE H1N1 PANDEMIA

PREDICTING THE H1N1 PANDEMIA

Real



Projected

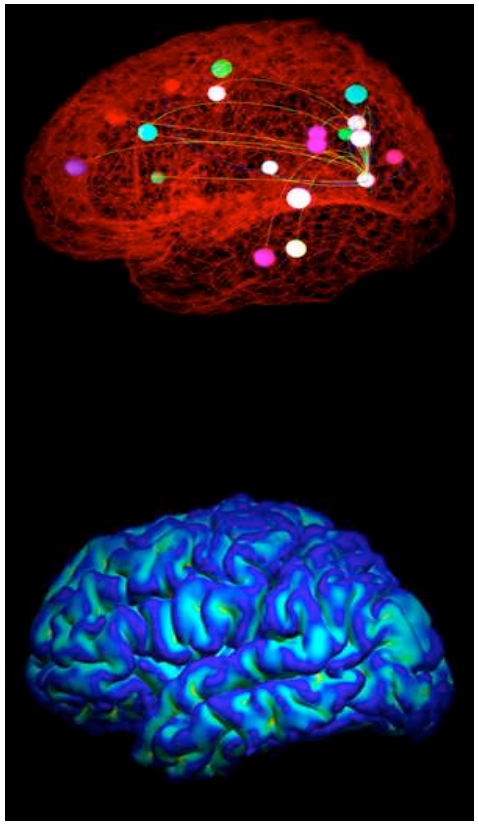


BRAIN RESEARCH

In September 2010 the National Institutes of Health awarded $40 million to researchers at Harvard, Washington University in St. Louis, the University of Minnesota and UCLA, to develop the technologies that could systematically map out brain circuits

The Human Connectome Project (HCP) with the ambitious goal to construct a map of the complete structural and functional neural connections in vivo within and across individuals

http://www.humanconnectomeproject.org/overview/

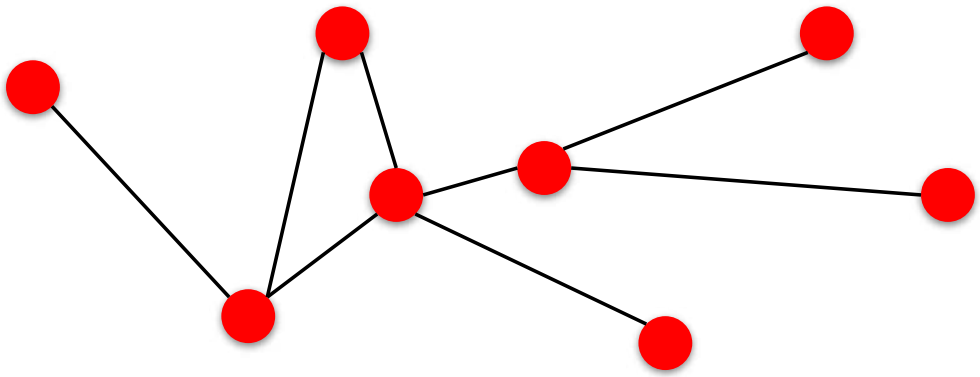


NETWORKS REALLY MATTER

- If you were to understand the spread of diseases, **can you do it without networks?**
- If you were to understand the WWW structure, searchability, etc, **hopeless without invoking the Web's topology**
- If you want to understand human diseases, **it is hopeless without considering the wiring diagram of the cell**

Networks and graphs

COMPONENTS OF A COMPLEX SYSTEM



- **components:** nodes, vertices N
- **interactions:** links, edges L
- **system:** network, graph (N,L)

NETWORKS OR GRAPHS?

network often refers to real systems

www, social network, metabolic network

Language: (Network, node, link)

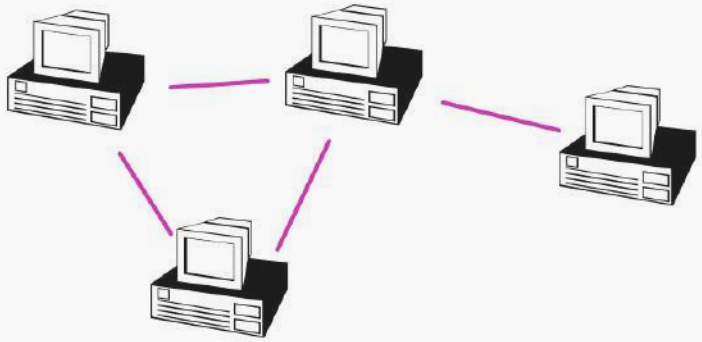
graph: mathematical representation of a network

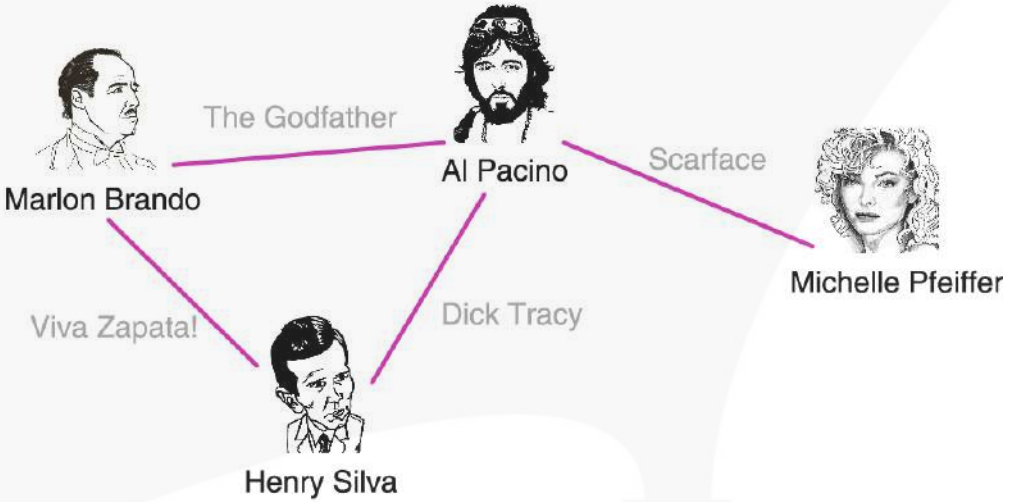
web graph, social graph (a Facebook term)

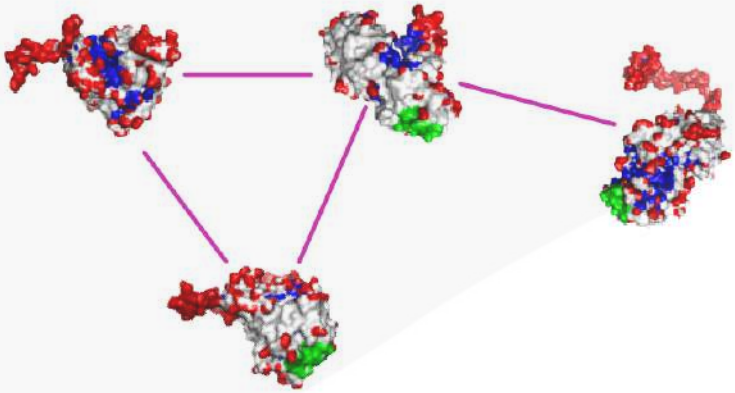
Language: (Graph, vertex, edge)

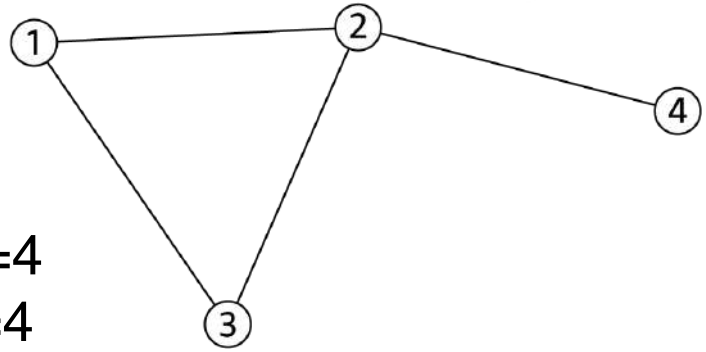
We will try to make this distinction whenever it is appropriate, but in most cases we will use the two terms interchangeably

A COMMON LANGUAGE







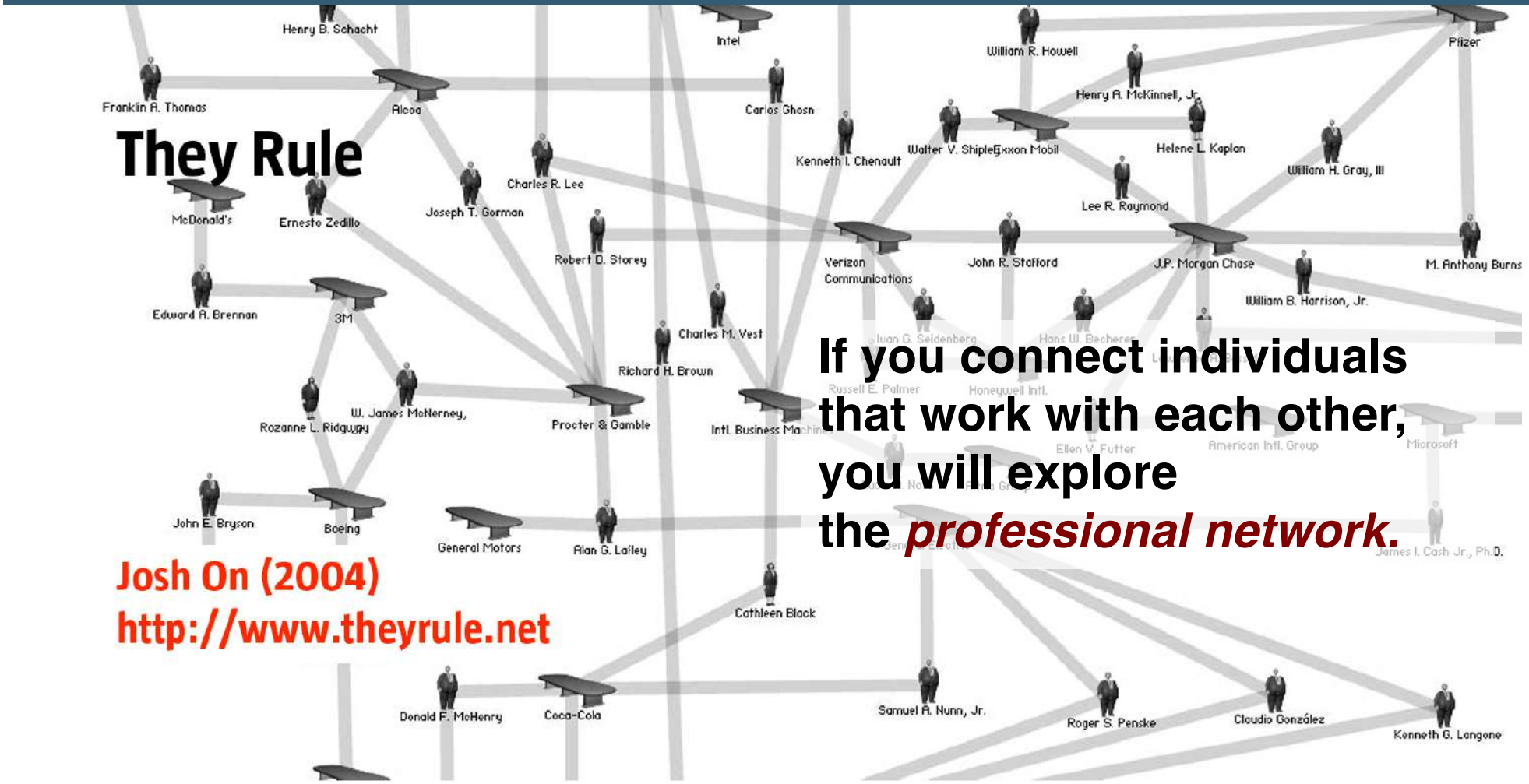


N=4
L=4

CHOOSING A PROPER REPRESENTATION

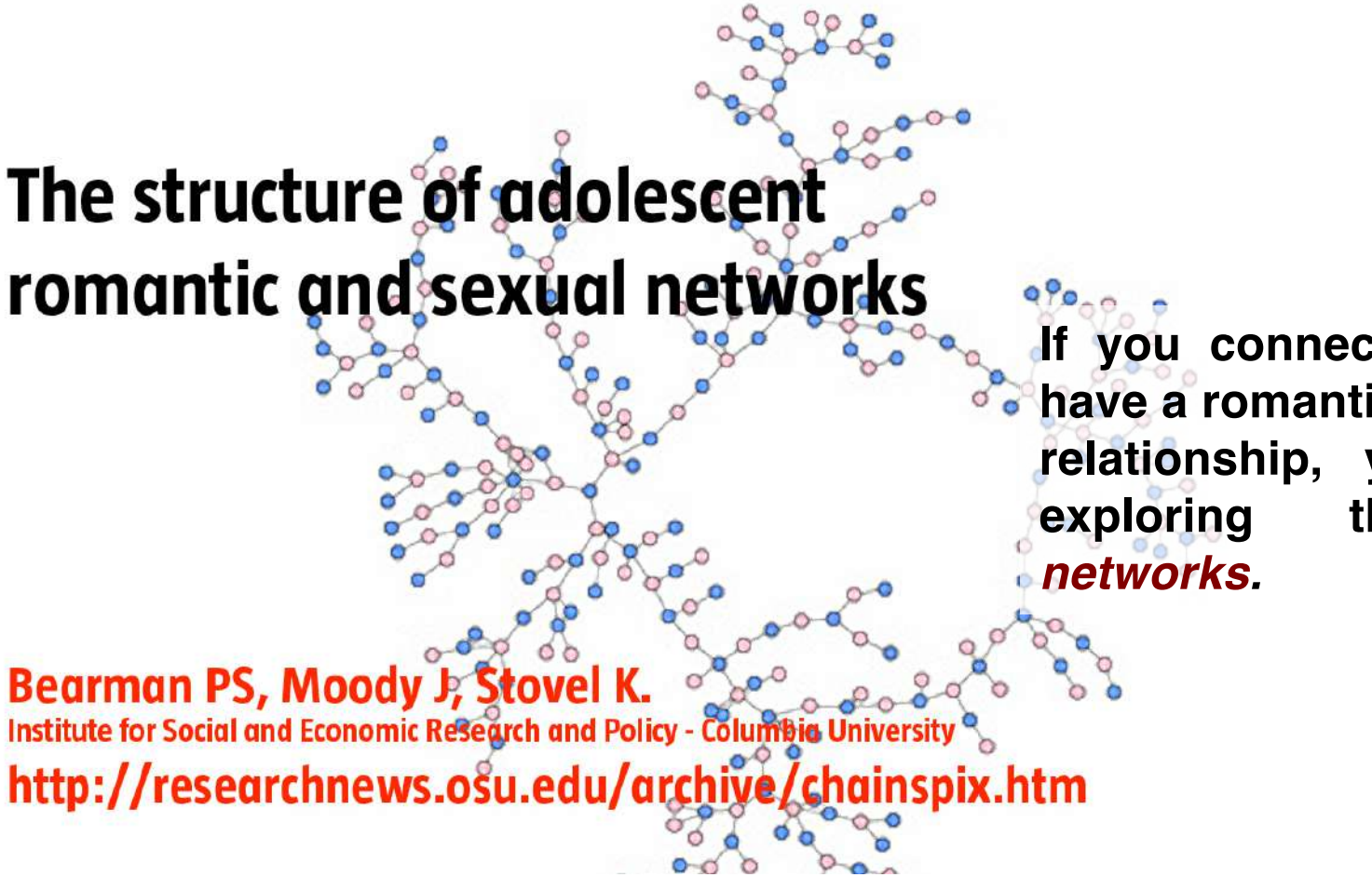
- The choice of the **proper network representation** determines our ability to use network theory successfully
- In some cases there is
  - a unique, unambiguous representation
  - the representation is by no means unique
  - for example, the way we assign the **links between a group of individuals** will **determine** the nature of the **question** we can study

CHOOSING A PROPER REPRESENTATION



If you connect individuals that work with each other, you will explore the *professional network.*

CHOOSING A PROPER REPRESENTATION



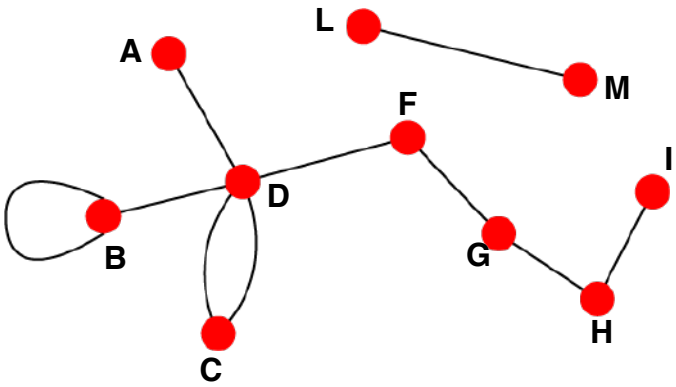
If you connect those that have a romantic and sexual relationship, you will be exploring the *sexual networks*.

UNDIRECTED VS. DIRECTED NETWORKS

Undirected

Links: undirected (*symmetrical*)

Graph:

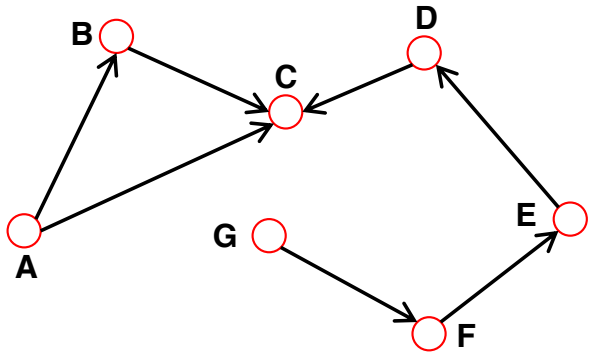


Undirected links :
coauthorship links
Actor network
protein interactions

Directed

Links: directed (*arcs*).

Digraph = directed graph:



An undirected link is the superposition of two opposite directed links.

Directed links :
URLs on the www
phone calls
metabolic reactions

REFERENCE NETWORKS

NETWORK	NODES	LINKS	DIRECTED UNDIRECTED	N	L
Internet	Routers	Internet connections	Undirected	192,244	609,066
WWW	Webpages	Links	Directed	325,729	1,497,134
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594
Mobile Phone Calls	Subscribers	Calls	Directed	36,595	91,826
Email	Email addresses	Emails	Directed	57,194	103,731
Science Collaboration	Scientists	Co-authorship	Undirected	23,133	93,439
Actor Network	Actors	Co-acting	Undirected	702,388	29,397,908
Citation Network	Paper	Citations	Directed	449,673	4,689,479
E. Coli Metabolism	Metabolites	Chemical reactions	Directed	1,039	5,802
Protein Interactions	Proteins	Binding interactions	Undirected	2,018	2,930

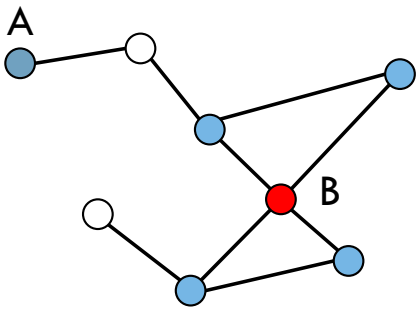
Degree, Average Degree and Degree Distribution

DEGREE OF A NODE

- The number of links it has to other nodes
  - The number of different individuals the person has talked to from her call graph
  - The number of citations a research paper gets in the citation network
- We denote k_i the degree of the ith node in the network

NODE DEGREES

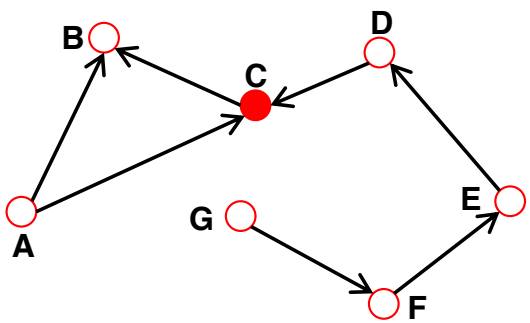
Undirected



Node degree: the number of links connected to the node.

$$k_A = 1 \qquad k_B = 4$$

Directed



In *directed networks* we can define an in-degree and out-degree.

