Welcome to IE 532

Chrysafis Vogiatzis

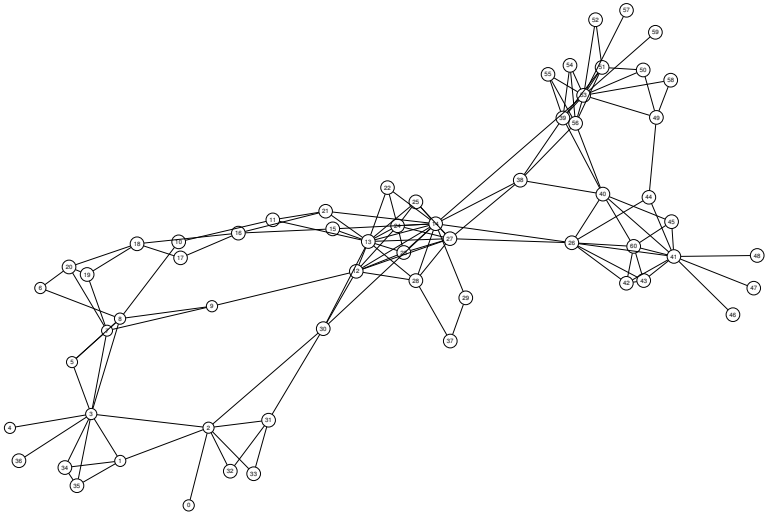
Department of Industrial and Enterprise Systems Engineering
University of Illinois at Urbana-Champaign

Lecture 1: 08/23/2022

ILLINOIS
ISE | Industrial & Enterprise Systems Engineering
GRAINGER COLLEGE OF ENGINEERING

Welcome to IE 532

In this class, we will study **networks**.



"A collection of points together with lines that connect some subset of the points."

– Wolfram Alpha

We define a mathematical construct called graph G with two sets:

1. a set of "points" (vertices V or nodes N);
2. a set of "lines" connecting "points" (edges E or arcs A).

Analysis of networks

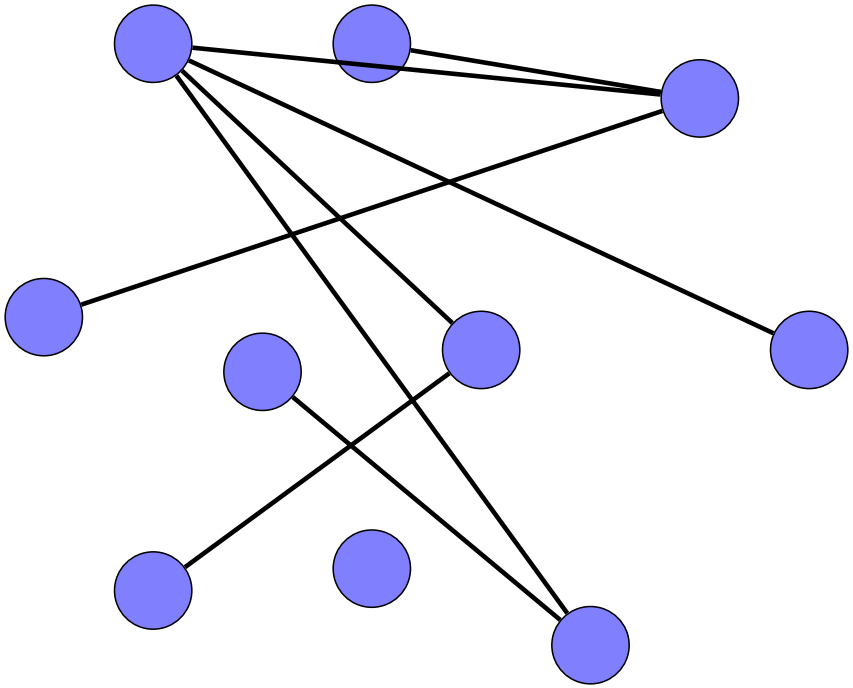
"Networks are present everywhere. All we need is an eye for them."

– Albert László Barabási

Networks are the "language" of complex systems:

- social networks;
- transportation networks;
- communications networks;
- power networks;
- biological networks; etc.

In the above, we are presented with **entities** and their **interactions**.



Who cares about networks?

Computer scientists

Physicists

Engineers, including

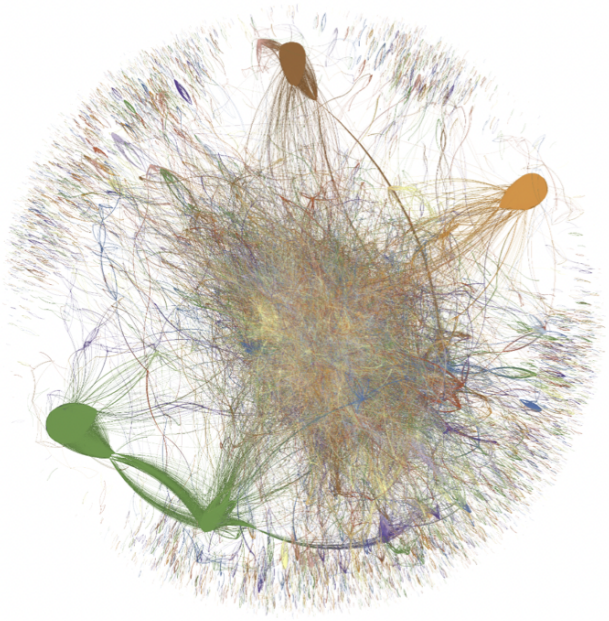
- Civil (transportation, infrastructure, energy-water nexus, etc.).
- Electrical (communication, power, microchip design)
- Of course, Industrial & Systems!

Political scientists, social scientists, sociologists

Bioinformaticians, biostatisticians, biologists

Mathematicians

Each of these communities have their own preferred *languages* and their own preferred *tools*.



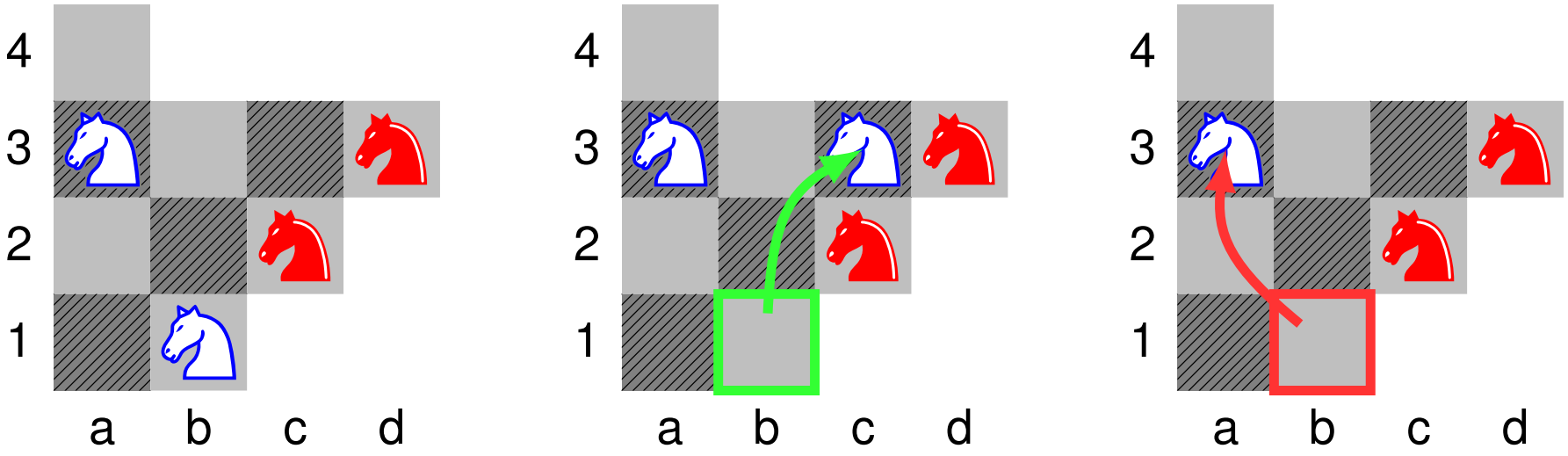
Co-authorship network for network science papers. Taken from Molontay, R., Nagy, M. (2021). **Twenty Years of Network Science: A Bibliographic and Co-authorship Network Analysis.** In: Çakırtaş, M., Ozdemir, M.K. (eds) Big Data and Social Media Analytics. Lecture Notes in Social Networks. Springer, Cham.

ILLINOIS

Why networks?

Networks provide us with a different, *more intuitive* way of viewing system dynamics and interactions.

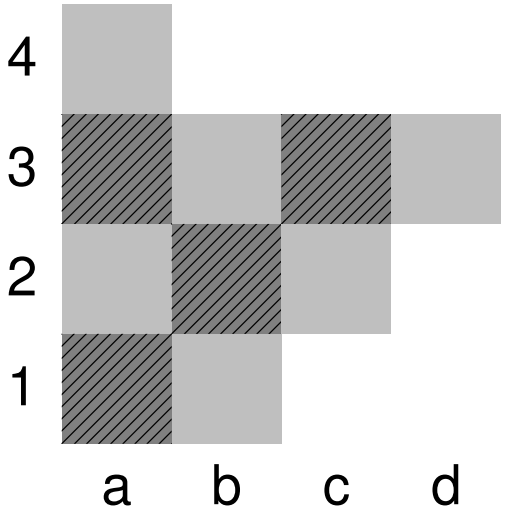
Let's see that with a game! Can you swap the positions of the knights in this weird chessboard? [1]

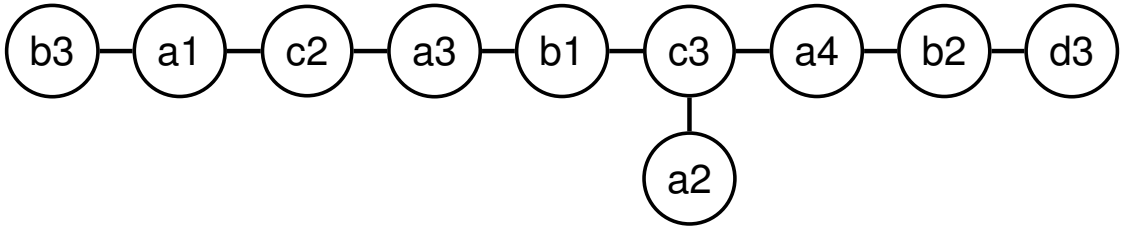


[1] Puzzle taken from the 11th Hour video game (1995).

Why networks? A small puzzle

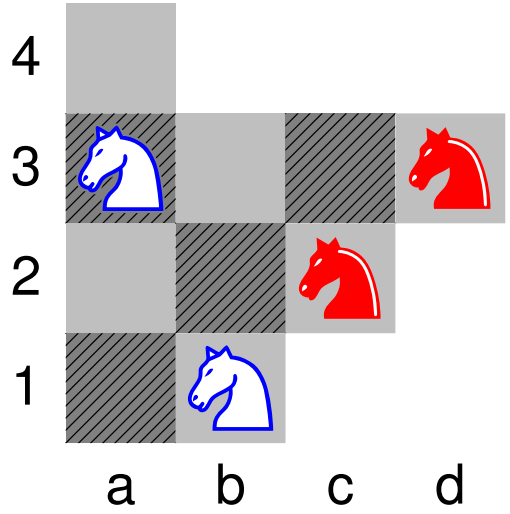
As humans, we like visual representations. Here though, the visual chessboard hinders our understanding! It doesn't matter at all.

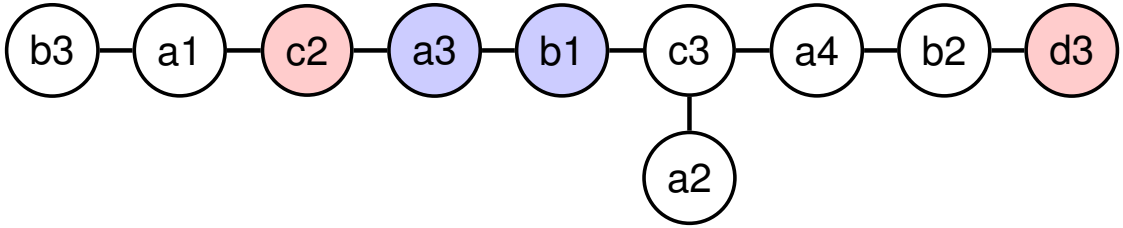


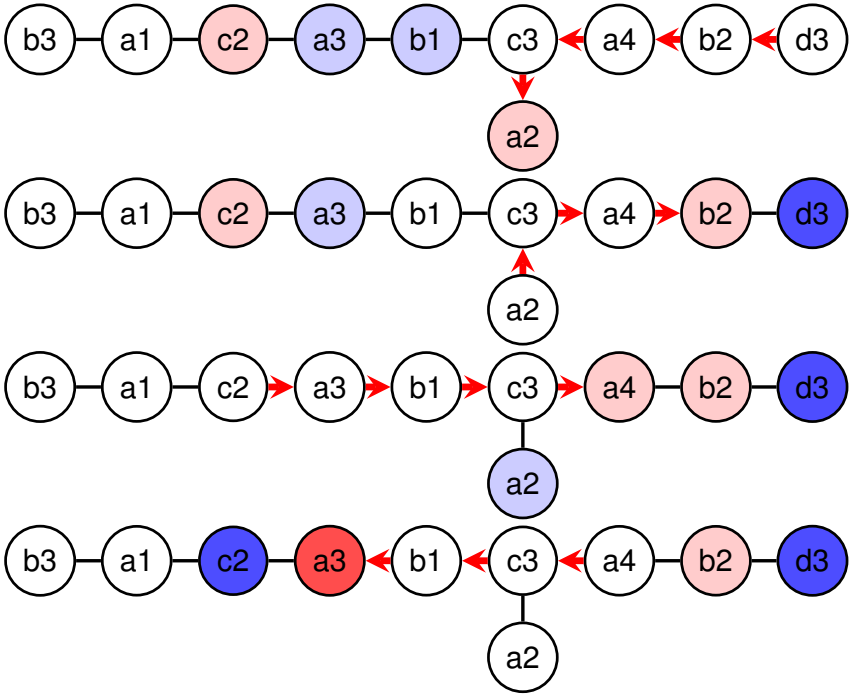


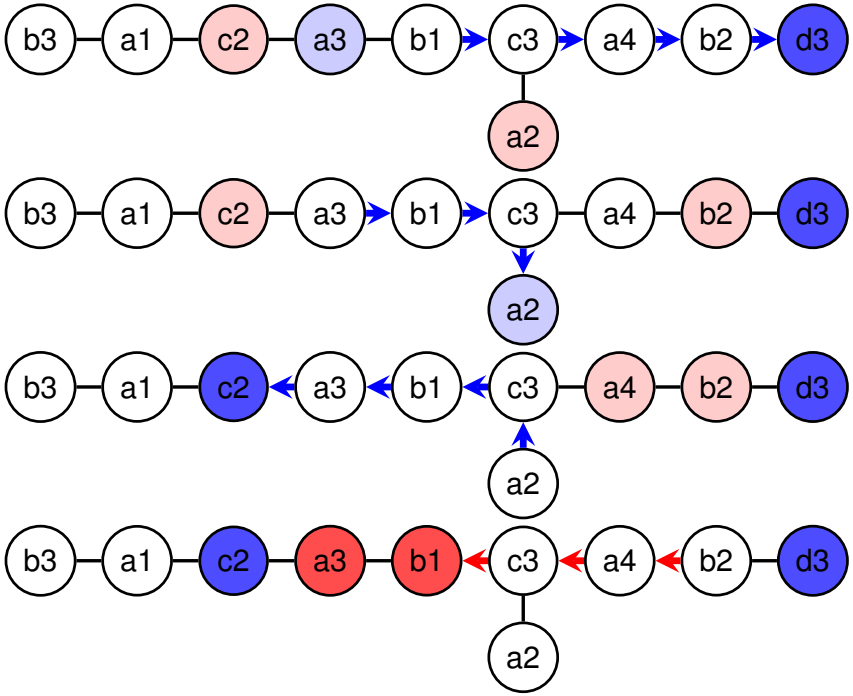
- Let's create a node for every position on the board (10 of them).
- Let's connect two nodes if a knight is allowed to move between them (a4 to c3, a4 to b2, etc.).