The (total) degree is the sum of in- and out-degree.

$$k_C^{in} = 2 \qquad k_C^{out} = 1 \qquad k_C = 3$$

Source: a node with $k^{in} = 0$; Sink: a node with $k^{out} = 0$.

A BIT OF STATISTICS

BRIEF STATISTICS REVIEW

Four key quantities characterize a sample of N values $x_1, \ldots, x_N$:

Average (mean):

$$\langle x \rangle = \frac{x_1 + x_2 + \ldots + x_N}{N} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

The nth moment:

$$\langle x^n \rangle = \frac{x_1^n + x_2^n + \ldots + x_N^n}{N} = \frac{1}{N}\sum_{i=1}^{N} x_i^n$$

Standard deviation:

$$\sigma_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_i - \langle x \rangle\right)^2}$$

Distribution of x:

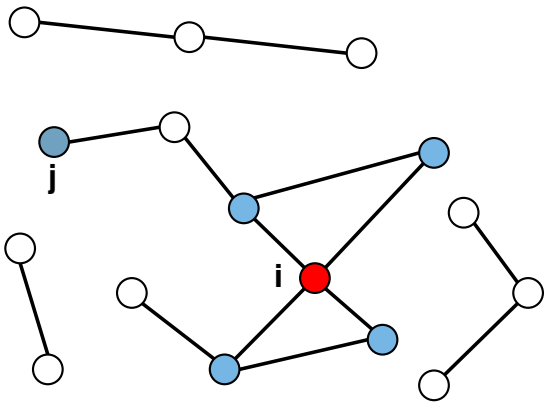
$$p_x = \frac{1}{N}\sum_i \delta_{x,x_i}$$

where p_x follows

$$\sum_i p_x = 1 \left(\int p_x \, dx = 1\right)$$

AVERAGE DEGREE

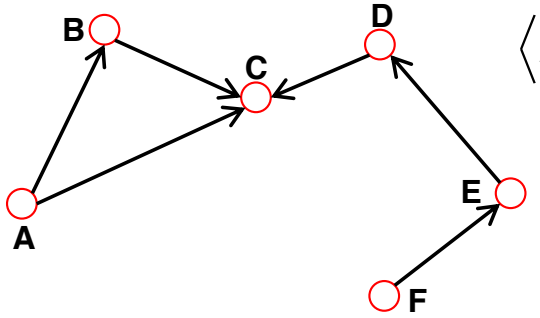
Undirected



$$\langle k \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i \qquad \langle k \rangle \equiv \frac{2L}{N}$$

N – the number of nodes in the graph

Directed



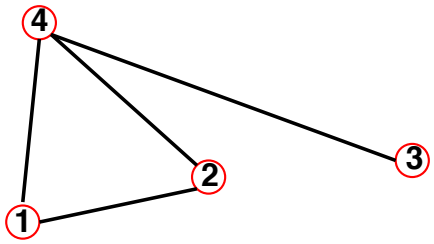
$$\langle k^{in} \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i^{in}, \quad \langle k^{out} \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i^{out}, \quad \langle k^{in} \rangle = \langle k^{out} \rangle$$

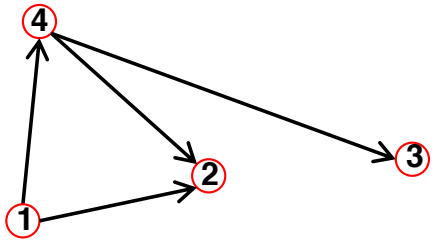
$$\langle k \rangle \equiv \frac{L}{N}$$

AVERAGE DEGREE

NETWORK	NODES	LINKS	DIRECTED UNDIRECTED	N	L	⟨k⟩
Internet	Routers	Internet connections	Undirected	192,244	609,066	6.33
WWW	Webpages	Links	Directed	325,729	1,497,134	4.60
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594	2.67
Mobile Phone Calls	Subscribers	Calls	Directed	36,595	91,826	2.51
Email	Email addresses	Emails	Directed	57,194	103,731	1.81
Science Collaboration	Scientists	Co-authorship	Undirected	23,133	93,439	8.08
Actor Network	Actors	Co-acting	Undirected	702,388	29,397,908	83.71
Citation Network	Paper	Citations	Directed	449,673	4,689,479	10.43
E. Coli Metabolism	Metabolites	Chemical reactions	Directed	1,039	5,802	5.58
Protein Interactions	Proteins	Binding interactions	Undirected	2,018	2,930	2.90

ADJACENCY MATRIX





$A_{ij} = 1$ *if there is a link between node i and j*

$A_{ij} = 0$ if nodes *i* and *j* are not connected to each other.

$$A_{ij} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \qquad A_{ij} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Note that for a directed graph (right) the matrix is not symmetric.

$A_{ij} = 1$ if there is a link pointing from node *j* and *i*

$A_{ij} = 0$ if there is no link pointing from *j* to *i*.

Paths, connectedness, clustering coefficient, other concepts

other necessary concepts for doing graph analytics

GRAPHOLOGY: REAL NETWORKS — MANY FEATURES

WWW > **d**irected multigraph with self-interactions

Protein Interactions > **u**ndirected unweighted with self-interactions

Collaboration network > **u**ndirected multigraph or weighted.

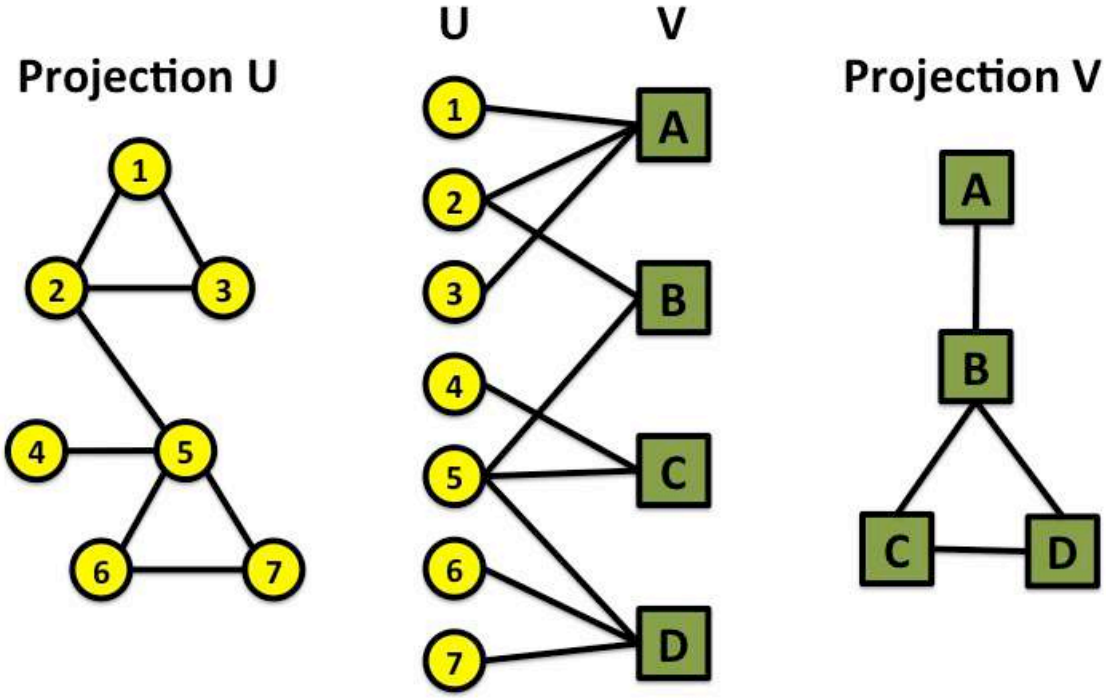
Mobile phone calls > directed, weighted.

Facebook Friendship links > **u**ndirected,
unweighted.

BIPARTITE NETWORKS

BIPARTITE GRAPH

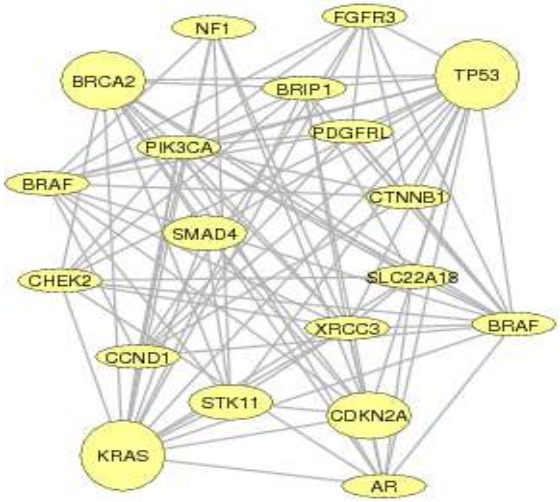
bipartite graph (or **bigraph**) is a graph whose nodes can be divided into two disjoint sets *U* and *V* such that every link connects a node in *U* to one in *V*; that is, *U* and *V* are independent sets.



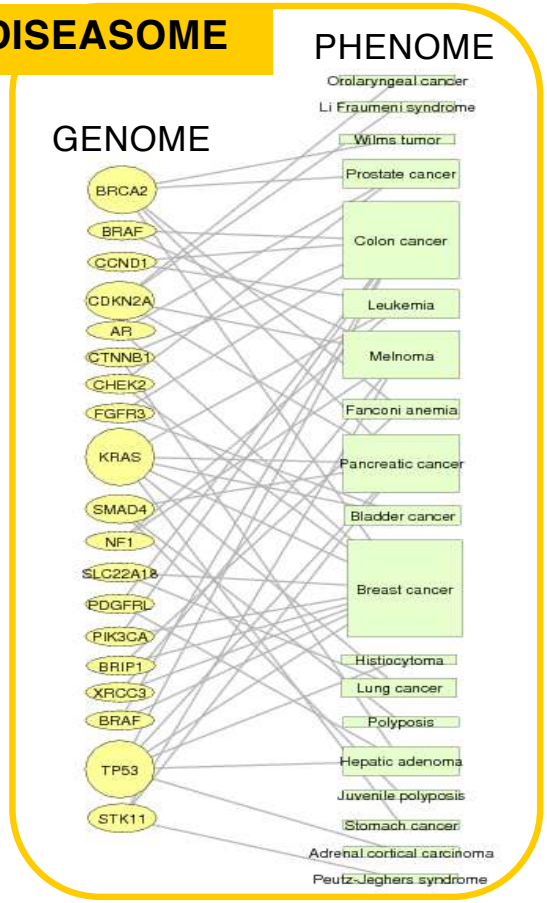
Examples:

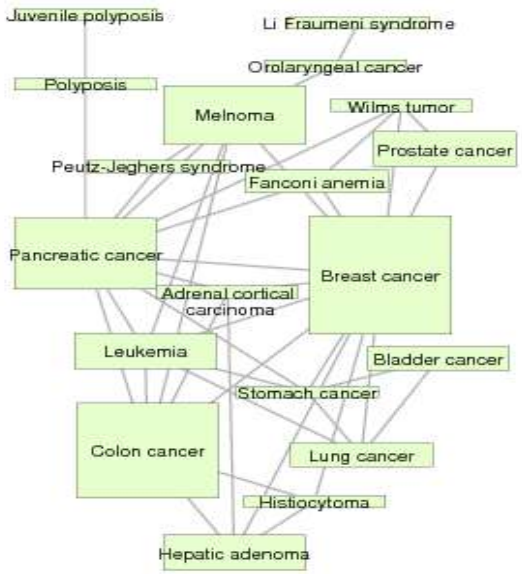
Hollywood actor network
Collaboration networks
Disease network (diseasome)

GENE NETWORK — DISEASE NETWORK



Gene network

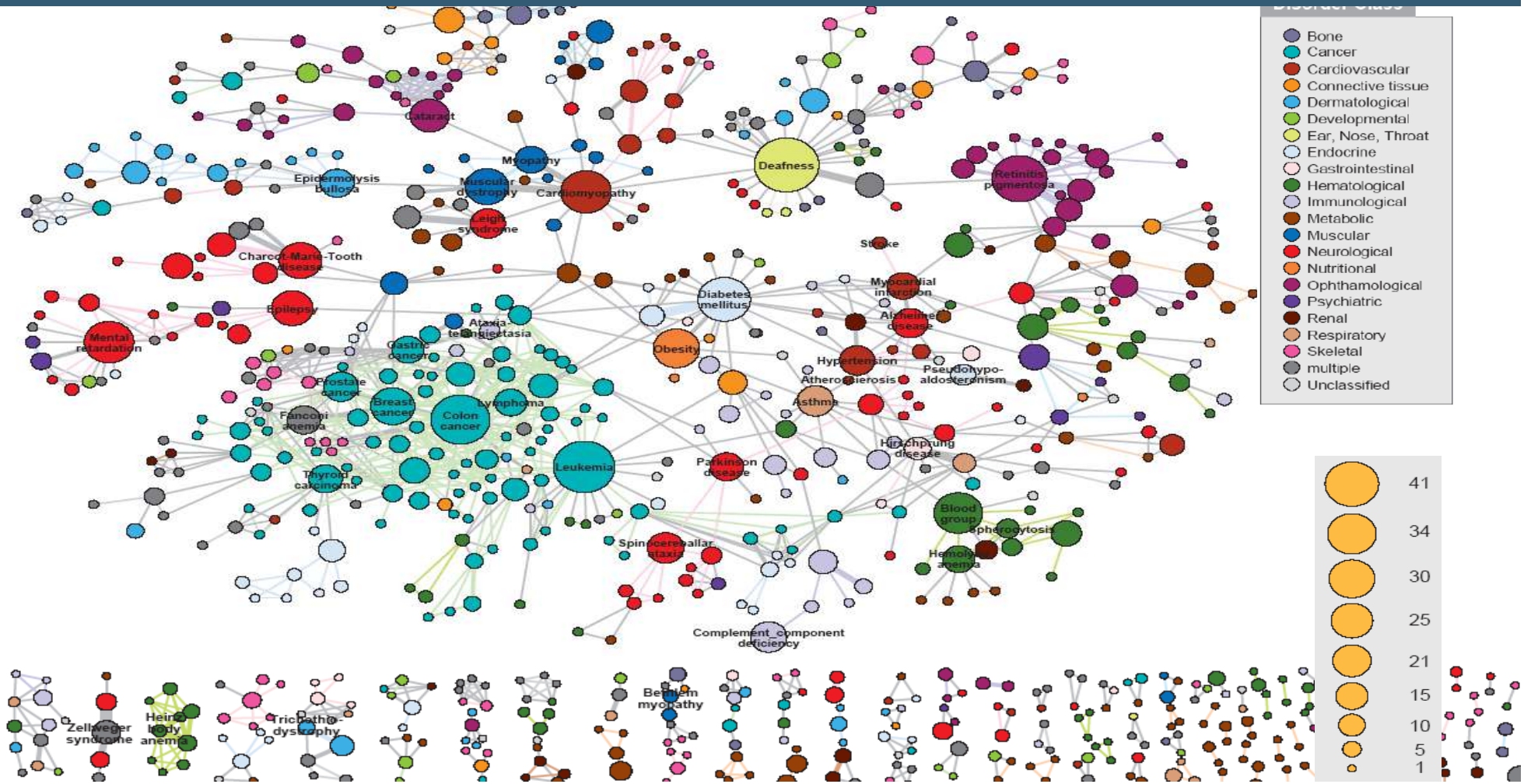




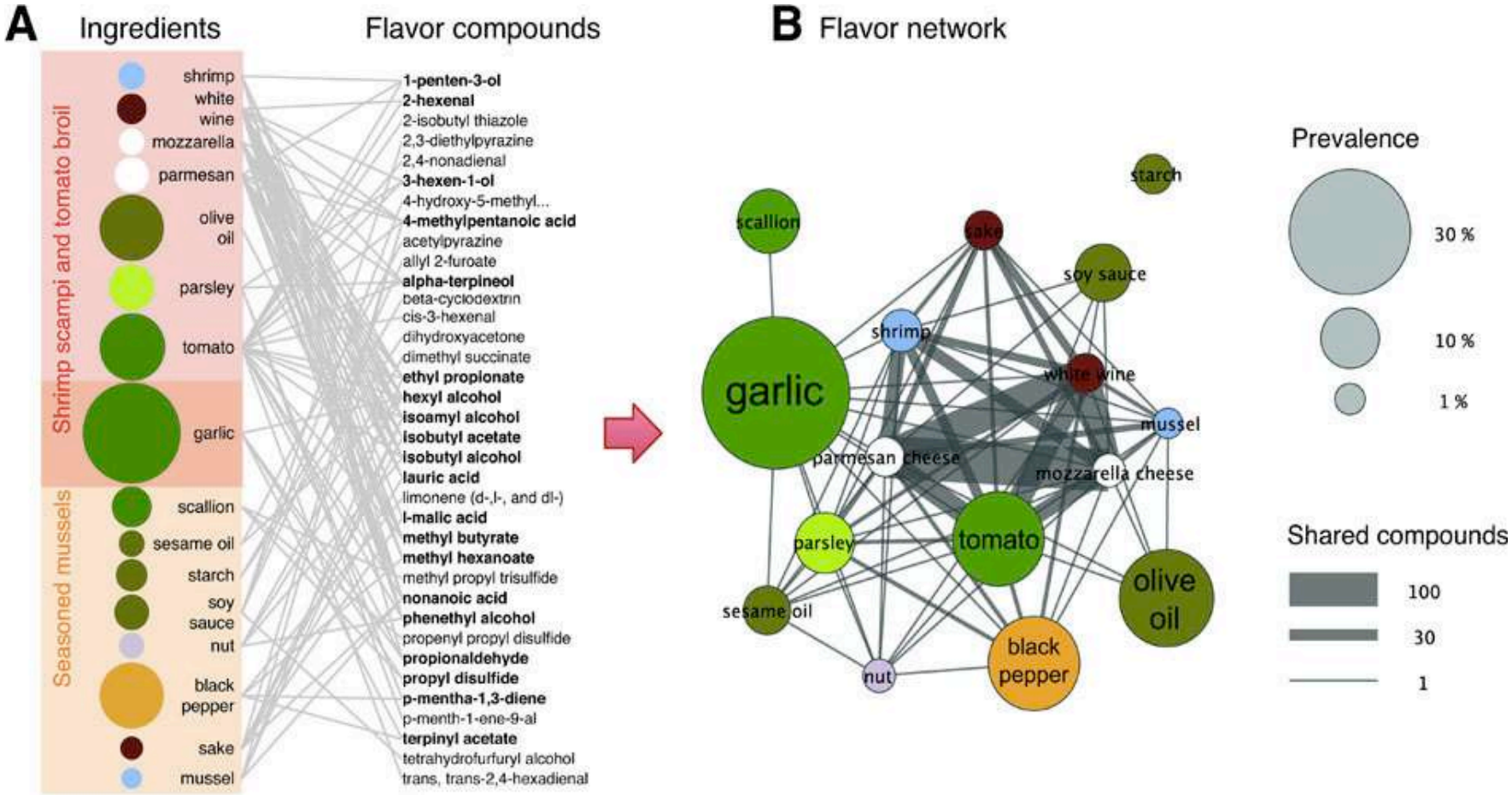
Disease network

Goh, Cusick, Valle, Childs, Vidal & Barabási, PNAS (2007)