Why networks? A small puzzle









Examples of networks: from transportation

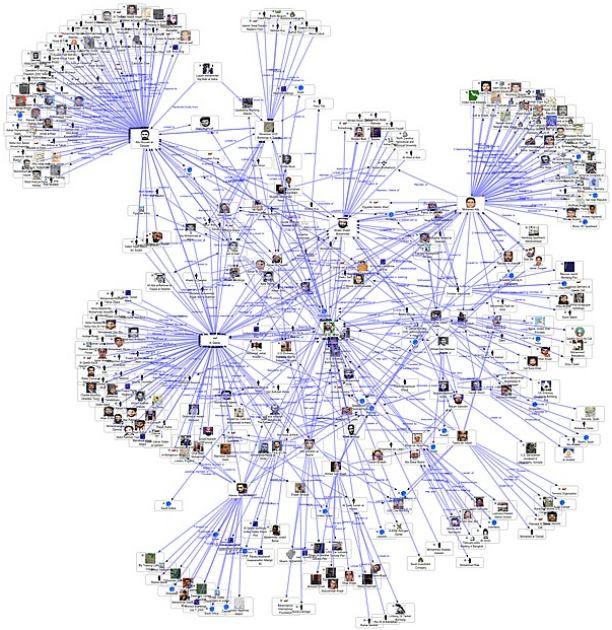
The transportation network of the state of WV, including only major highways. Obtained using `osmnx` (https://github.com/gboeing/osmnx).

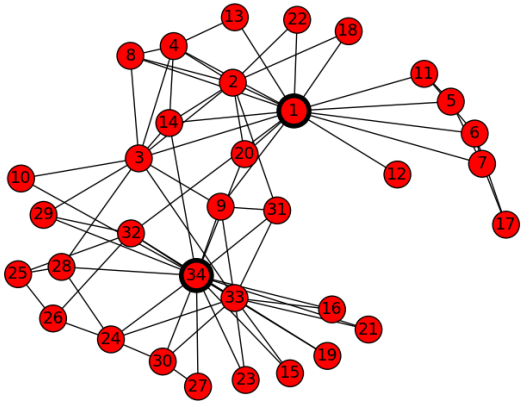
Examples of networks: airport networks

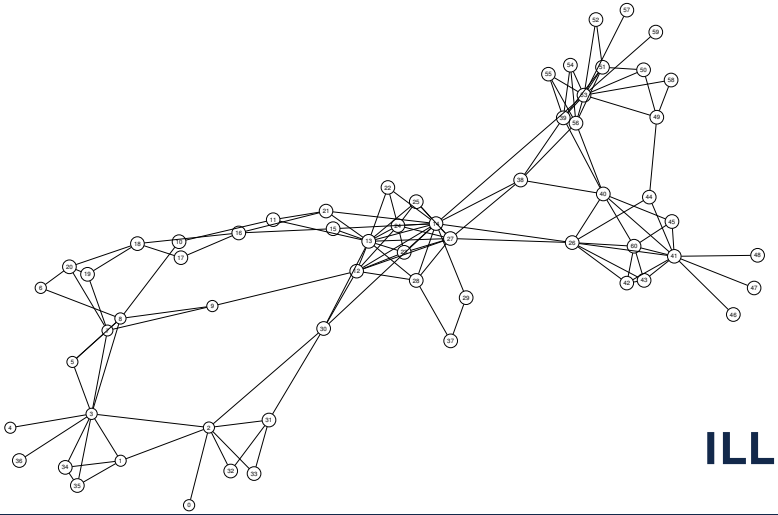
The airport network (in 2016, as shown in http://www.martingrandjean.ch/wp-content/uploads/2016/05/airports-map.png).

Examples of networks: social networks

Both online (facebook, twitter, instagram, email, phone calls) and offline (social interactions, karate club graph, dolphins, 9-11, book graphs).

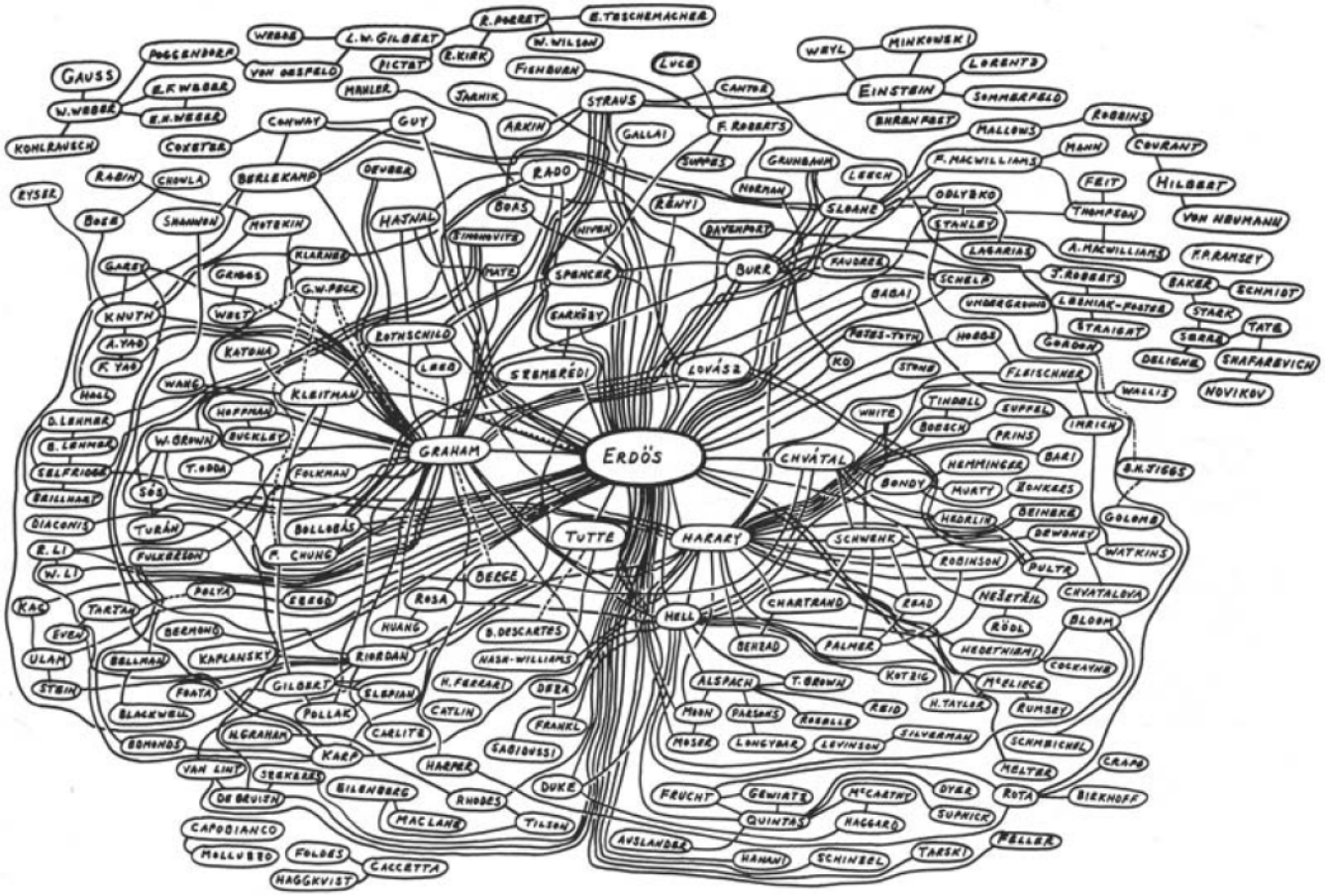






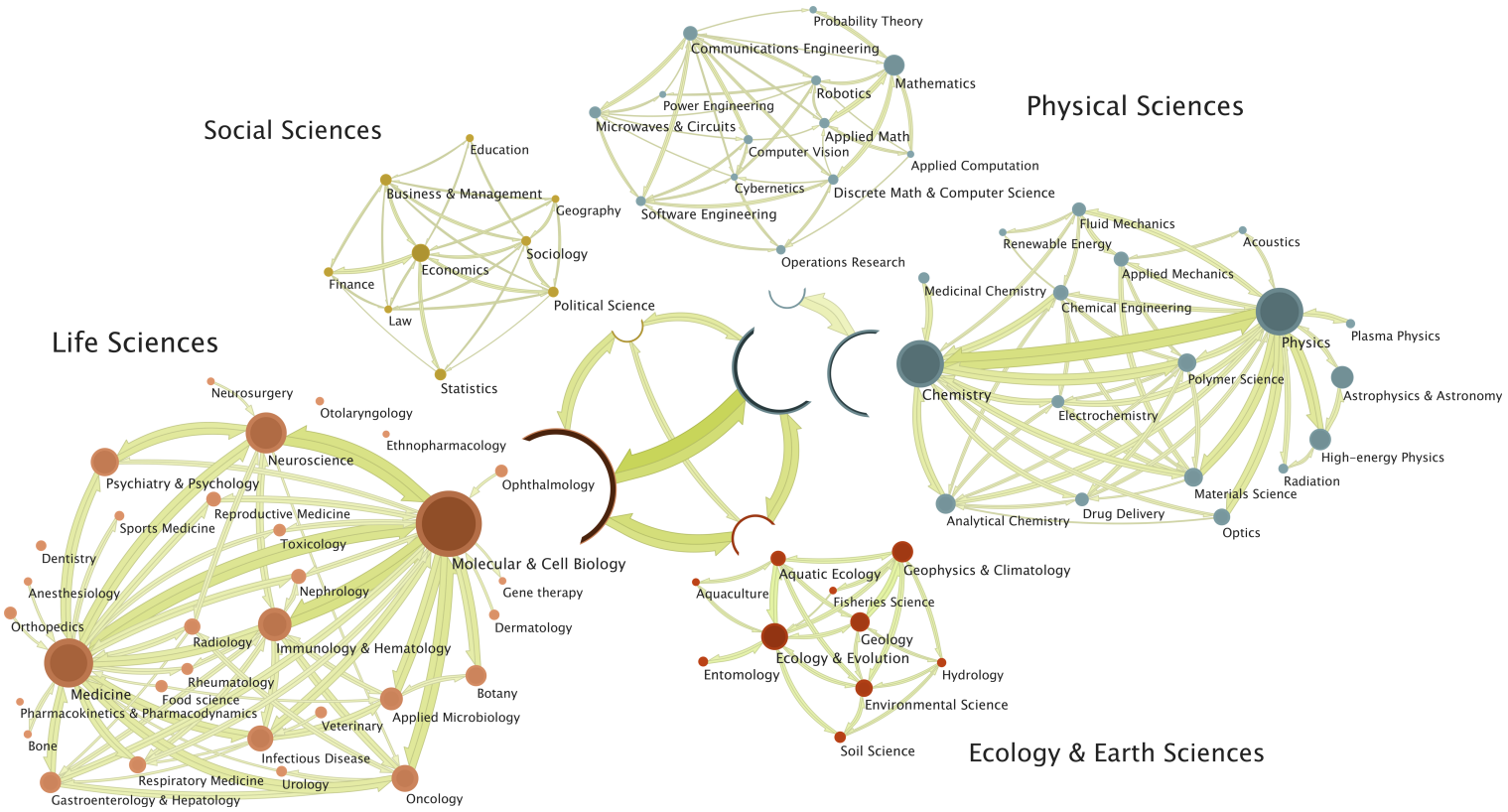
Examples of networks: collaboration networks

Part of the collaboration network of Paul Erdös (can be seen near the center). Hand-drawn by Ronald Graham. What is your Erdös number?



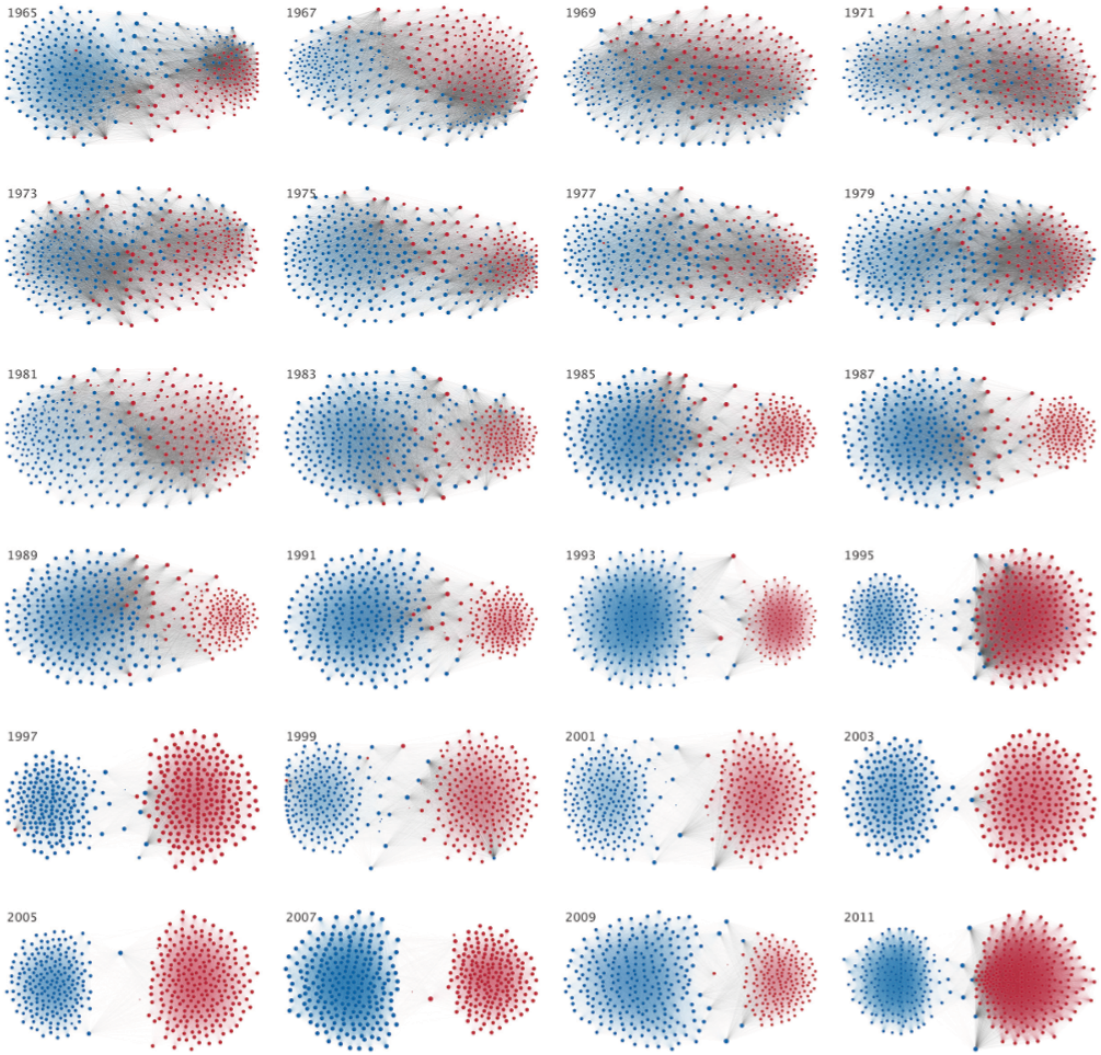
Examples of networks: mapping the sciences

Relationships between disciplines based on citations. Taken from “Multilevel Compression of Random Walks on Networks Reveals Hierarchical Organization in Large Integrated Systems” by Rosvall and Bergstrom (2011), PLOS ONE 6(4): e18209. Accessed through this link.