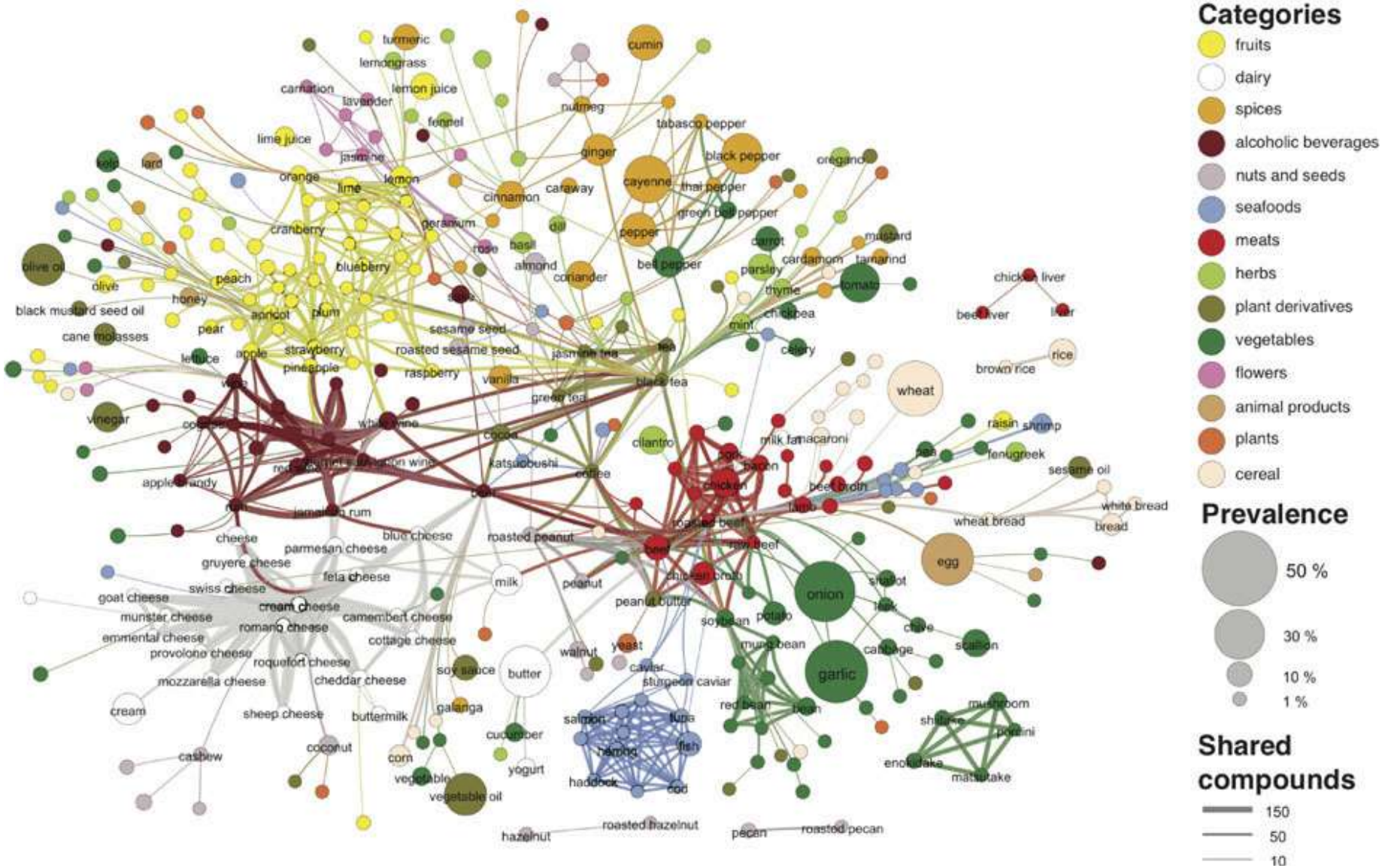
HUMAN DISEASE NETWORK



INGREDIENT — FLAVOR BIPARTITE NETWORK



Y.-Y. Ahn, S. E. Ahnert, J. P. Bagrow, A.-L. Barabási Flavor network and the principles of food pairing , Scientific Reports 196, (2011).


Categories
fruits
dairy
spices
alcoholic beverages
nuts and seeds
seafoods
meats
herbs
plant derivatives
vegetables
flowers
animal products
plants
cereal
Prevalence
50 %
30 %
10 %
1 %
Shared compounds
150
50
10
turmeric
lemongrass
cumin
carnation
lavender
lemon juice
nutmeg
fennel
tabasco pepper
lime juice
jasmine
ginger
black pepper
oregano
orange
lime
lemon
cinnamon
caraway
cayenne
thai pepper
green bell pepper
cranberry
geranium
dill
pepper
mustard
rose
basil
carrot
olive oil
almond
bell pepper
cardamom
tamarind
blueberry
coriander
parsley
tomato
olive
peach
thyme
chicken liver
honey
black mustard seed oil
apricot
plum
chickpea
beef liver
liver
pear
mint
cane molasses
sesame seed
celery
lettuce
apple
strawberry
roasted sesame seed
jasmine tea
tea
rice
pineapple
raspberry
vanilla
brown rice
black tea
wheat
green tea
vinegar
white wine
raisin
shrimp
cocoa
cilantro
milk
macaroni
red wine
katsuobushi
fenugreek
sesame oil
apple brandy
coffee
pork
bacon
chicken
beef broth
beer
jamaican rum
lamb
white bread
wheat bread
bread
cheese
blue cheese
roasted peanut
parmesan cheese
beef
raw beef
gruyere cheese
egg
fela cheese
milk
peanut
chicken broth
shallot
swiss cheese
onion
goat cheese
cream cheese
peanut butter
leek
munster cheese
camembert cheese
soybean
potato
romano cheese
chive
emmental cheese
cottage cheese
mung bean
cabbage
scallion
provolone cheese
walnut
yeast
roquefort cheese
soy sauce
butter
caviar
garlic
mozzarella cheese
cheddar cheese
sturgeon caviar
red bean
bean
cream
sheep cheese
buttermilk
galanga
salmon
tuna
mushroom
shiitake
cucumber
herring
fish
porcini
cashew
coconut
corn
yogurt
enokitake
vegetable
haddock
cod
matsutake
vegetable oil
hazelnut
roasted hazelnut
pecan
roasted pecan

Network models

ERDŐS–RÉNYI RANDOM GRAPH MODEL

Used for generating random graphs in which edges are set between nodes with equal probabilities

- prove the existence of graphs satisfying various properties, or
- provide a rigorous definition of what it means for a property to hold for almost all graphs.

Generating an Erdős–Rényi model

- the number of nodes in the graph generated as N
- the probability that a link should be formed between any two nodes as p
- A constant $\langle k \rangle$ may derived from these two components with the formula
  - $\langle k \rangle = 2 \cdot E / N = p \cdot (N - 1)$, where
  - E is the expected number of edges

http://igraph.org/r/doc/erdos.renyi.game.html

WATTS-STROGATZ SMALL WORLD MODEL

A random graph generation model that produces graphs with small-world properties

An initial lattice structure is used to generate a Watts-Strogatz model.

- Each node in the network is initially linked to its <k> closest neighbours
- Another parameter is specified as the rewiring probability:
  - Each edge has a probability p that it will be rewired to the graph as a random edge.
  - The expected number of rewired links in the model is pE = pN<k>/2.

http://www.mathworks.com/help/matlab/math/build-watts-strogatz-small-world-graph-model.html

BARABÁSI–ALBERT (BA) PREFERENTIAL ATTACHMENT MODEL

Random network model used to demonstrate a preferential attachment

- "rich-get-richer" effect
- An edge is most likely to attach to nodes with higher degrees

The network begins with an initial network of m_0 nodes

- $m_0 \geq 2$
- the degree of each node in the initial network should be at least 1
- otherwise it will always remain disconnected from the rest of the network

New nodes are added to the network one at a time.

- Each new node is connected to m existing nodes
- With a probability that is proportional to the number of links that the existing nodes already have

BARABÁSI–ALBERT (BA) PREFERENTIAL ATTACHMENT MODEL

Random network model used to demonstrate a preferential attachment

- "rich-get-richer" effect

Some remarks

- Heavily linked nodes ("hubs") tend to quickly accumulate even more links
- Nodes with only a few links are unlikely to be chosen as the destination for a new link
- New nodes have a "preference" to attach themselves to the already heavily linked nodes

New nodes are added to the network one at a time

- Each new node is connected to m existing nodes
- With a probability that is proportional to the number of links that the existing nodes already have

Network analysis

NETWORK ANALYSIS

Social network analysis

- Examines the structure of relationships between social entities
- Entities are often people, but may also be groups, organizations nation states, web sites, scholarly publications

Dynamic network analysis:

- examines the shifting structure of relationships among different classes of entities in complex socio-technical systems effects
- reflects social stability and changes such as the emergence of new groups, topics, and leaders

NETWORK ANALYSIS

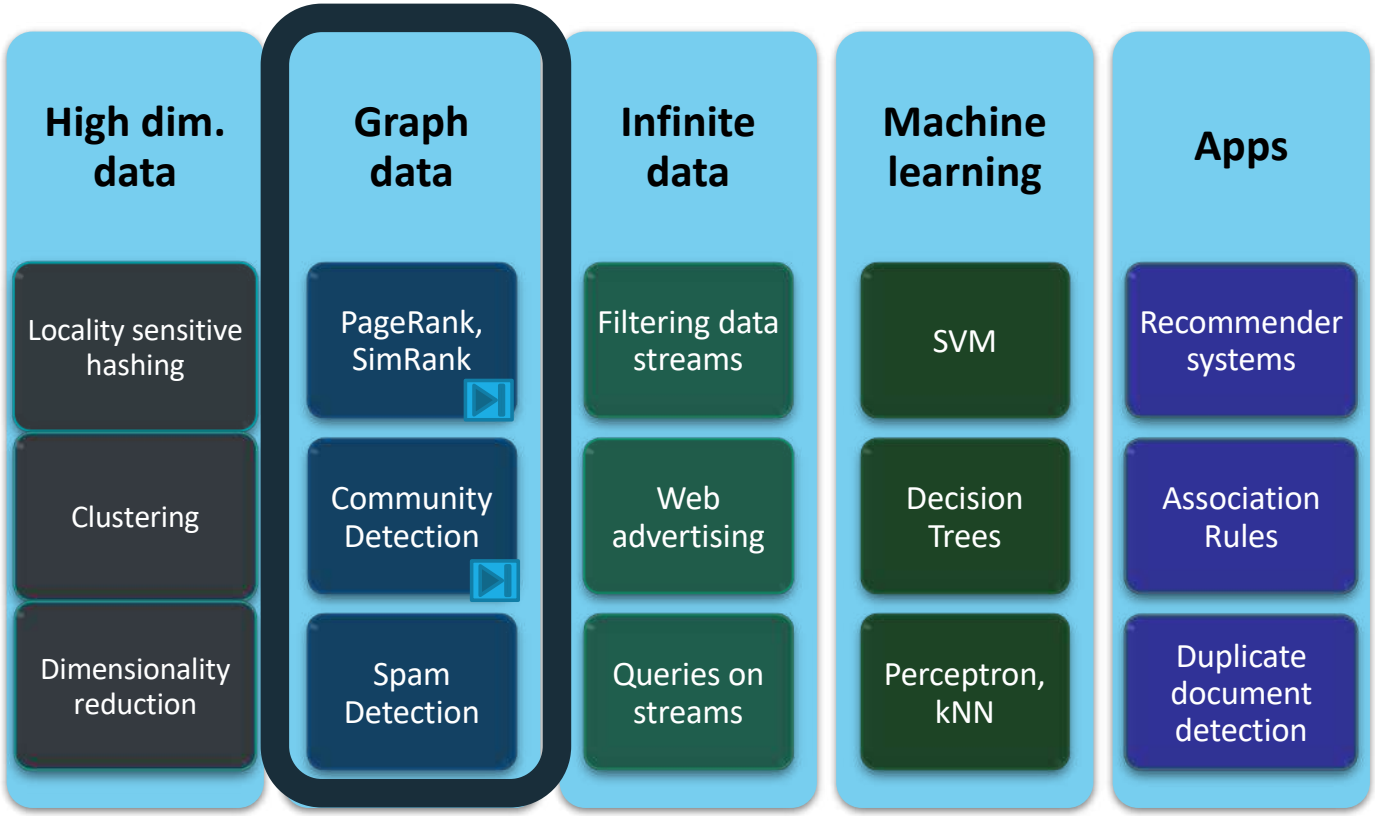
Biological network analysis

- closely related to social network analysis
- focusing on local patterns in the network
  - network motifs are small sub-graphs that are over-represented in the network.
  - analysis of biological networks has led to the development of network medicine

Link analysis

- Exploring associations between objects.
- examining the addresses of suspects and victims, the telephone numbers they have dialled and financial transactions that they have partaken in during a given timeframe, and the familial relationships between these subjects as a part of police investigation.
- Link analysis here provides the crucial relationships and associations between very many objects of different types that are not apparent from isolated pieces of information
- Pandemic analysis, Web link analysis, Page Rank, ..

Analysis of large graphs


High dim. data
Locality sensitive hashing
Clustering
Dimensionality reduction
Graph data
PageRank, SimRank
Community Detection
Spam Detection
Infinite data
Filtering data streams
Web advertising
Queries on streams
Machine learning
SVM
Decision Trees
Perceptron, kNN
Apps
Recommender systems
Association Rules
Duplicate document detection

WEB AS A GRAPH

Web as a directed graph:

- **Nodes: Webpages**
- **Edges: Hyperlinks**

I teach a class on Networks.

CS224W: Classes are in the Gates building

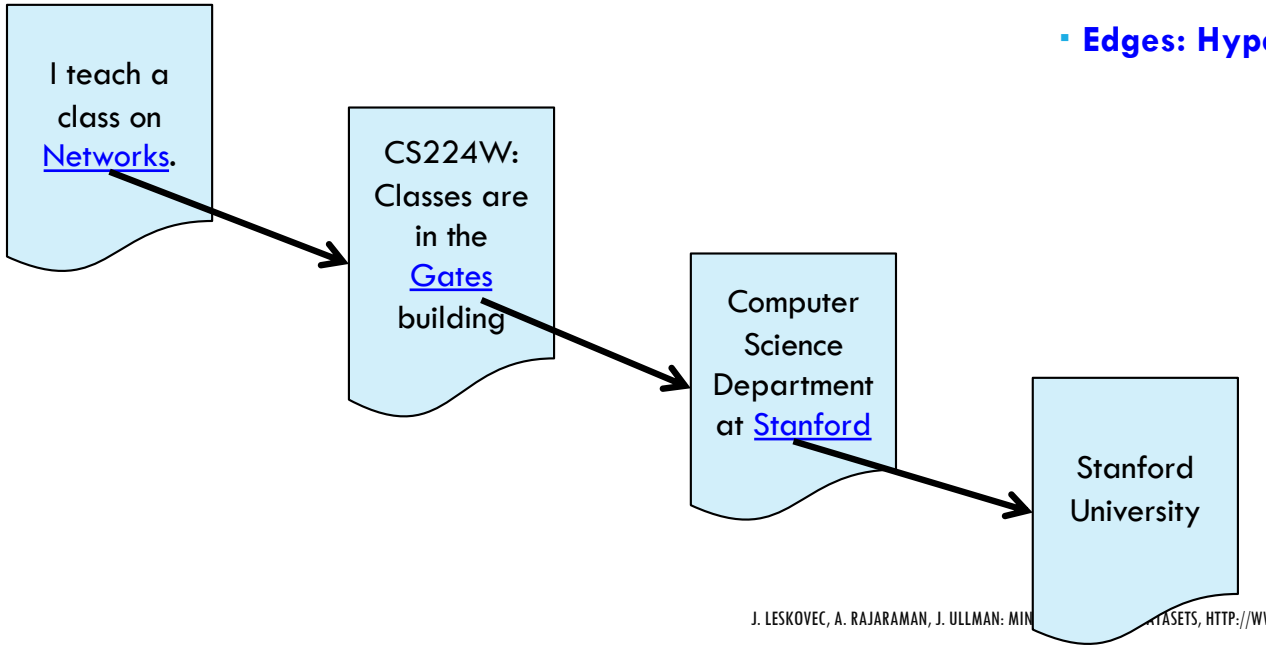
Computer Science Department at Stanford

Stanford University

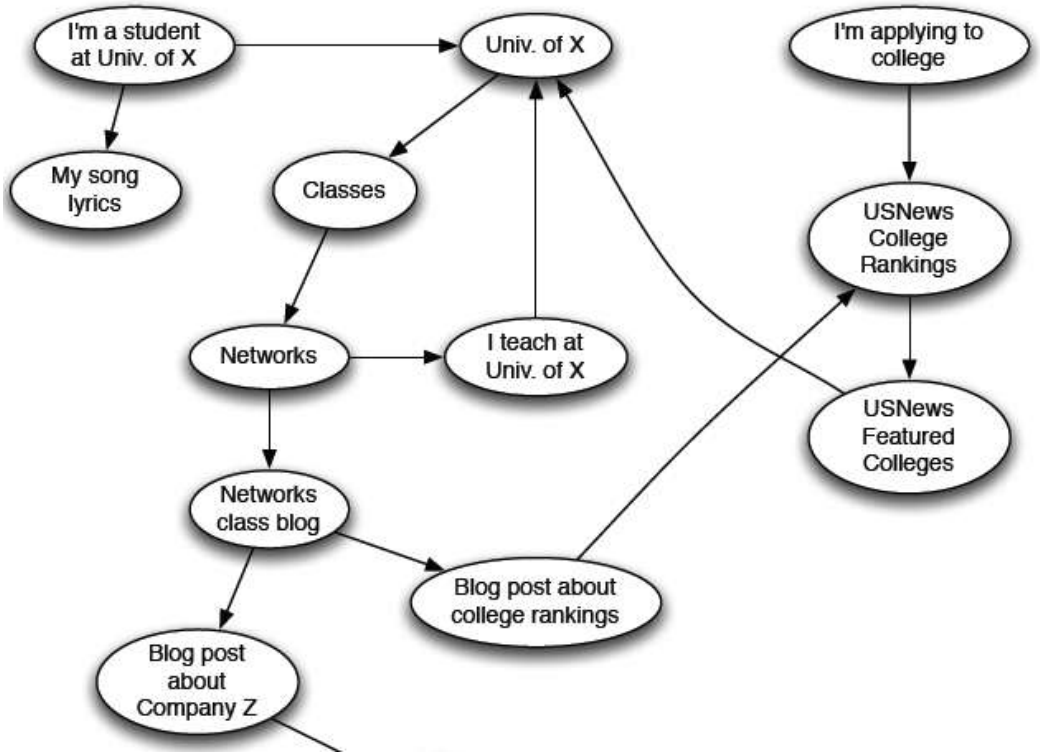
WEB AS A GRAPH

Web as a directed graph:

- **Nodes: Webpages**
- **Edges: Hyperlinks**



WEB AS A DIRECTED GRAPH



BROAD QUESTION

How to organize the Web?

First try: Human curated
Web directories

- Yahoo, DMOZ, LookSmart

Second try: Web Search

- **Information Retrieval** investigates: Find relevant docs in a small and trusted set
  - Newspaper articles, Patents, etc.
- **But:** Web is **huge**, full of untrusted documents, random things, web spam, etc.

WEB SEARCH: 2 CHALLENGES

2 challenges of web search:

(1) Web contains many sources of information
Who to "trust"?

- **Trick:** Trustworthy pages may point to each other!

(2) What is the "best" answer to query "newspaper"?

- No single right answer
- **Trick:** Pages that actually know about newspapers might all be pointing to many newspapers

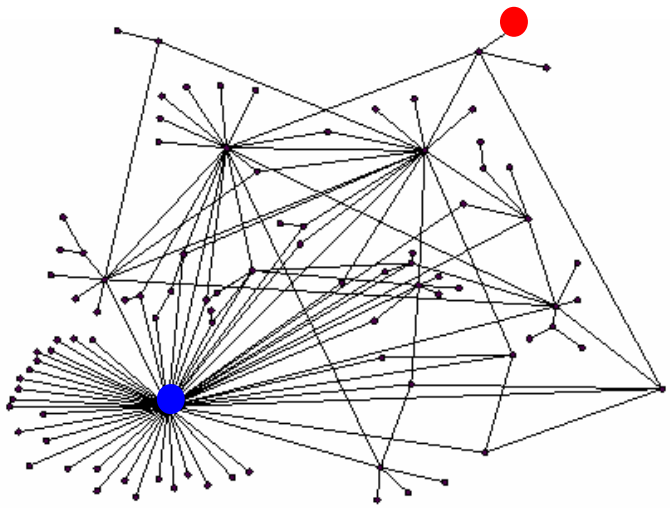
RANKING NODES ON THE GRAPH

All web pages are not equally "important"

www.joe-schmoe.com vs. www.stanford.edu

There is large diversity in the web-graph node connectivity

Let's rank the pages by the link structure!



LINK ANALYSIS ALGORITHMS

Link Analysis approaches for computing **importance** of nodes in a graph:

- Page Rank
- Topic-Specific (Personalized) Page Rank
- Web Spam Detection Algorithms

Map reduced K-means

Prajesh P Anchalia, Anjan K Koundinya, Srinath N K, MapReduce Design of K-Means Clustering Algorithm, IEEE, 2013

Implementation 3: Spark platform

SPARK

Fast, Interactive, Language-Integrated Cluster Computing

Matei Zaharia, Mosharaf Chowdhury, Tathagata Das,
Ankur Dave, Justin Ma, Murphy McCauley, Michael Franklin,
Scott Shenker, Ion Stoica

www.spark-project.org

amplab
UC BERKELEY

PROJECT GOALS

Extend the MapReduce model to better support two common classes of analytics apps:

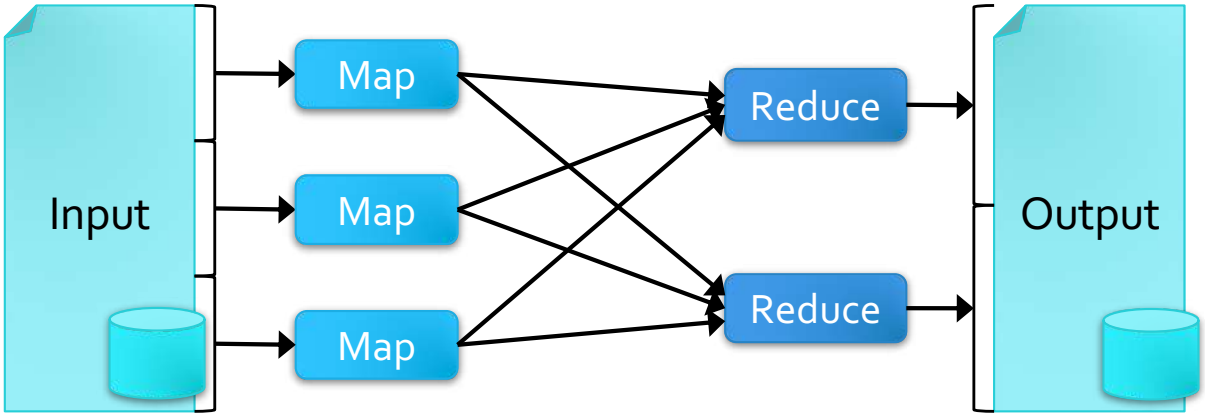
- **Iterative** algorithms (machine learning, graphs)
- **Interactive** data mining

Enhance programmability:

- Integrate into Scala programming language
- Allow interactive use from Scala interpreter

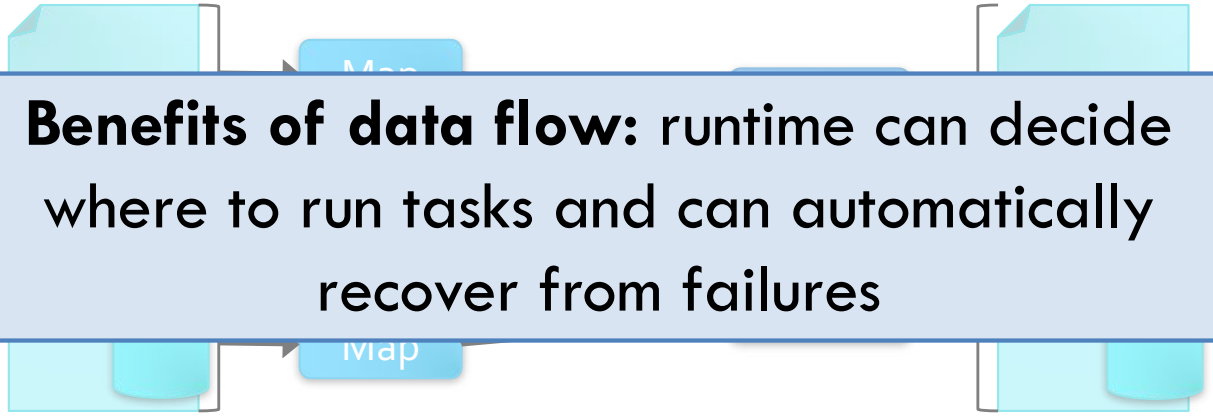
MOTIVATION

Most current cluster programming models are based on *acyclic data flow* from stable storage to stable storage



MOTIVATION

Most current cluster programming models are based on *acyclic data flow* from stable storage to stable storage



Benefits of data flow: runtime can decide where to run tasks and can automatically recover from failures

MOTIVATION

Acyclic data flow is inefficient for applications that repeatedly reuse a *working set* of data:

- **Iterative** algorithms (machine learning, graphs)
- **Interactive** data mining tools (R, Excel, Python)

With current frameworks, apps reload data from stable storage on each query

SOLUTION: RESILIENT DISTRIBUTED DATASETS (RDDS)

Allow apps to keep working sets in memory for efficient reuse

Retain the attractive properties of MapReduce

- Fault tolerance, data locality, scalability

Support a wide range of applications

SPARK OPERATIONS

Transformations
(define a new RDD)
	`map`
	`filter`
	`sample`
	`groupByKey`
	`reduceByKey`
	`sortByKey`
		`flatMap`
		`union`
		`join`
		`cogroup`
		`cross`
		`mapValues`

Actions
(return a result to driver program)
	`collect`
	`reduce`
	`count`
	`save`
	`lookupKey`	

OUTLINE

Spark programming model

Implementation

User applications

PROGRAMMING MODEL

Resilient distributed datasets (RDDs)

- Immutable, partitioned collections of objects
- Created through parallel *transformations* (map, filter, groupBy, join, …) on data in stable storage
- Can be *cached* for efficient reuse

Actions on RDDs

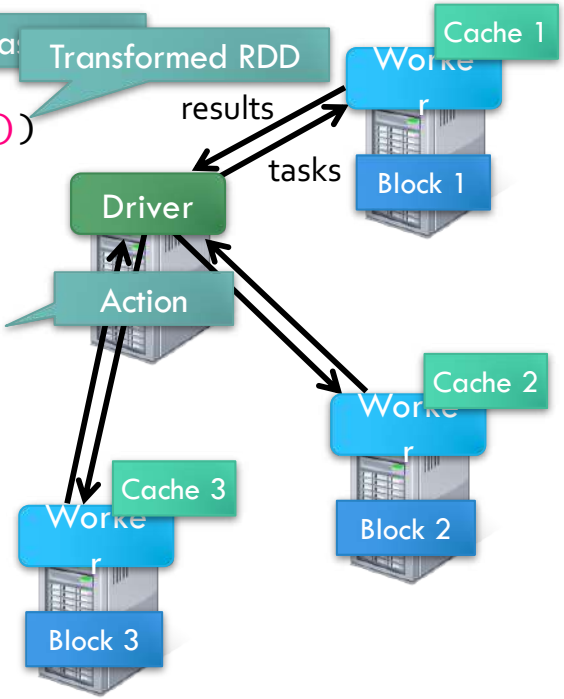
- Count, reduce, collect, save, …

EXAMPLE: LOG MINING

Load error messages from a log into memory, then interactively search for various patterns

```
lines = spark.textFile("hdfs://...")
errors = lines.filter(_.startsWith("ERROR"))
messages = errors.map(_.split('\t')(2))
cachedMsgs = messages.cache()

cachedMsgs.filter(_.contains("foo")).count
cachedMsgs.filter(_.contains("bar")).count
. . .
```



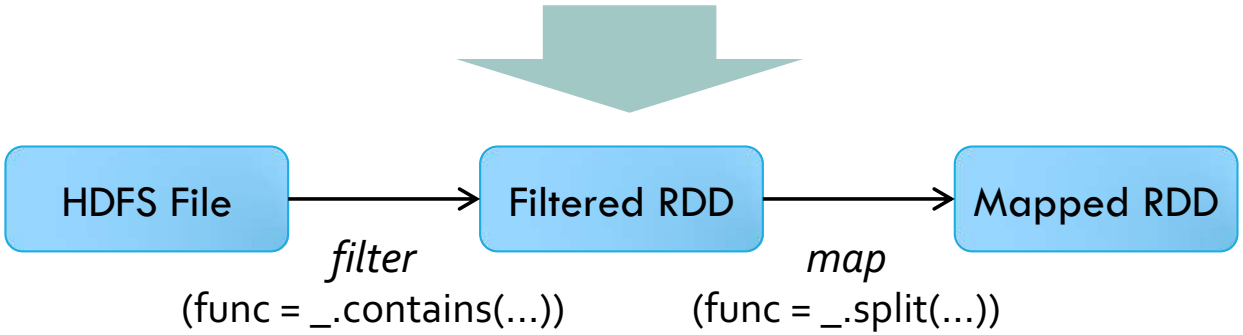
Result: scaled to 1 TB data in 5-7 sec
(vs 170 sec for on-disk data)

RDD FAULT TOLERANCE

RDDs maintain *lineage* information that can be used to reconstruct lost partitions

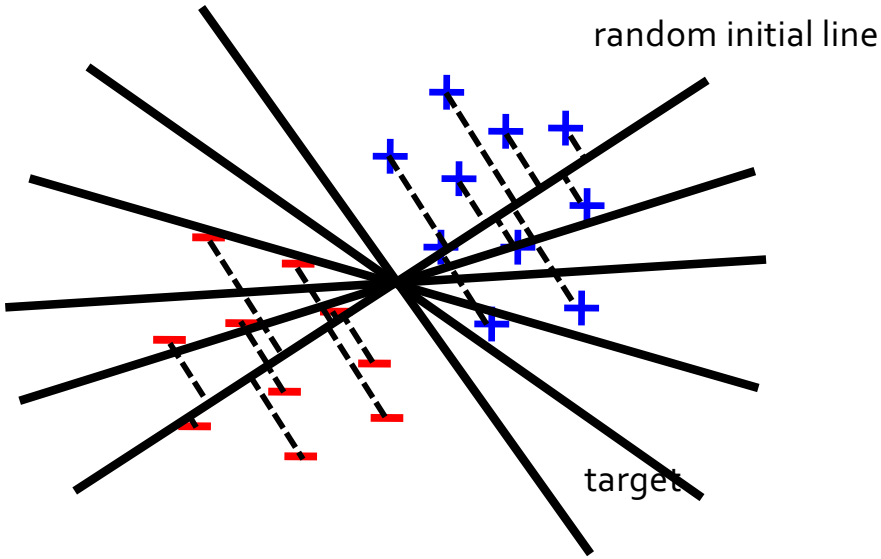
Ex:

```
messages = textFile(...).filter(_.startsWith("ERROR"))
                         .map(_.split('\t')(2))
```



EXAMPLE: LOGISTIC REGRESSION

Goal: find best line separating two sets of points



EXAMPLE: LOGISTIC REGRESSION

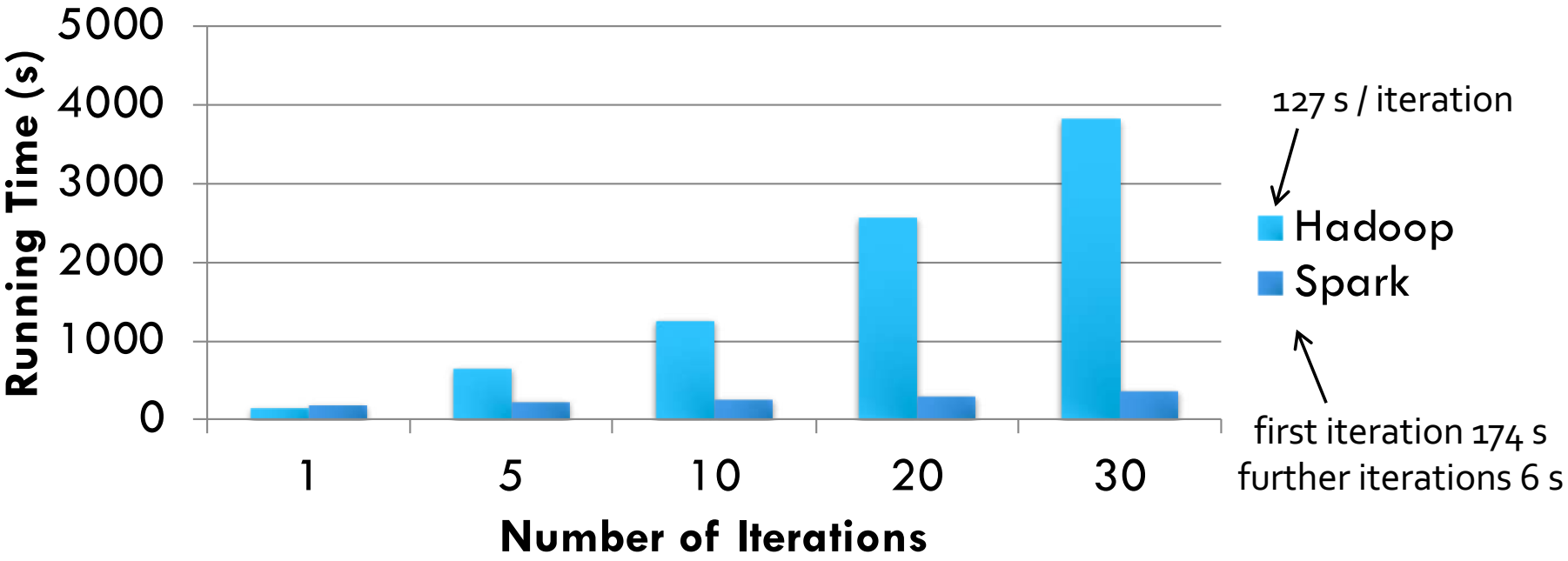
```
val data = spark.textFile(...).map(readPoint).cache()

var w = Vector.random(D)

for (i <- 1 to ITERATIONS) {
  val gradient = data.map(p =>
    (1 / (1 + exp(-p.y*(w dot p.x))) - 1) * p.y * p.x
  ).reduce(_ + _)
  w -= gradient
}

println("Final w: " + w)
```

LOGISTIC REGRESSION PERFORMANCE



This is for a 29 GB dataset on 20 EC2 m1.xlarge machines (4 cores each)

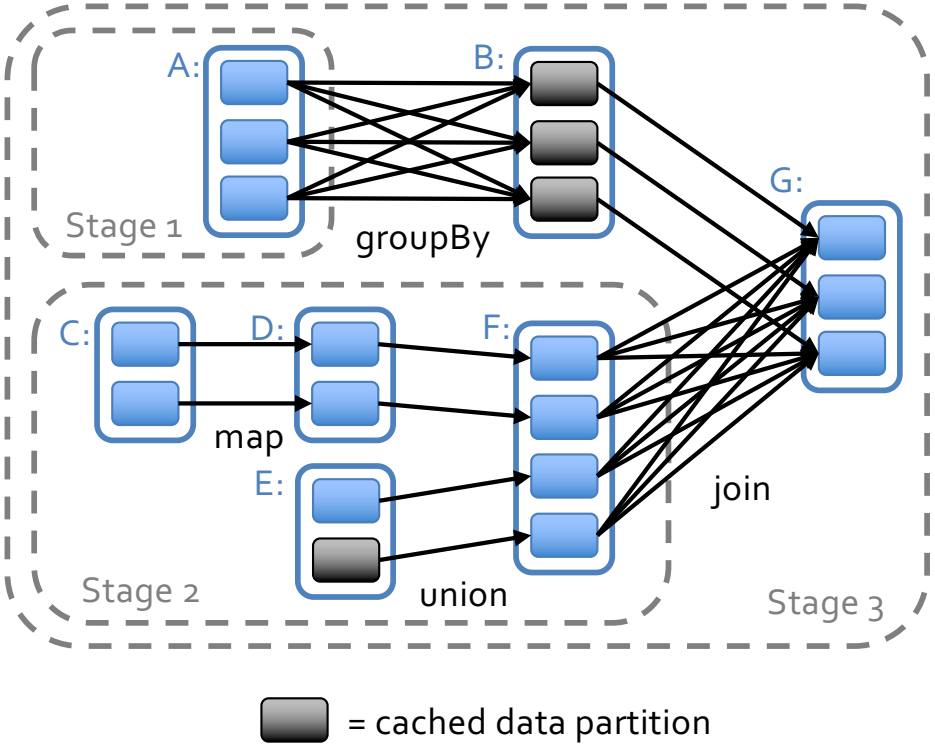
SPARK SCHEDULER

Dryad-like DAGs

Pipelines functions within a stage

Cache-aware work reuse & locality

Partitioning-aware to avoid shuffles



CONCLUSION

Spark provides a simple, efficient, and powerful programming model for a wide range of apps

Download our open source release:

www.spark-project.org

matei@berkeley.edu

RELATED WORK

DryadLINQ, FlumeJava

- Similar "distributed collection" API, but cannot reuse datasets efficiently *across* queries

Relational databases

- Lineage/provenance, logical logging, materialized views

GraphLab, Piccolo, BigTable, RAMCloud

- Fine-grained writes similar to distributed shared memory

Iterative MapReduce (e.g. Twister, HaLoop)

- Implicit data sharing for a fixed computation pattern

Caching systems (e.g. Nectar)

- Store data in files, no explicit control over what is cached

Let's dive on Spark for executing and analyzing K-Means

https://databricks.com/blog/2015/01/28/introducing-streaming-k-means-in-spark-1-2.html

hands on

Genoveva Vargas-Solar

CR1, CNRS, LIG-LAFMIA

Genoveva.Vargas@imag.fr

http://vargas-solar.com/big-linked-data-keystone/

Page rank

LINKS AS VOTES

Idea: Links as votes

- **Page is more important if it has more links**
  - In-coming links? Out-going links?