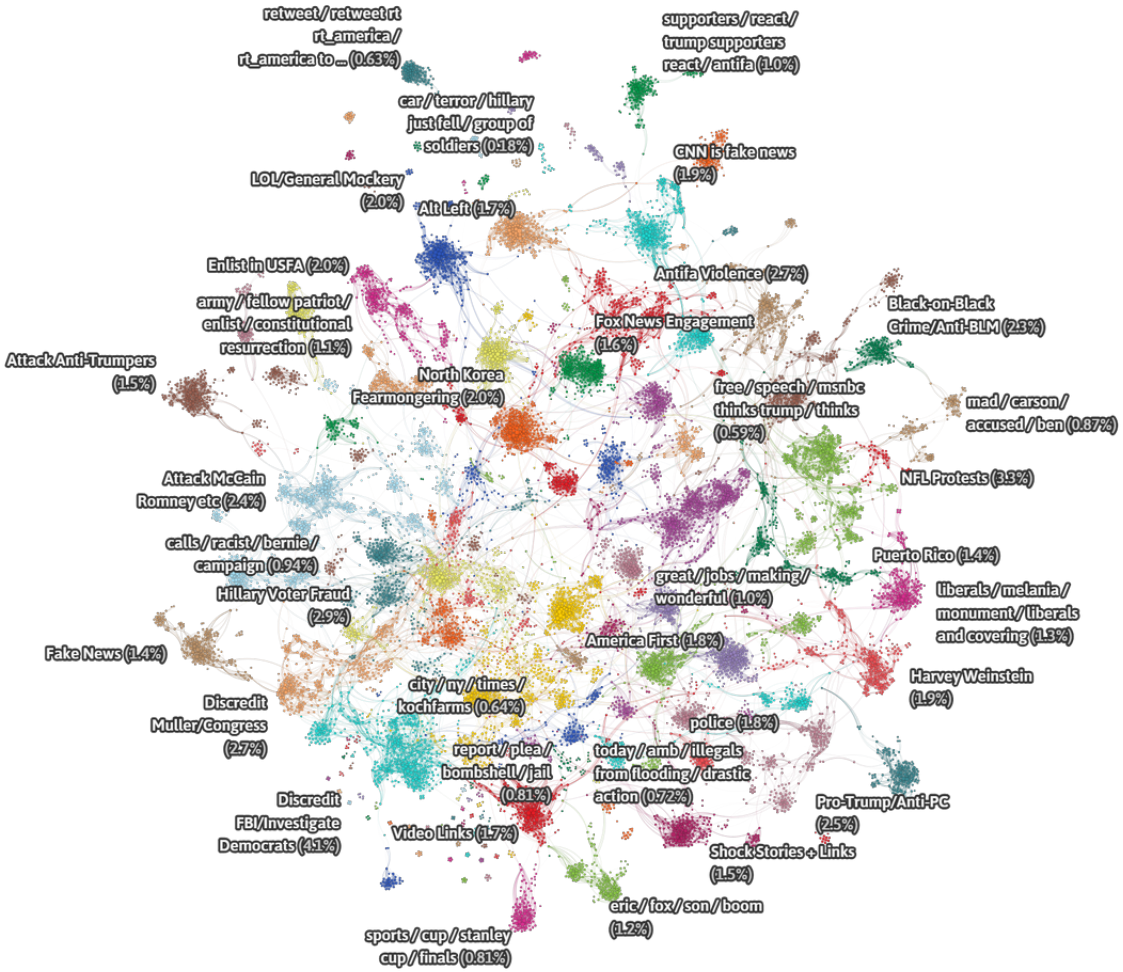
Examples of networks: from politics

Polarization in Senate voting from 1965 to 2011. Taken from this post.



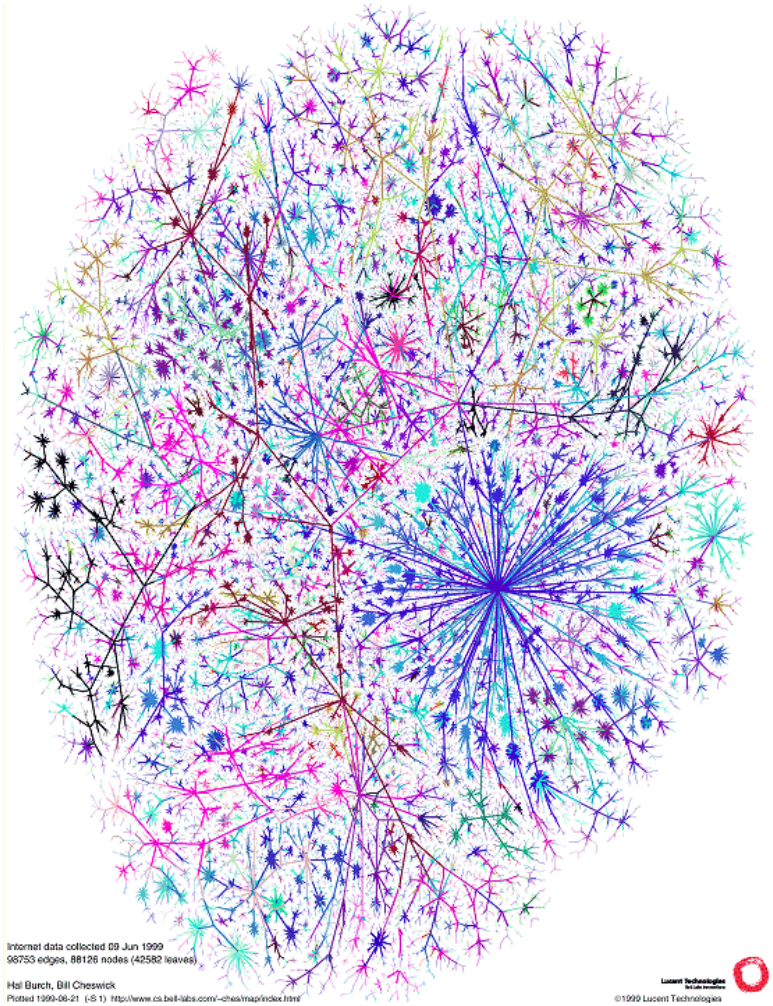
Examples of networks: from reddit

Network of pro-Trump subreddits during election period 2016.