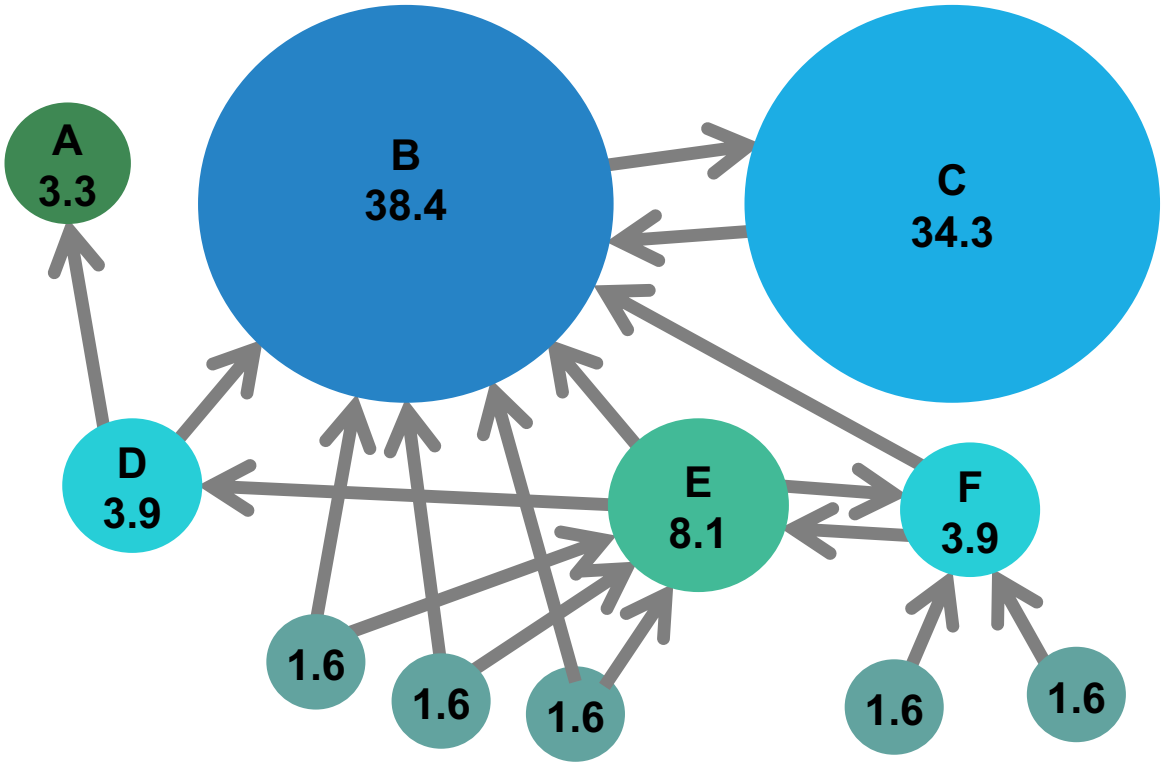
Think of in-links as votes:

- www.stanford.edu has 23,400 in-links
- www.joe-schmoe.com has 1 in-link

Are all in-links are equal?

- **Links from important pages count more**
- Recursive question!

EXAMPLE: PAGERANK SCORES



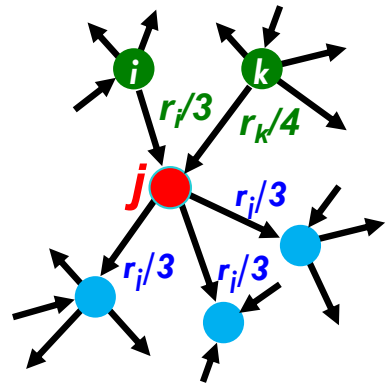
SIMPLE RECURSIVE FORMULATION

Each link's vote is proportional to the **importance** of its source page

If page ***j*** with importance r_j has ***n*** out-links, each link gets r_j / n votes

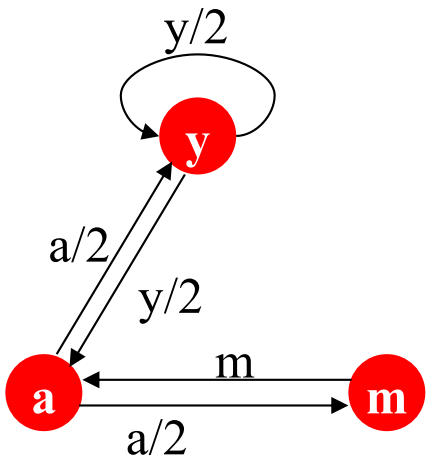
Page ***j***'s own importance is the sum of the votes on its in-links

$$r_j = r_i/3 + r_k/4$$



PAGERANK: THE "FLOW" MODEL

The web in 1839



A "vote" from an important page is worth more

A page is important if it is pointed to by other important pages

Define a "rank" r_j for page j

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i … out-degree of node i

"Flow" equations:

$$r_y = r_y/2 + r_a/2$$

$$r_a = r_y/2 + r_m$$

$$r_m = r_a/2$$

PAGERANK: THREE QUESTIONS

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d_i}}$$

Does this converge?

Does it converge to what we want?

Are results reasonable?

PAGERANK: PROBLEMS

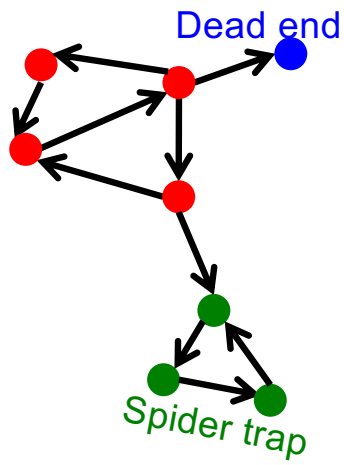
2 problems:

(1) Some pages are **dead ends** (have no out-links)

- Random walk has "nowhere" to go to
- Such pages cause importance to "leak out"

(2) Spider traps:
(all out-links are within the group)

- Random walked gets "stuck" in a trap
- And eventually spider traps absorb all importance

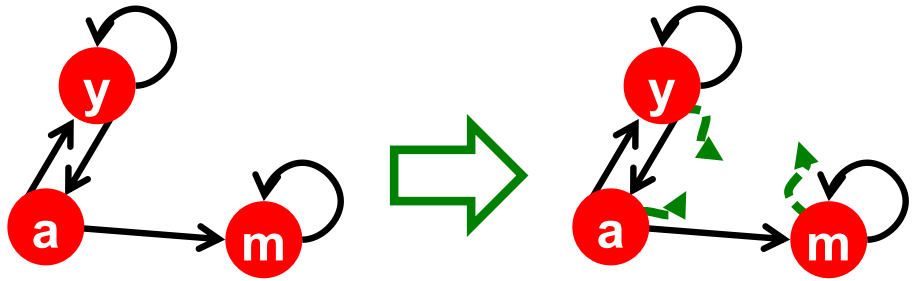


SOLUTION: TELEPORTS!

The Google solution for spider traps: At each time step, the random surfer has two options

- With prob. β, follow a link at random
- With prob. **1-**β, jump to some random page
- Common values for β are in the range 0.8 to 0.9

Surfer will teleport out of spider trap within a few time steps



SOME PROBLEMS WITH PAGE RANK

Measures generic popularity of a page

- Biased against topic-specific authorities
- **Solution:** Topic-Specific PageRank (**next**)

Uses a single measure of importance

- Other models of importance
- **Solution:** Hubs-and-Authorities

Susceptible to Link spam

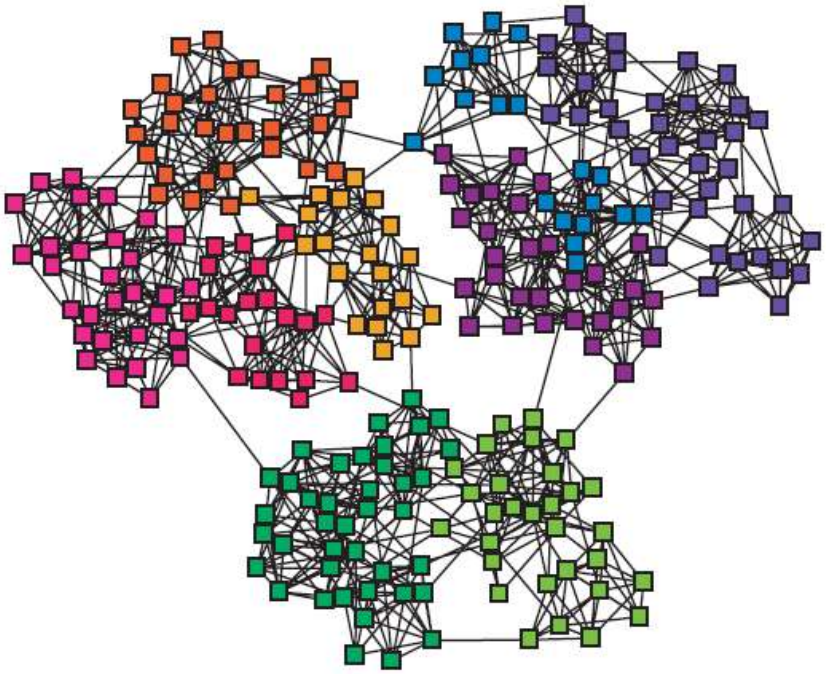
- Artificial link topographies created in order to boost page rank
- **Solution:** TrustRank

Challenge: implement a map reduce page rank algorithm

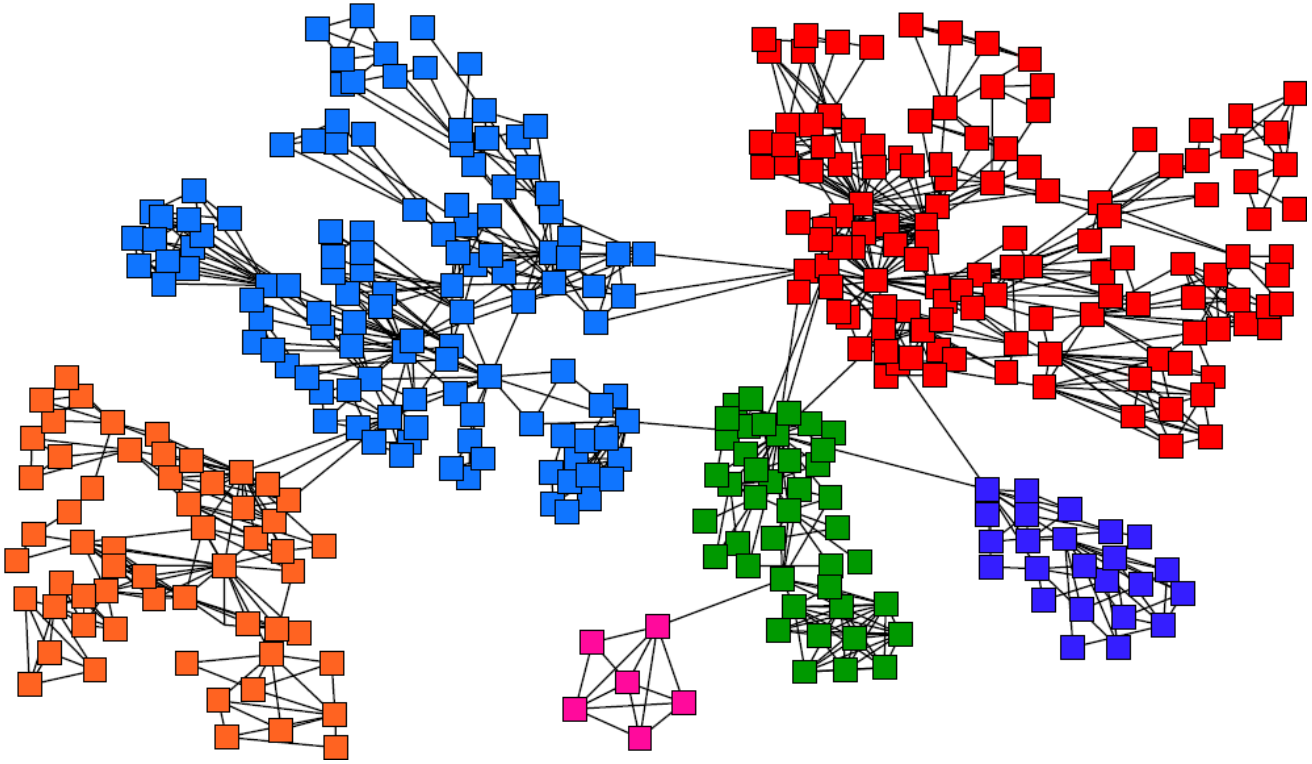
Community detection

NETWORKS & COMMUNITIES

We often think of networks being organized into modules, cluster, communities:



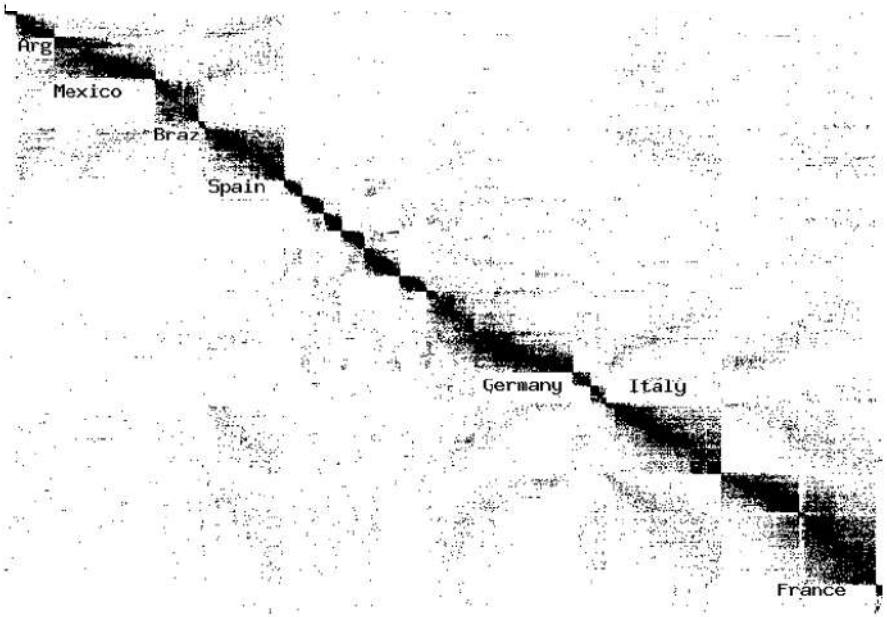
GOAL: FIND DENSELY LINKED CLUSTERS



MOVIES AND ACTORS

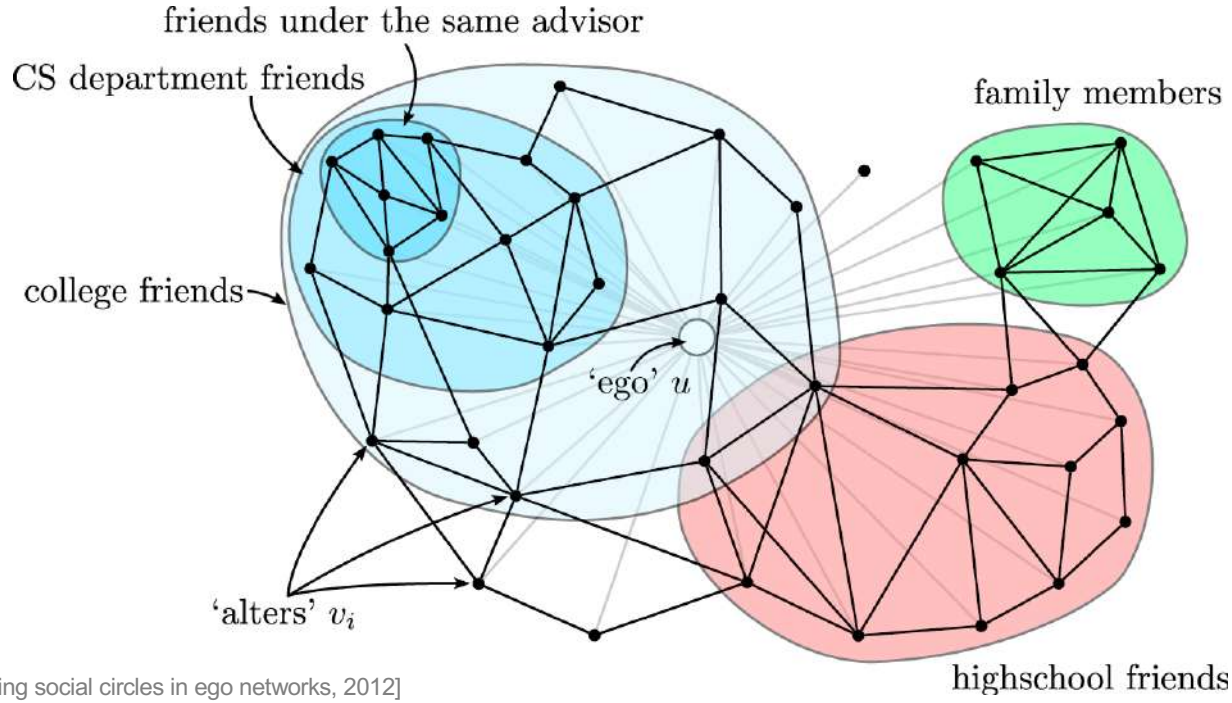
Clusters in Movies-to-Actors graph:

[Andersen, Lang: Communities from seed sets, 2006]



TWITTER & FACEBOOK

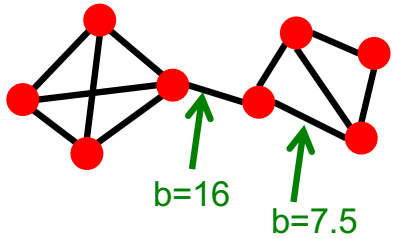
Discovering social circles, circles of trust:



[McAuley, Leskovec: Discovering social circles in ego networks, 2012]

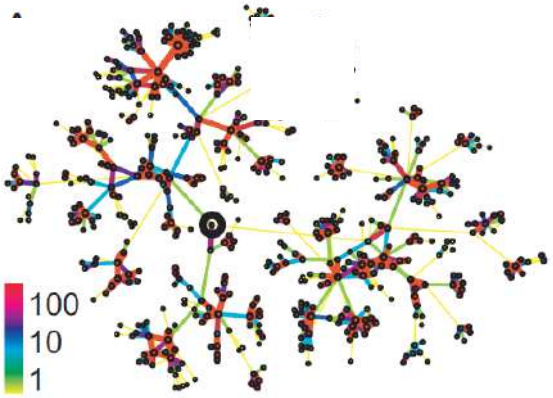
How to find communities?

METHOD 1: STRENGTH OF WEAK TIES

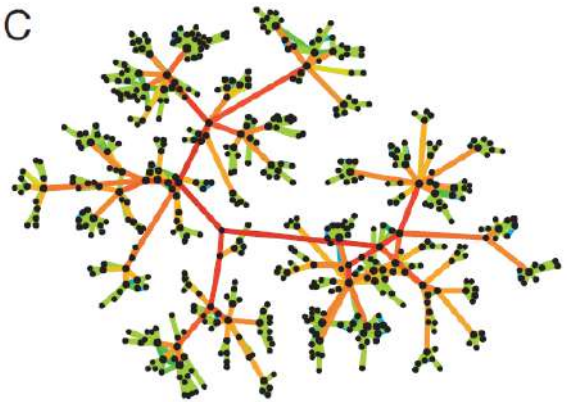


Edge betweenness: **Number of shortest paths passing over the edge**

Intuition:



Edge strengths (call volume) in a real network



Edge betweenness in a real network

METHOD 1: GIRVAN-NEWMAN

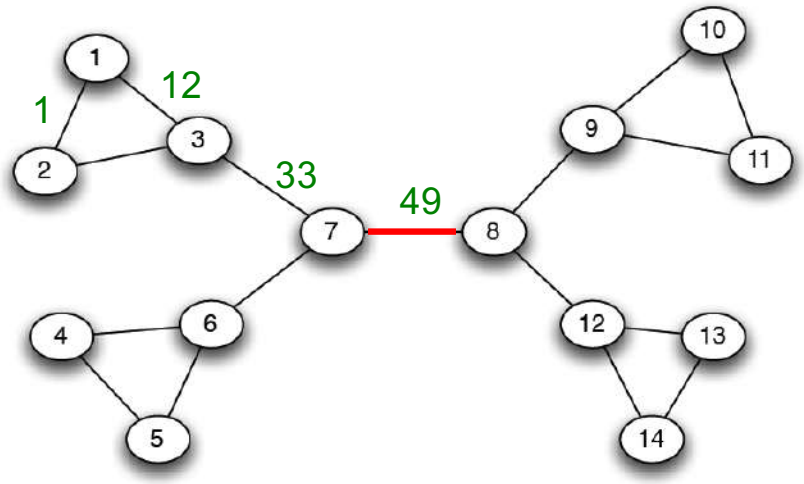
Divisive hierarchical clustering based on the notion of edge **betweenness:**

Number of shortest paths passing through the edge

Girvan-Newman Algorithm:

- **Undirected unweighted networks**
- **Repeat until no edges are left:**
  - Calculate betweenness of edges
  - Remove edges with highest betweenness
- Connected components are communities
- Gives a hierarchical decomposition of the network

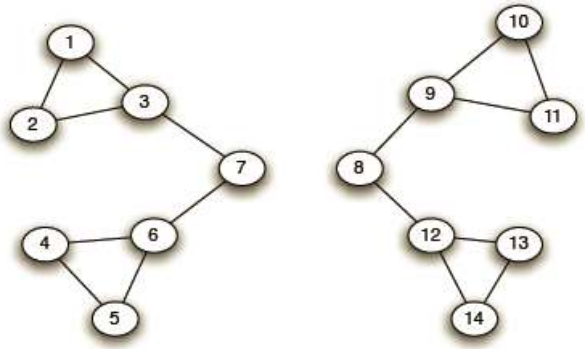
GIRVAN-NEWMAN: EXAMPLE



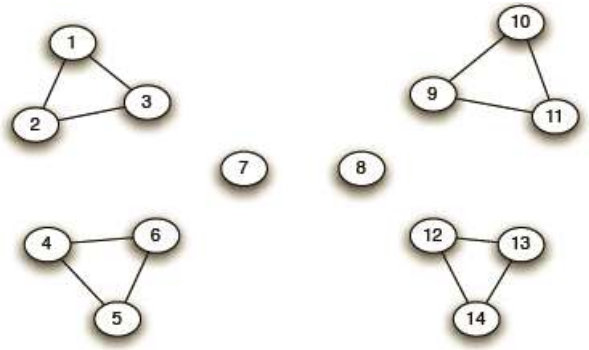
Need to re-compute betweenness at every step

GIRVAN-NEWMAN: EXAMPLE

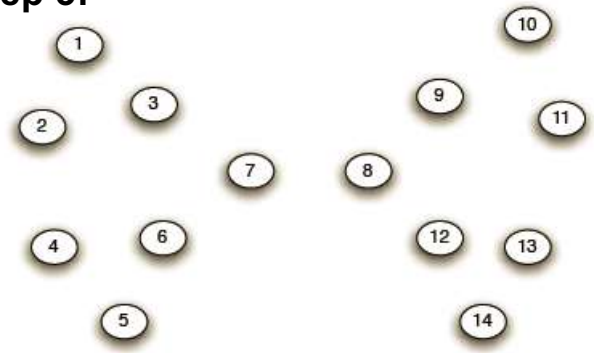
Step 1:



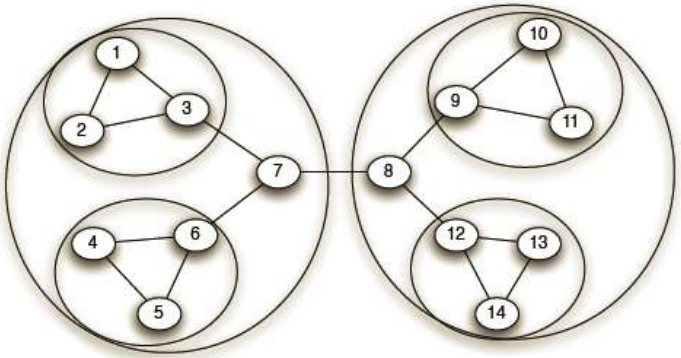
Step 2:



Step 3:

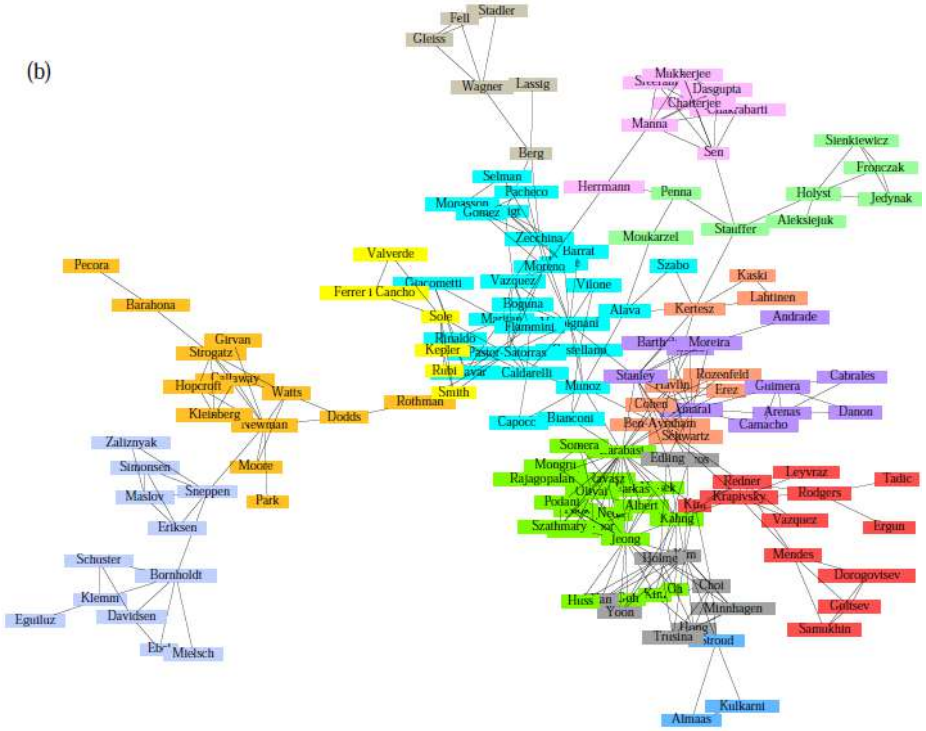


Hierarchical network decomposition:



GIRVAN-NEWMAN: RESULTS

Communities in physics collaborations



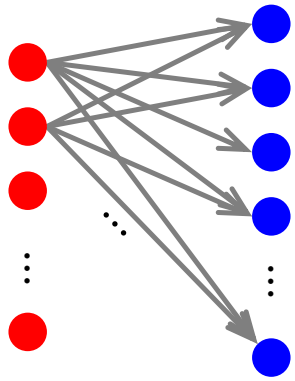
WE NEED TO RESOLVE 2 QUESTIONS

1. **How to compute betweenness?**
2. **How to select the number of clusters?**

TRAWLING

Searching for small communities in the Web graph

What is the signature of a community / discussion in a Web graph?



Dense 2-layer graph

Use this to define "topics":
What the same people on
the left talk about on the right
Remember HITS!

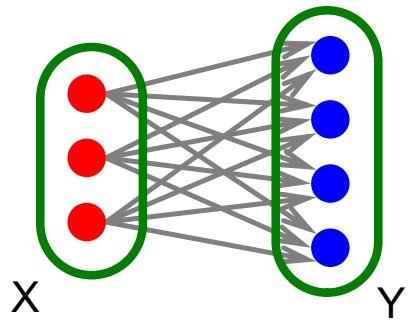
Intuition: Many people all talking about the same things

SEARCHING FOR SMALL COMMUNITIES

A more well-defined problem:

Enumerate complete bipartite subgraphs $K_{s,t}$

- Where $K_{s,t}$: s nodes on the "left" where each links to the same t other nodes on the "right"



$K_{3,4}$

|X| = s = 3
|Y| = t = 4

Fully connected

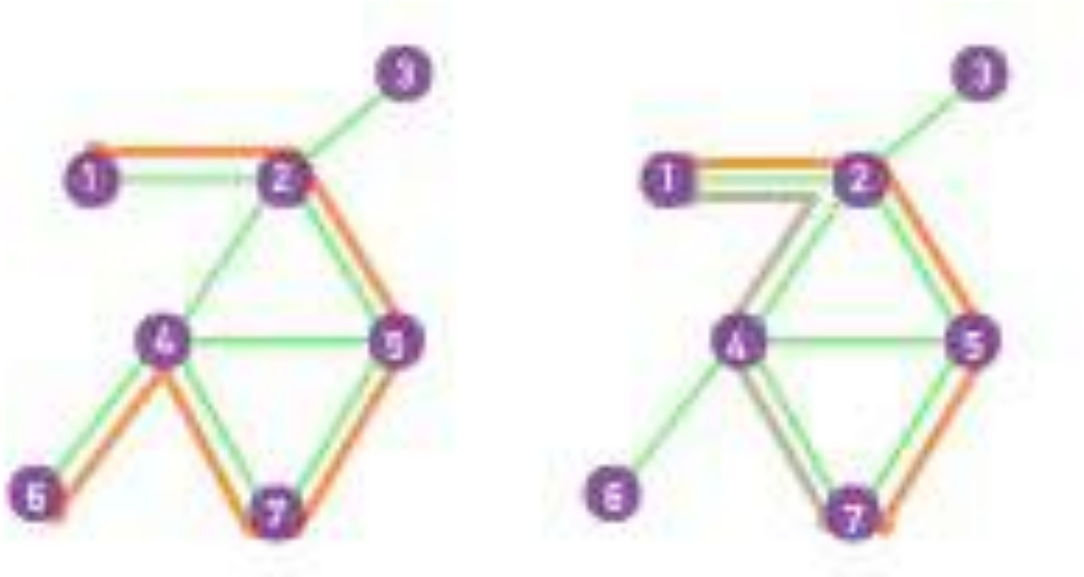
PATHOLOGY

PATHS

A *path is* a sequence of nodes in which each node is adjacent to the next one

$P_{i0,in}$ of length *n* between nodes i_0 and i_n is an ordered collection of *n+1* nodes and *n* links

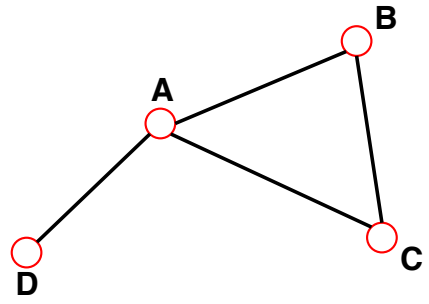
$$P_n = \{i_0, i_1, i_2, \ldots, i_n\} \qquad P_n = \{(i_0, i_1), (i_1, i_2), (i_2, i_3), \ldots, (i_{n-1}, i_n)\}$$



- In a directed network, the path can follow only the direction of an arrow.

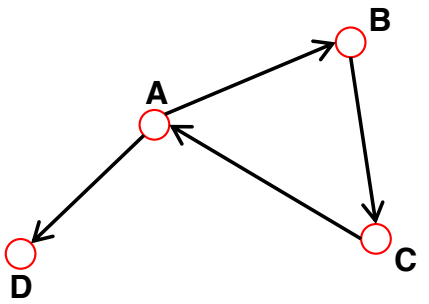
DISTANCE IN A GRAPH

SHORTEST PATH, GEODESIC PATH



The *distance (shortest path, geodesic path)* between two nodes is defined as the number of edges along the shortest path connecting them

*If the two nodes are disconnected, the distance is infinity



In directed graphs each path needs to follow the direction of the arrows

Thus in a digraph the distance from node A to B (on an AB path) is generally different from the distance from node B to A (on a BCA path)

N_{ij},number of paths between any two nodes *i* and *j*:

***Length n=1*:** If there is a link between *i* and *j*, then A_{ij}=1 and A_{ij}=0 otherwise

Length n=2: If there is a path of length two between *i* and *j*, then $A_{ik}A_{kj}$=1, and $A_{ik}A_{kj}$=0 otherwise.
The number of paths of length 2:

$$N_{ij}^{(2)} = \sum_{k=1}^{N} A_{ik}A_{kj} = [A^2]_{ij}$$

Length n: In general, if there is a path of length *n* between *i* and *j*, then $A_{ik}\ldots A_{lj}$=1 and $A_{ik}\ldots A_{lj}$=0 otherwise.
The number of paths of length *n* between *i* and *j* is*

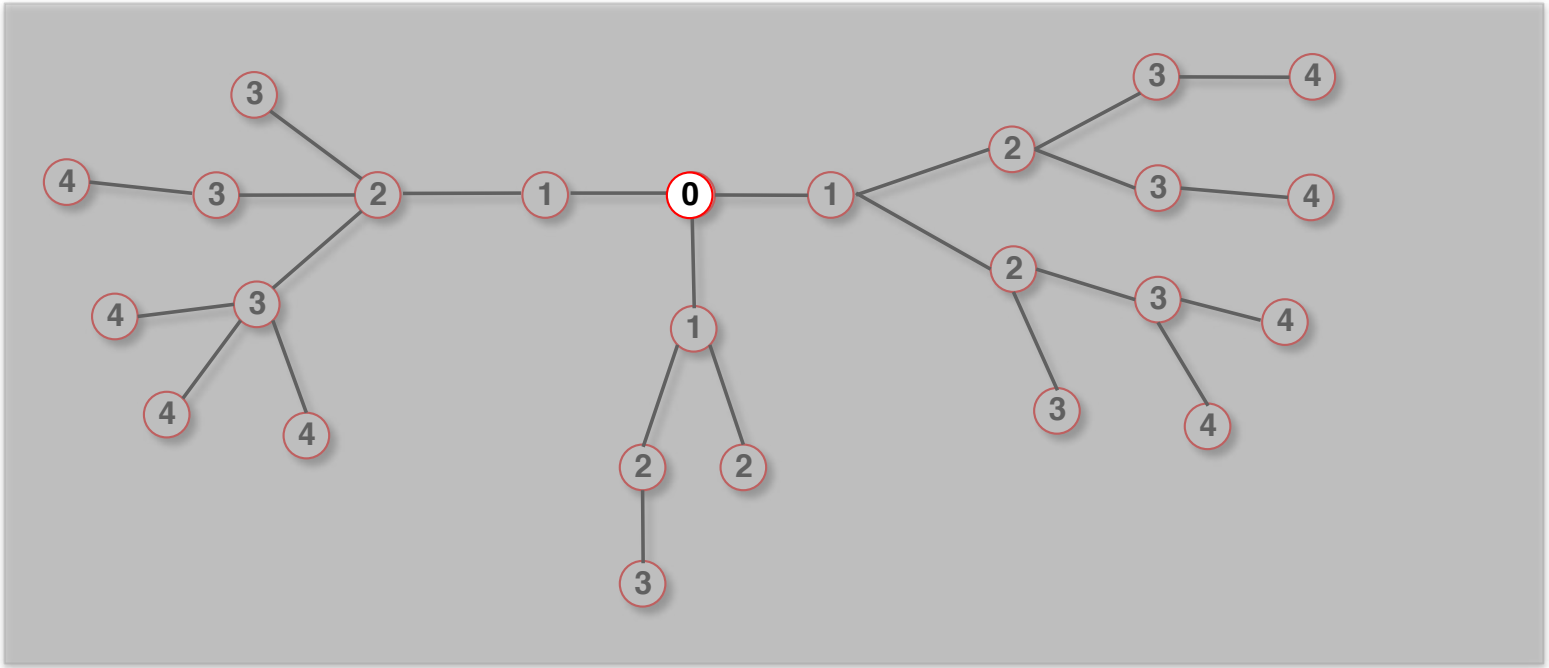
$$N_{ij}^{(n)} = [A^n]_{ij}$$

**holds for both directed and undirected networks.*

FINDING DISTANCES: BREATH FIRST SEARCH

Distance between node 0 and node 4:

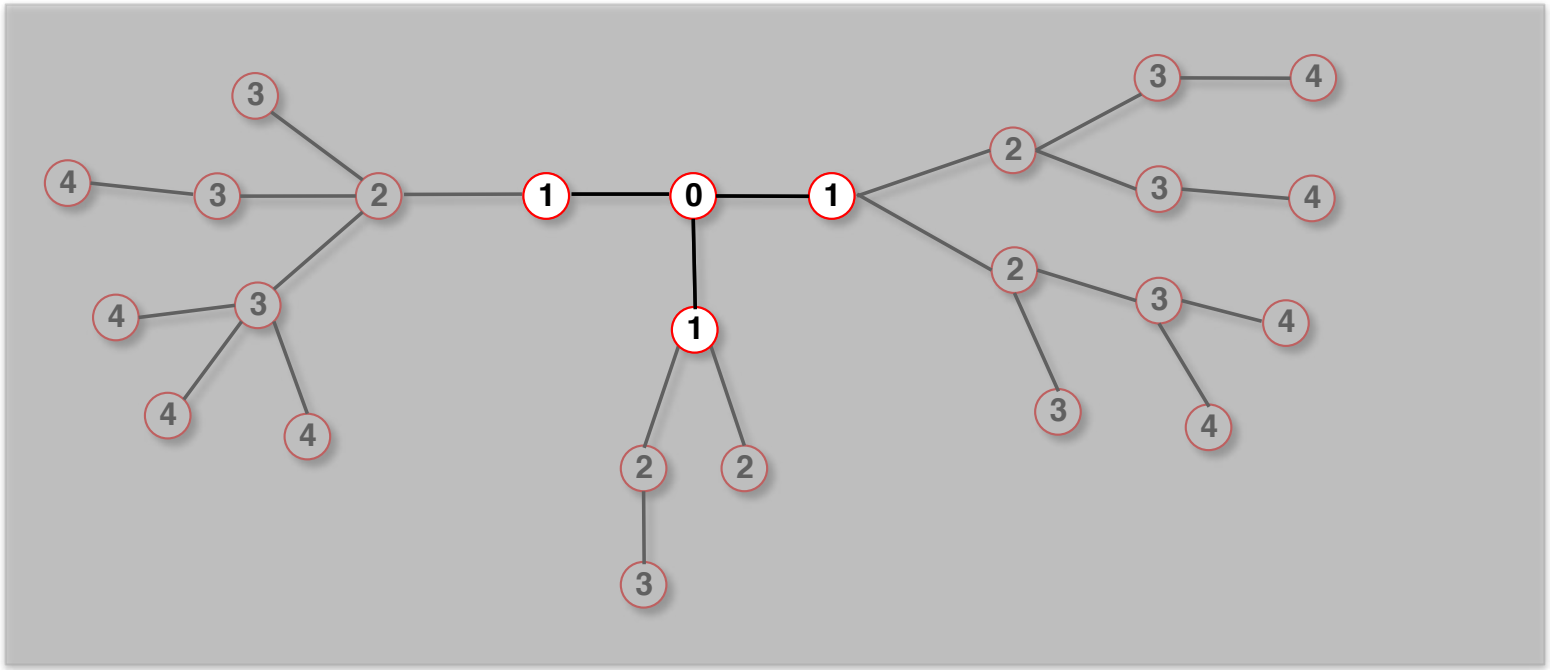
1.Start at 0.