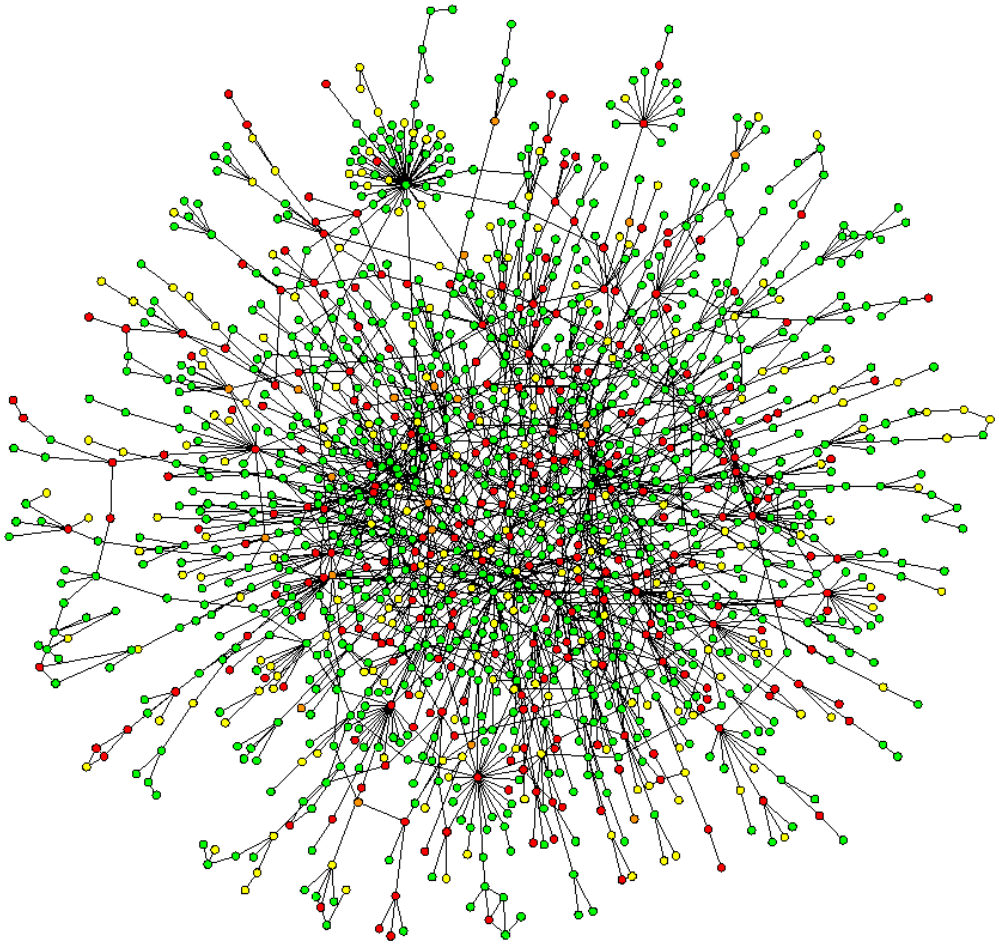
Examples of networks: the Internet

The directory of the main domains of the world wide web in 1999. Created by Bill Cheswick and accessed through here: http://www.cheswick.com/ches/map/index.html.



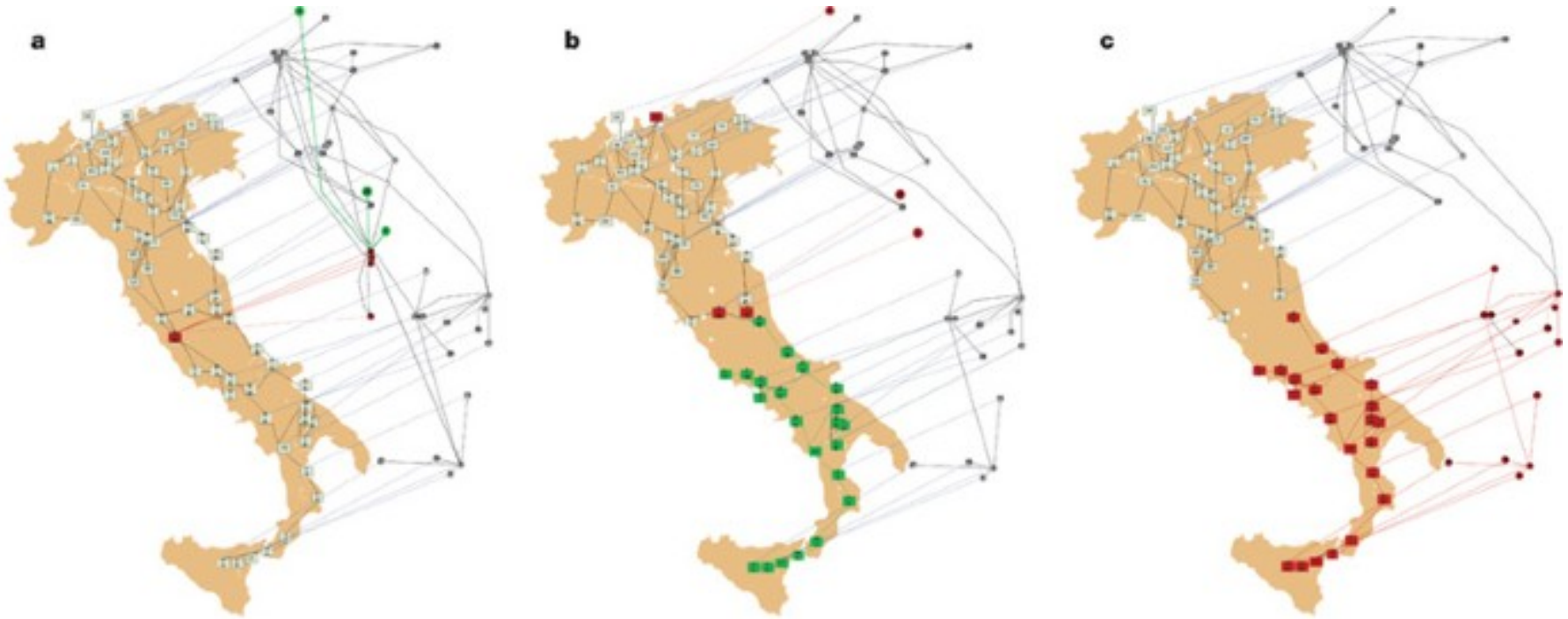
Examples of networks: biological networks

The protein-protein interaction network of Saccharomyces cerevisiae (source by Hawoong Jeong, KAIST, Korea).



Examples of networks: cyber-physical systems

The cascading failures that led to the catastrophic blackout of Northern Italy in 2003. Taken from “Catastrophic cascade of failures in interdependent networks" by Buldyrev, Parshani, Paul et al. Nature 464 (2010). https://doi.org/10.1038/nature08932.



Learning goals

The learning goal for the semester is *simple*:

1. Develop a theoretical and applied intuition for networks and the phenomena surrounding them.

Specifically, I will be very happy if, by the end of the semester, we are able to:

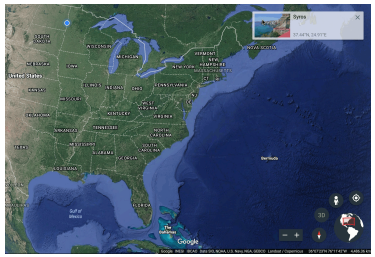
1. Formulate real-life problems as network analysis problems.
2. Use network representations, terminology, concepts in your research.
3. Solve network flow problems and combinatorial optimization problems using specialized techniques (such as the ones taught in class).
4. Rank entities and structures as far as their "importance" in a network. Identify "important" entities and structures.
5. Identify structures of interest (such as communities, partitions, cliques) in large-scale networks.

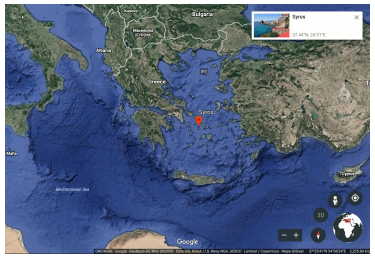
ILLINOIS

Who I am

- Ph.D. in ISE @ UF.
- Worked at NDSU, NCAT.
- Teaching Assistant Professor in ISE @ UIUC.
- OR, mathematical programming, combinatorial optimization, **network optimization and analysis**.
- Teaching IE 300 (Analysis of Data) and IE 532 (Analysis of Network Data).
- I come from a small island in the Aegean Sea in Greece

What do you mean "in the middle of the sea"?






Who I am

Problems I work on:

- **Network** analysis and optimization.
- Evacuation and disaster management.
- Biological **networks** and phylogenetics.
- Data analytics using **combinatorial optimization**.

Awards I have received:

- Graduate student **teaching** award (2012, UF).
- ISE **teaching** award (2012, UF).
- Now You See Me: Best paper award of the European Intelligence and Security Informatics Conference (2018).
- Best undergraduate **teaching** award (2019, NCAT).
- Best graduate **teaching** award (2019, NCAT).
- Sharp outstanding **teaching** award in Industrial Engineering (2020, UIUC).

ILLINOIS

How do I call you?

Over the years, I've been called or emailed as:

- Professor/Prof. Chrysafis/Chrys;
- Sir/Mister/Mr. Chrysafis/Chrys;
- Man/Dude/Bro;
- Chrysafis/Chrys;
- Dr. Vogiatzis/Dr. V./Dr. Chrys;
- Greek guy.

I'm 100% fine with:

- Professor/Prof. Vogiatzis/V./Chrysafis/Chrys;
- Dr. Vogiatzis/V./Chrysafis/Chrys;
- Chrysafis/Chrys.