FINDING DISTANCES: BREATH FIRST SEARCH

Distance between node 0 and node 4:

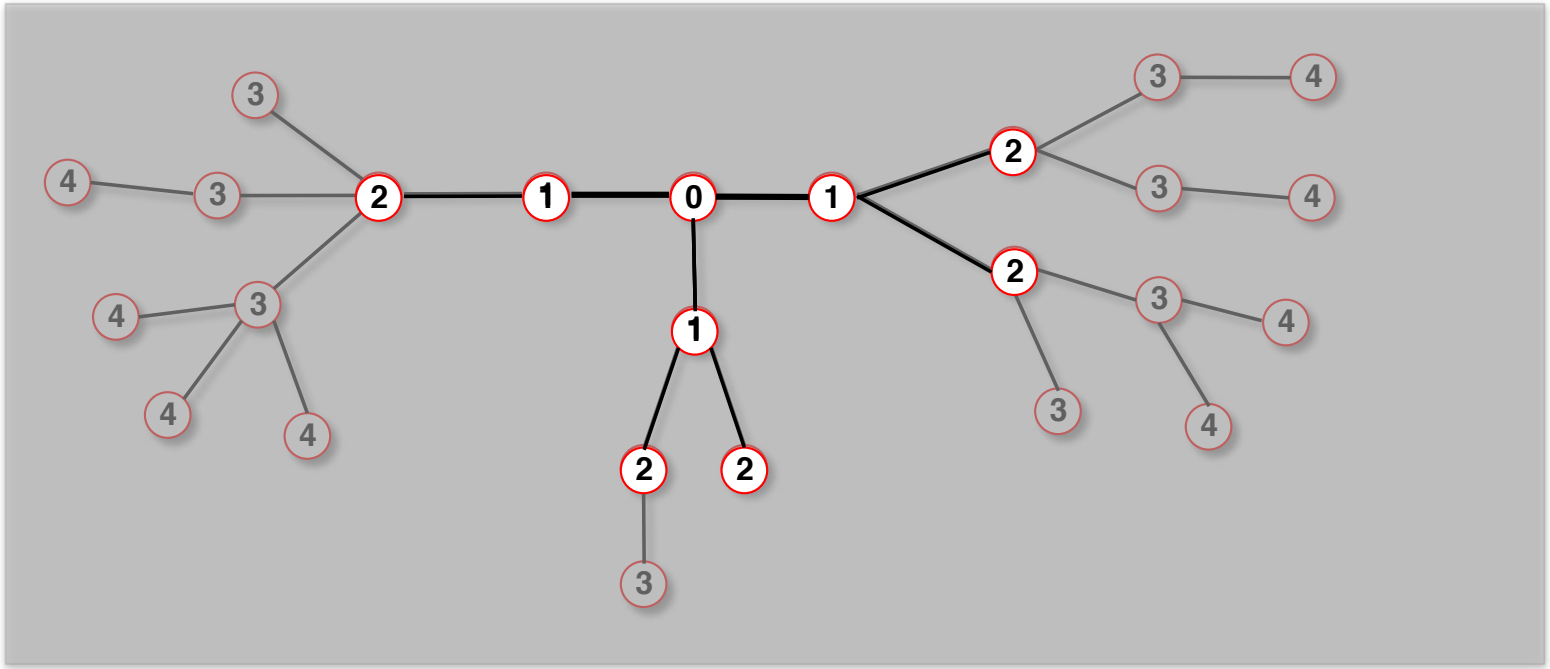
1.Start at 0.
2.Find the nodes adjacent to 1. Mark them as at distance 1. Put them in a queue.



FINDING DISTANCES: BREATH FIRST SEARCH

Distance between node 0 and node 4:

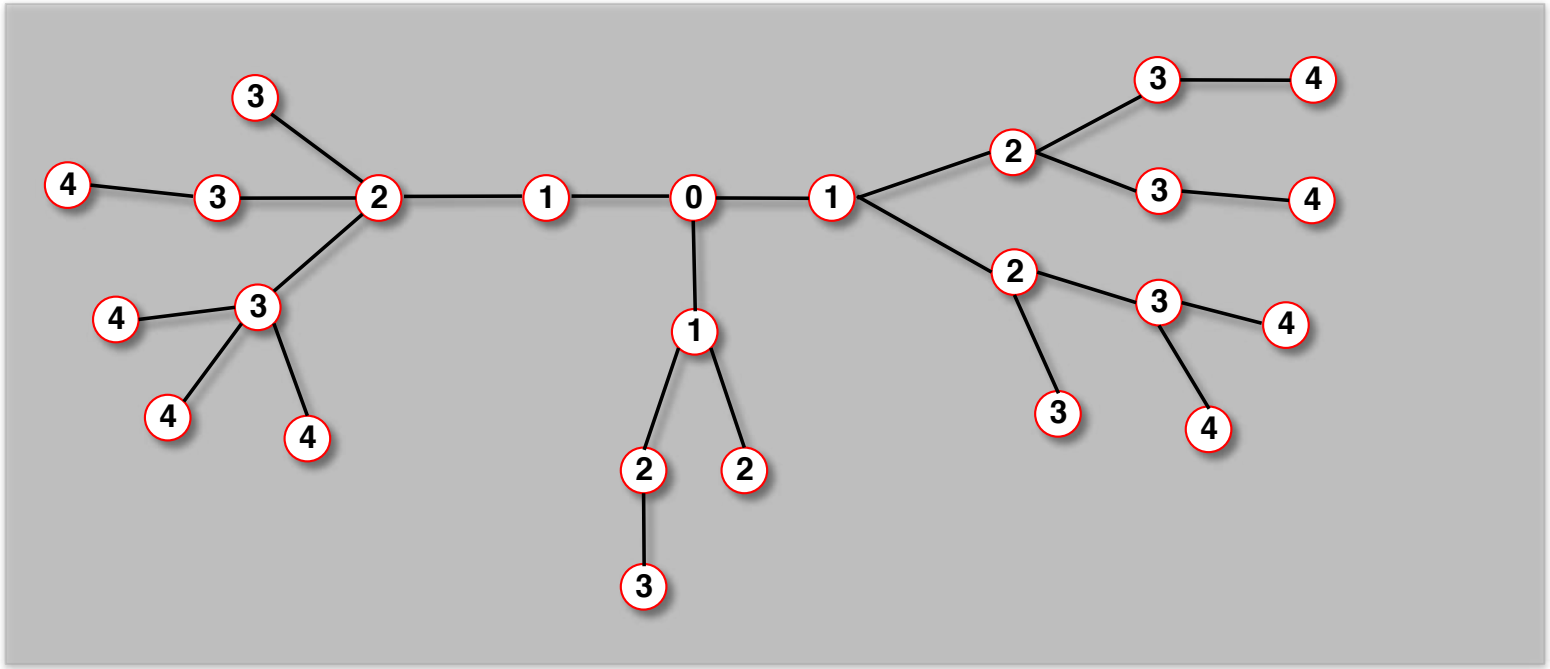
1.Start at 0.
2.Find the nodes adjacent to 0. Mark them as at distance 1. Put them in a queue.
3.Take the first node out of the queue. Find the unmarked nodes adjacent to it in the graph. Mark them with the label of 2. Put them in the queue.



FINDING DISTANCES: BREATH FIRST SEARCH

Distance between node 0 and node 4:

1.Repeat until you find node 4 or there are no more nodes in the queue.
2.The distance between 0 and 4 is the label of 4 or, if 4 does not have a label, infinity.



NETWORK DIAMETER AND AVERAGE DISTANCE

Diameter: $\boldsymbol{d_{max}}$ the maximum distance between any pair of nodes in the graph.

Average path length/distance, <d>, for a connected graph:

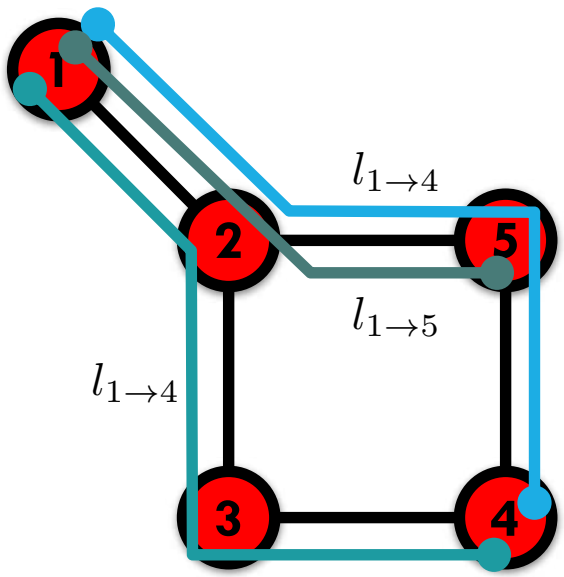
$$\langle d \rangle \equiv \frac{1}{2L_{\max}} \sum_{i,j \neq i} d_{ij}$$

where d_{ij} is the distance from node *i* to node j

In an *undirected graph* $d_{ij} = d_{ji}$, *so* we only need to count them once:

$$\langle d \rangle \equiv \frac{1}{L_{\max}} \sum_{i,j > i} d_{ij}$$

PATHOLOGY: SHORTEST PATH



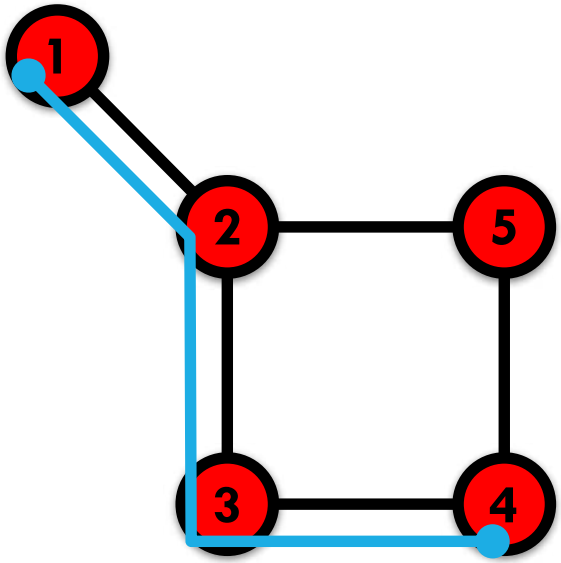
$$l_{1\to4} = 3$$

$$l_{1\to5} = 2$$

The path with the shortest length between two nodes (distance).

PATHOLOGY: DIAMETER & AVG PATH LENGTH

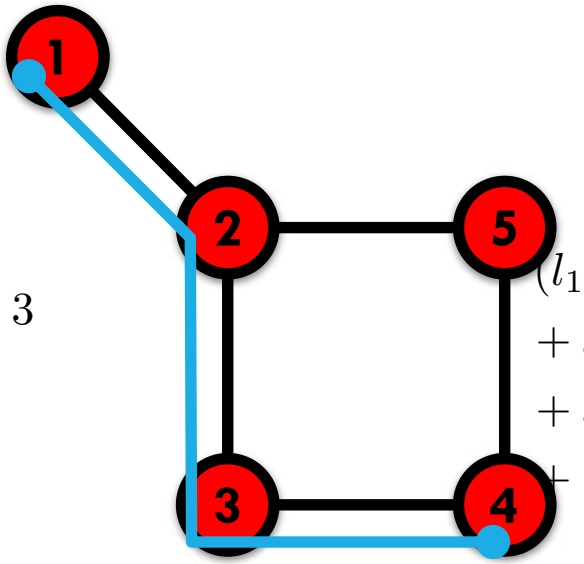
Diameter



$$l_{1\to 4} = 3$$

The longest shortest path in a graph

Average Path Length

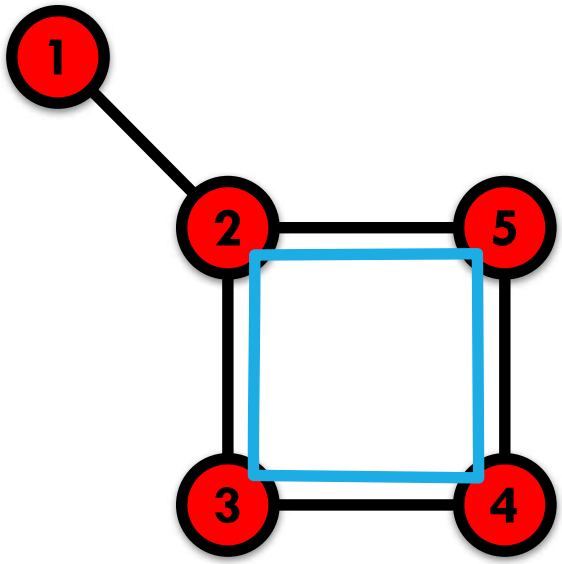


$$(l_{1\to 2} + l_{1\to 3} + l_{1\to 4} + l_{1\to 5} + l_{2\to 3} + l_{2\to 4} + l_{2\to 5} + l_{3\to 4} + l_{3\to 5} + l_{4\to 5})/10 = 1.6$$

The average of the shortest paths for all pairs of nodes.

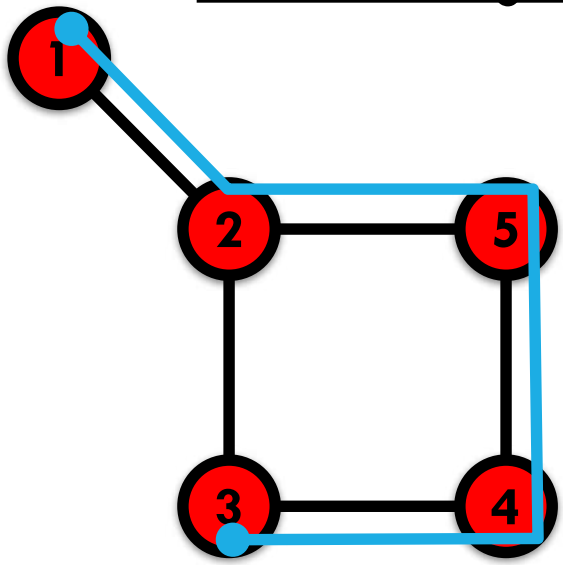
PATHOLOGY: CYCLE & SELF-AVOIDING PATH

Cycle



A path with the same start and end node.

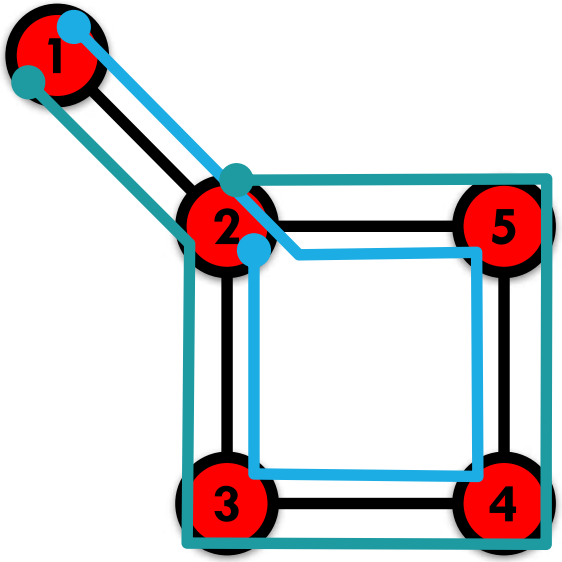
Self-avoiding Path



A path that does not intersect itself.

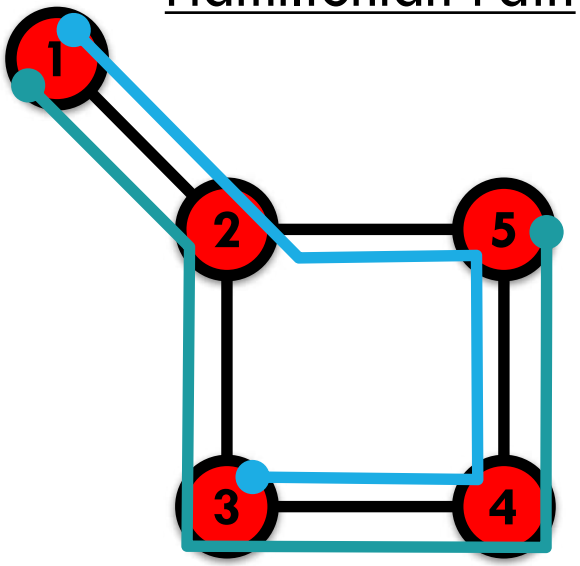
PATHOLOGY: EUCLIDEAN & HAMILTONIAN PATHS

Eulerian Path



A path that traverses each link exactly once.

Hamiltonian Path

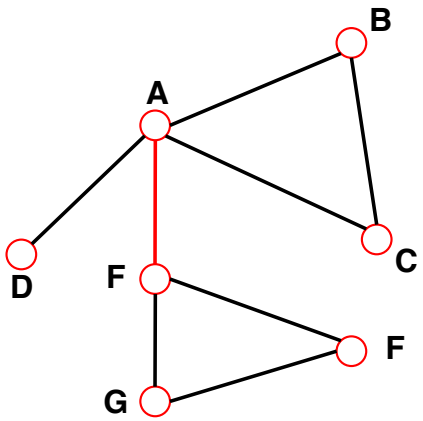


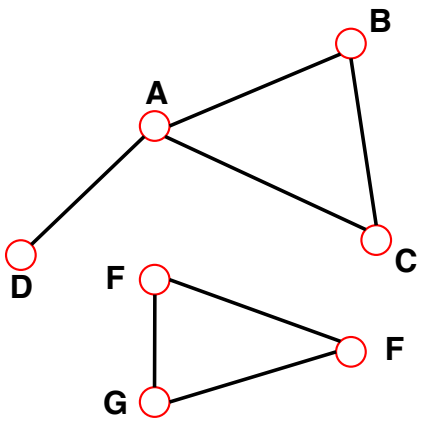
A path that visits each node exactly once.

CONNECTEDNESS

CONNECTIVITY OF UNDIRECTED GRAPHS

Connected (undirected) graph: any two vertices can be joined by a path.
A disconnected graph is made up by two or more connected components.