Phonetically:

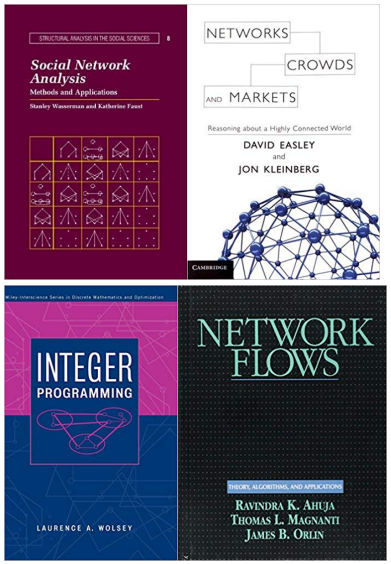
chree-SAH-fees vo-yah-JEES

For the last name it helps to think of VOYAGE + YAHTZEE.

ILLINOIS

Class logistics

- Class time:
  - TR 2:00pm–3:20pm at 1035 Campus Instructional Facility (CIF).
  - All lectures will be recorded!
- Textbooks that I will be using:
  - **Social Network Analysis: Methods and Applications by Faust and Wasserman**
  - **Networks, Crowds, and Markets** by Easley and Kleinberg.
  - **Network Flows** by Ahuja, Magnanti, Orlin.
  - **Integer Programming** by Laurence A. Wolsey.
  - **Highly** recommended, not required.



- Software:
  - You will also need to have access to an optimization solver software (`Gurobi`) and a network analysis package (`networkX`) in Python.
  - We will see how to install and use these packages in due time.
- Course materials:
  - All material (lecture notes, HW assignments, codes, examples, recordings, etc.) will be posted on Canvas.

ILLINOIS

Grading

Your letter grade in the class will be calculated based on:

1. Two take home exams, each worth 20% of your grade.
   - Open books/open notes, but **no collaboration** between students allowed.
     - You can ask me! I will be posting office hours throughout the take home period.
   - The second exam will be cumulative.
2. Homework assignments, approximately 4-5 throughout the semester (20%) – collaboration between students allowed and encouraged, so long as it is acknowledged.
3. In-class, small activities, approximately 5-6 throughout the semester (20%).
4. Term project (20%), which can be individual or done in a small group of up to 3 students – more details will be announced later in the semester.

ILLINOIS

What I know about you

...

Help me amend this by filling in the (anonymous) survey online. I will give you 10 minutes to go through it.

(If you are watching this at thome) The link to the survey is here (becomes active Tuesday 08/23 at 2pm and will stay open until Monday 08/29 at end-of-day):

https://surveys.illinois.edu/sec/87296876

ILLINOIS

Where to begin

- The “Seven Bridges of Königsberg”.
- Leonhard Euler in 1736.
- In the city of Königsberg in Prussia (now Kaliningrad, Russia):

“Can we cross every bridge **exactly once**
and end up back where we started?”

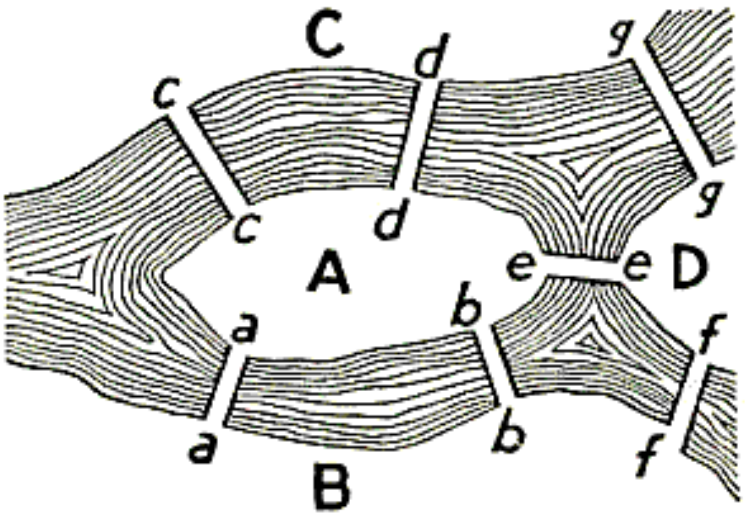
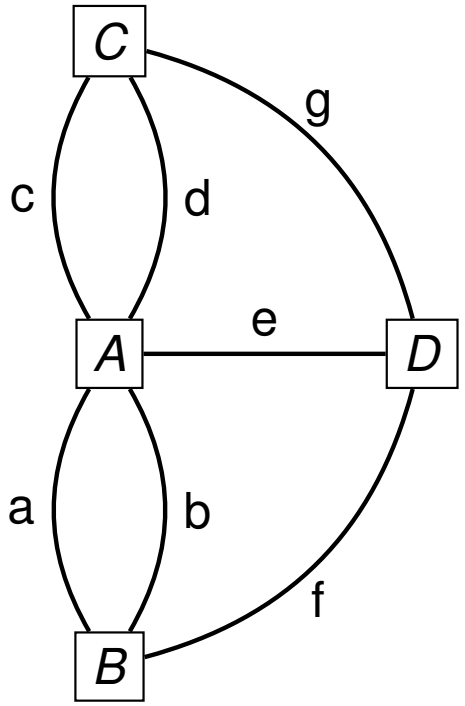


FIGURE 98. *Geographic Map: The Königsberg Bridges.*

$\Rightarrow$

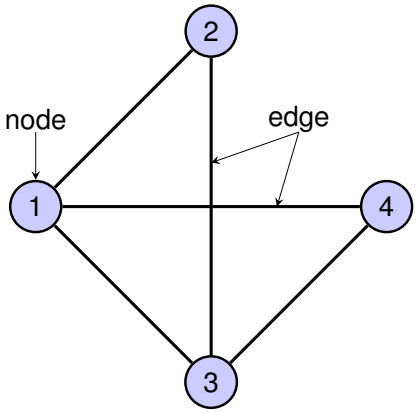


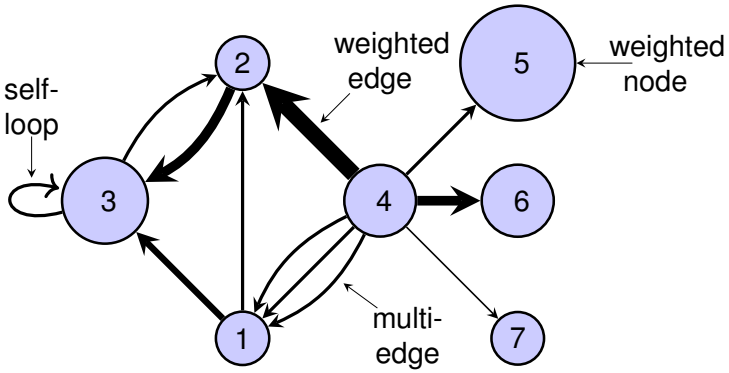
- This negative result gave birth to the field of graph theory.

Fundamentals

Given graph (sometimes termed network) $G(V, E)$:

- V: set of vertices (sometimes nodes), usually labeled $1, \ldots, n$.
- E: set of edges (sometimes arcs), usually labeled $1, \ldots, m$. Note that $E \subseteq V \times V$.
- *undirected edge/arc*: unordered pair of nodes $(i, j) = (j, i)$.
- *directed edge/arc*: ordered pair of nodes $(i, j) \neq (j, i)$.
- *multigraph*: graph with multiple edges connecting the same two nodes.
- *self-loop*: an edge (i, i).
- *weighted* graph: a graph with weights on the edges/vertices.





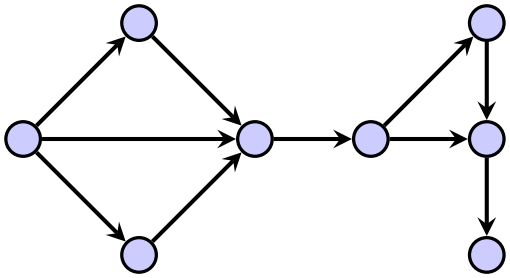
Simple graphs

A graph without self-loops or multi-edges is called **simple**. Most graphs in our class will be simple (directed and undirected).

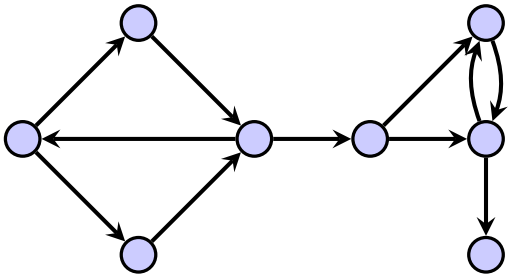
Nodes: weighted/unweighted, attributed.