Largest Component:
Giant Component

The rest: **Isolates**

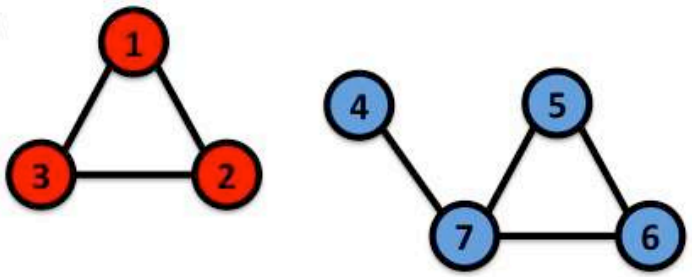
Bridge: if we erase it, the graph becomes disconnected.

ADJACENCY MATRIX

CONNECTIVITY OF UNDIRECTED GRAPHS

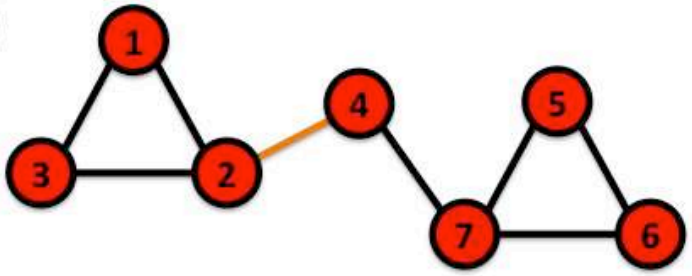
The adjacency matrix of a network with several components can be written in a block-diagonal form, so that nonzero elements are confined to squares, with all other elements being zero:

(a)



$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

(b)

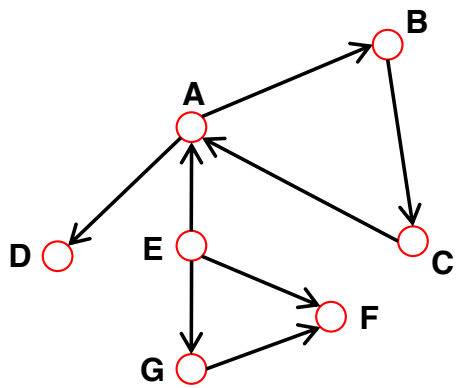


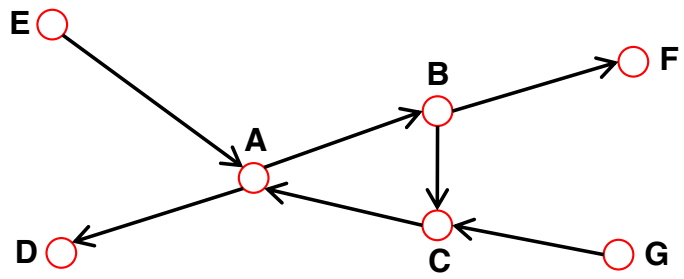
$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

CONNECTIVITY OF DIRECTED GRAPHS

Strongly connected directed graph: has a path from each node to every other node and vice versa (e.g. AB path and BA path).
Weakly connected directed graph: it is connected if we disregard the edge directions.

Strongly connected components can be identified, but not every node is part of a nontrivial strongly connected component.





In-component: nodes that can reach the scc,
Out-component: nodes that can be reached from the scc.

FINDING THE CONNECTED COMPONENTS OF A NETWORK

1. Start from a randomly chosen node i and perform a BFS (BOX 2.5). Label all nodes reached this way with $n = 1$.

2. If the total number of labeled nodes equals N, then the network is connected. If the number of labeled nodes is smaller than N, the network consists of several components. To identify them, proceed to step 3.

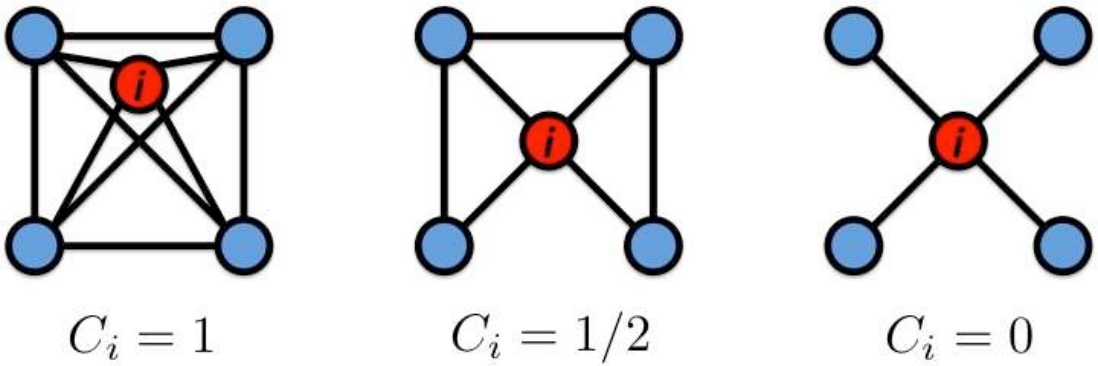
3. Increase the label $n \to n + 1$. Choose an unmarked node j, label it with n. Use BFS to find all nodes reachable from j, label them all with n. Return to step 2.

Clustering coefficient

CLUSTERING COEFFICIENT

- what fraction of your neighbors are connected?
  - Node i with degree k_i
  - C_i in [0,1]

$$C_i = \frac{2e_i}{k_i(k_i - 1)}$$

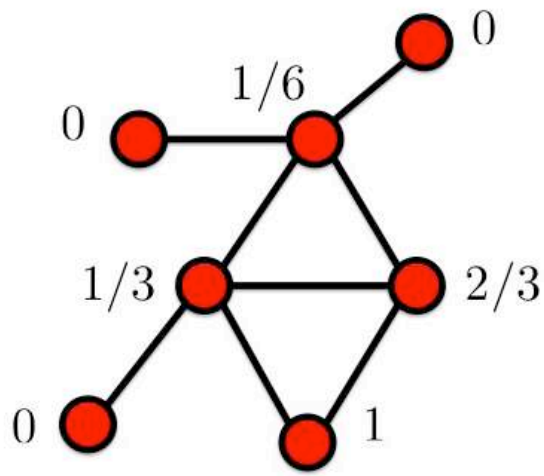


Watts & Strogatz, Nature 1998.

CLUSTERING COEFFICIENT

- what fraction of your neighbors are connected?
  - Node i with degree k_i
  - C_i in [0,1]

$$C_i = \frac{2e_i}{k_i(k_i - 1)}$$



$$\langle C \rangle = \frac{13}{42} \approx 0.310$$

$$C = \frac{3}{8} = 0.375$$

Watts & Strogatz, Nature 1998.

Useful concepts about graphs

NETWORK PROPERTIES

Density: defined as a ratio of the number of edges E to the number of possible edges

Size: the number of nodes N or, less commonly, the number of edges E which can range from N-1 (a tree) to E_{max} (a complete graph).

Average degree: the number of edges connected to it.

- Density of a network average degree,
- Random graph model, <k> = p(N-1) where p is the probability of two nodes being connected

NETWORK PROPERTIES

Average path length: calculated by finding the shortest path between all pairs of nodes

- adding them up, and then dividing by the total number of pairs
- This shows, on average, the number of steps it takes to get from one member of the network to another

Diameter of a network: the longest of all the calculated shortest paths in a network

NETWORK PROPERTIES

Connectedness:

- **Clique/Complete Graph:** a completely connected network, where all nodes are connected to every other node. These networks are symmetric in that all nodes have in-links and out-links from all others
- **Giant Component:** A single connected component which contains most of the nodes in the network
- **Weakly Connected Component:** A collection of nodes in which there exists a path from any node to any other, ignoring directionality of the edges
- **Strongly Connected Component:** A collection of nodes in which there exists a directed path from any node to any other

NETWORK PROPERTIES

Node centrality: produce rankings which seek to identify the most important nodes in a network model

Different centrality indices encode different contexts for the word "importance"

- The **betweenness** centrality, considers a node highly important if it forms bridges between many other nodes
- The **eigenvalue** centrality considers a node highly important if many other highly important nodes link to it

NETWORK PROPERTIES

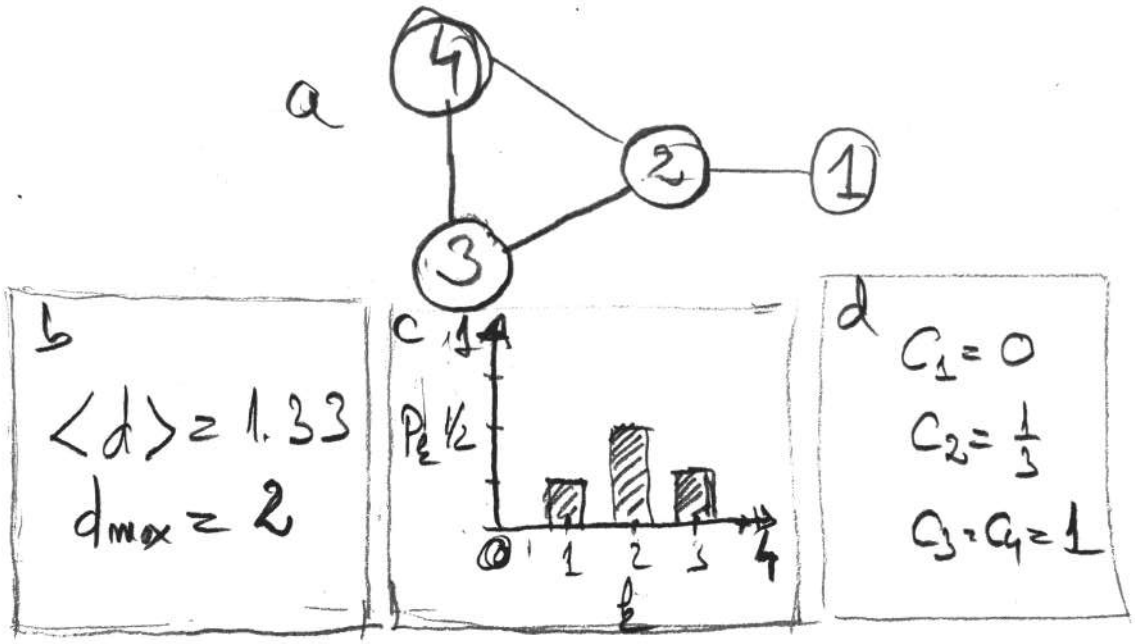
Node influence: measures that rank or quantify the influence of every node (vertex) within a graph.

Related to centrality indices.

Applications:

- measuring the influence of each person in a social network
- understanding the role of infrastructure nodes in transportation networks, the Internet, or urban networks
- participation of a given node in disease dynamics

THREE CENTRAL QUANTITIES IN NETWORK SCIENCE



A. Degree distribution: p_k

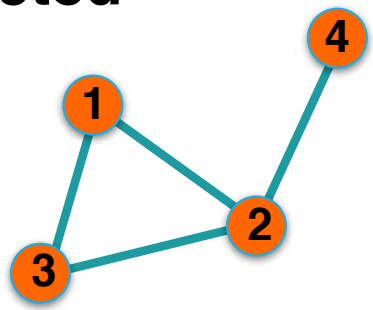
B. Path length: $\langle d \rangle$

C. Clustering coefficient:

$$C_i = \frac{2e_i}{k_i(k_i - 1)}$$

GRAPHOLOGY 1 SHORTEST PATH, GEODESIC PATH

Undirected



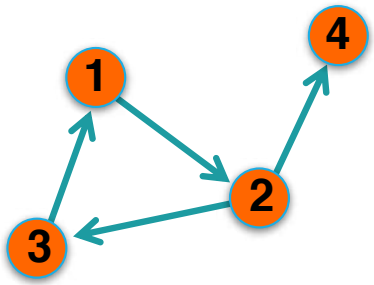
$$A_{ij} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

$$L = \frac{1}{2}\sum_{i,j=1}^{N} A_{ij} \qquad < k >= \frac{2L}{N}$$

Actor network, protein-protein interactions

Directed



$$A_{ij} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

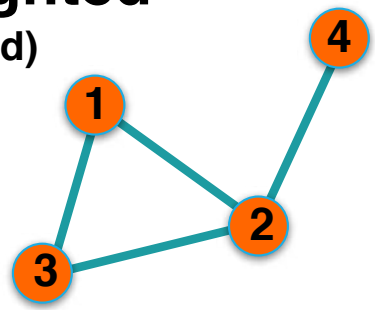
$$A_{ii} = 0 \qquad A_{ij} \neq A_{ji}$$

$$L = \sum_{i,j=1}^{N} A_{ij} \qquad < k >= \frac{L}{N}$$

WWW, citation networks

GRAPHOLOGY 2 SHORTEST PATH, GEODESIC PATH

Unweighted
(undirected)



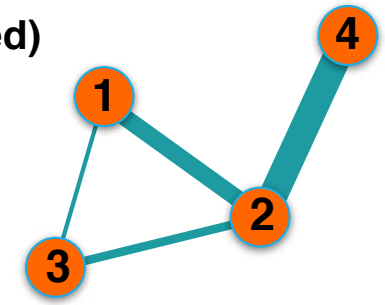
$$A_{ij} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

$$L = \frac{1}{2}\sum_{i,j=1}^{N} A_{ij} \qquad < k >= \frac{2L}{N}$$

protein-protein interactions, www

Weighted
(undirected)



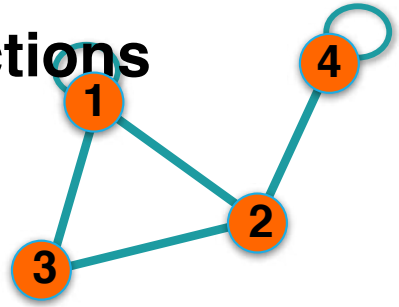
$$A_{ij} = \begin{pmatrix} 0 & 2 & 0.5 & 0 \\ 2 & 0 & 1 & 4 \\ 0.5 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 \end{pmatrix}$$

$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

$$L = \frac{1}{2}\sum_{i,j=1}^{N} nonzero(A_{ij}) \qquad < k >= \frac{2L}{N}$$

Call Graph, metabolic networks

Self-interactions



$$A_{ij} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

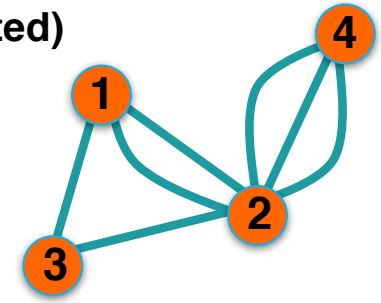
$$A_{ii} \neq 0 \qquad A_{ij} = A_{ji}$$

$$L = \frac{1}{2} \sum_{i,j=1, i \neq j}^{N} A_{ij} + \sum_{i=1}^{N} A_{ii} \qquad ?$$

Protein interaction network, www

Multigraph

(undirected)



$$A_{ij} = \begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 0 & 1 & 3 \\ 1 & 1 & 0 & 0 \\ 0 & 3 & 0 & 0 \end{pmatrix}$$

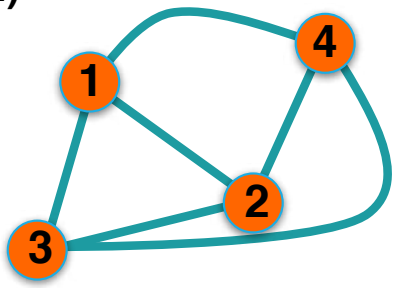
$$A_{ii} = 0 \qquad A_{ij} = A_{ji}$$

$$L = \frac{1}{2} \sum_{i,j=1}^{N} nonzero(A_{ij}) \qquad < k >= \frac{2L}{N}$$

Social networks, collaboration networks

Complete Graph

(undirected)

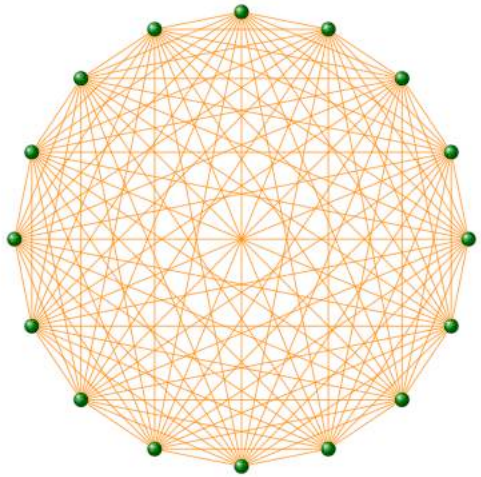


$$A_{ij} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

$$A_{ii} = 0 \qquad A_{i \neq j} = 1$$

$$L = L_{\max} = \frac{N(N-1)}{2} \qquad < k >= N - 1$$

Actor network, protein-protein interactions



Implementation 1

DESIGN STEPS

Define and handle the input and output of the implementation

- The input is given as a <key, value> pair

GENERAL PRINCIPLE

- In the map step
  - Read the cluster centers into memory from a sequencefile
  - Iterate over each cluster center for each input key/value pair.
  - Measure the distances and save the nearest center which has the lowest distance to the vector
  - Write the clustercenter with its vector to the filesystem.
- In the reduce step (we get associated vectors for each center)
  - Iterate over each value vector and calculate the average vector. (Sum each vector and devide each part by the number of vectors we received).
  - This is the new center, save it into a SequenceFile.
  - Check the convergence between the clustercenter that is stored in the key object and the new center.
    - If it they are not equal, increment an update counter
- Run this whole thing until nothing was updated anymore.

PRE-REQUISITE

Two files:

- F1: houses the clusters with their centroids
- F2: houses the vectors to be clustered

The initial set of centres is stored in the input directory of HDFS

- they form the 'key' field in the `<key,value>`

MAP & REDUCE ROUTINES

Mapper:

- Computes the distance between the given data set and cluster centre fed as a `<key,value>`
- Keeps track of the cluster to which the given vector is closest
- Assign the vector to the nearest cluster, once the computation of distances is complete

Reducer:

- Recalculates the centroid
- Restructures the cluster to prevent creations of clusters with extreme sizes i.e. cluster having too less data vectors or a cluster having too many data vectors
- Re-writes the new set of vectors and clusters to the disk

Ready for the next iteration

Algorithm 1 Mapper design for K-Means Clustering

```
0: procedure KMEANMAPDESIGN
0:     LOAD Cluster file
0:     fp = Mapclusterfile
0:     Create two list
0:     listnew = listold
0:     CALL read (Mapclusterfile)
0:     newfp = MapCluster()
0:     dv =0
0:     Assign correct centeroid
0:     read(dv)
0:     calculate centerorid
0:     dv = minCenter()
0:     CALL KmeansReduce()
0: end procedure=0
```

Algorithm 2 Reducer design for K-Means Clustering

```
0: procedure KMEANREDUCEDESIGN
0:     NEW ListofClusters
0:     COMBINE resultant clusters from MAP CLASS.
0:     if cluster size too high or too low then
   RESIZE the cluster
0:         C_Max = findMaxSize(ListofClusters)
0:         C_min = findMinSize(ListofClusters)
0:         if C_max > 1/20 totalSize then Resize(cluster)
0:             WRITE cluster FILE to output DIRECTORY.
0:
```

Algorithm 3 Implementing KMeans Function

```
0: procedure KMEANS FUNCTION
0:    if Initial Iteration then LOAD cluster file from DIREC-
   TORY
0:    elseREAD cluster file from previous iteration
0:       Create new JOB
0:       SET MAPPER to map class defined
0:       SET REDUCER to reduce class define
0:       paths for output directory
0:       SUBMIT JOB
0:
```

Implementation 2

http://codingwiththomas.blogspot.kr/2011/05/k-means-clustering-with-mapreduce.html