Edges: weighted/unweighted, directed/undirected, attributed (e.g., signed), multi-edges.

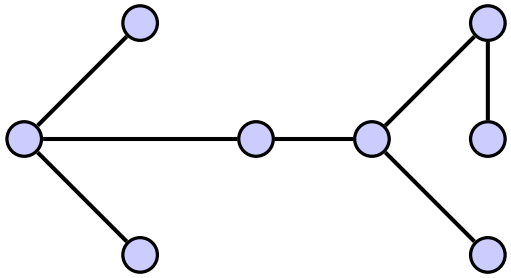
Graph: sparse/dense, connected/disconnected, tree, acyclic, hypergraph, multiplex, k-partite (e.g., bipartite).



A directed, **acyclic** graph.



A directed but **not** acyclic graph.

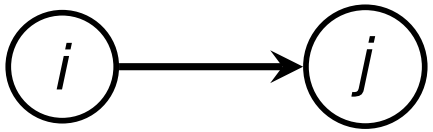


A tree (always acyclic).

ILLINOIS

Directed/undirected

A directed/undirected edge has two **endpoints**: node i and j.



We can express the directed relationship in a number of ways:

- i is called the tail and j the head of the edge;
- edge (i, j) emanates from node i and terminates at node j;
- edge (i, j) is outgoing from node i and incoming to node j.

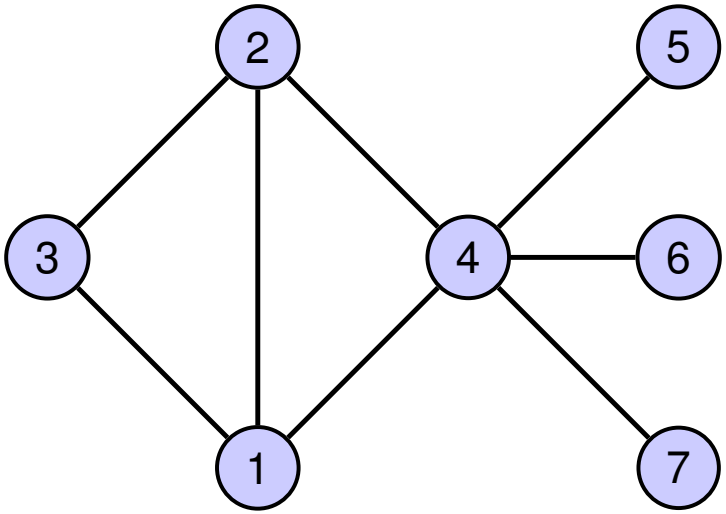
For both directed and undirected graphs, we also say that:

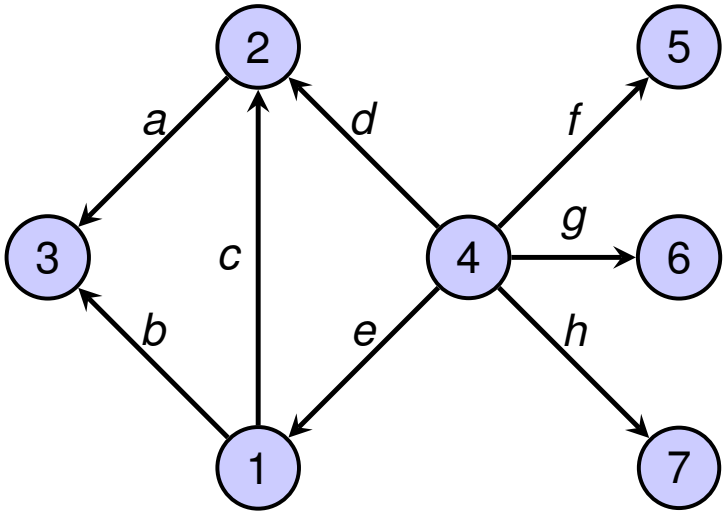
- edge (i, j) is incident to nodes i and j;
- nodes i and j are adjacent.

Directed/undirected

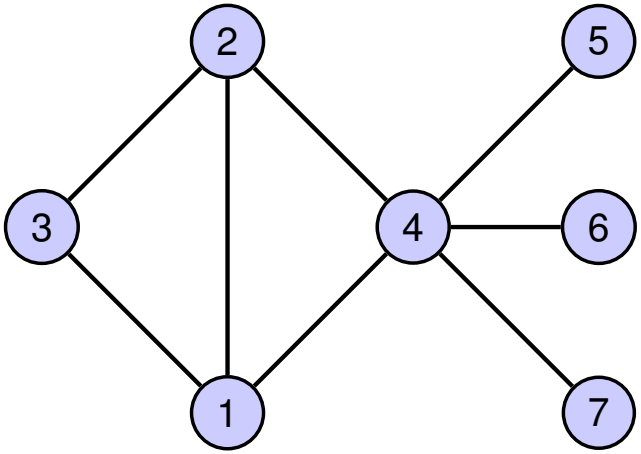
Graphs can be directed or undirected:

1. undirected graphs: with symmetric (reciprocal) edges.
   Typically represented by an **adjacency** matrix A, where $a_{ij} = 1$ if nodes i and j are connected by an edge.
2. directed graphs: with directed edges.
   Typically represented by a **node-arc** incidence matrix A, where for every (*directed*) edge $e = (i, j)$, we have $a_{ei} = 1$, $a_{ej} = -1$, $a_{ek} = 0$, for $k \neq i, j$.

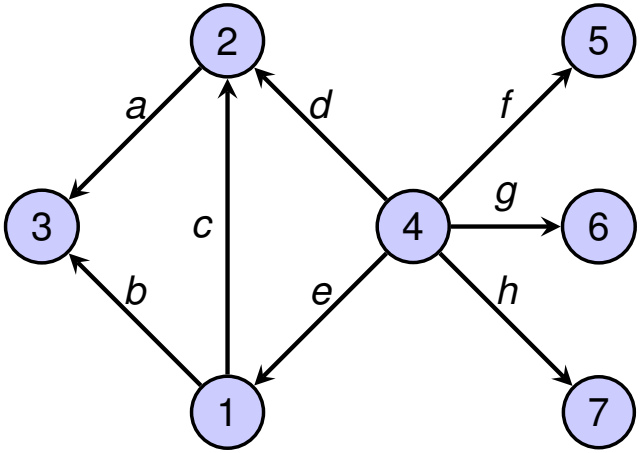




Representing directed and undirected networks



$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$



$$A = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 \end{pmatrix} \end{array}$$

ILLINOIS

How to define a network

In general, you need to provide two* things:

1. What are your vertices?
2. What are your edges?
3. Bonus question: what are your flows, if any?

For example, consider a water distribution network:

1. Vertices = homes, offices, factories, theaters, restaurants, etc.
2. Edges = pipes
3. Flow = water

Extra observation: water flow has a *direction*.

ILLINOIS

Biological systems

For the same family of systems, we can answer the previous questions in multiple, different ways. Remember that networks are **models** for representing reality, and more than one representation can be true!

Protein-protein interaction networks:

1. Vertices=proteins
2. Edges=bonding

Metabolic networks:

1. Vertices=metabolites
2. Edges=reaction

Gene regulatory networks:

1. Vertices=gene
2. Edges=regulation

ILLINOIS

More definitions

- **Walk** (or chain): a series of edges, such that each edge has exactly one node in common with the previous one without any respect for directions.

$$v_1 v_2 \dots v_k, \text{ such that } (v_i, v_{i+1}) \in E.$$

- **Directed walk**: a version of the walk where we respect all directions.
- **Path**: a walk, without any repetitions of nodes.

$$v_1 v_2 \dots v_k, \text{ such that } v_i \neq v_j.$$

- **Directed path**: a directed version of a path.
- **Cycle**: a path where the starting and ending nodes are the same.

$$v_1 v_2 \dots v_k, \text{ such that } v_1 = v_k.$$

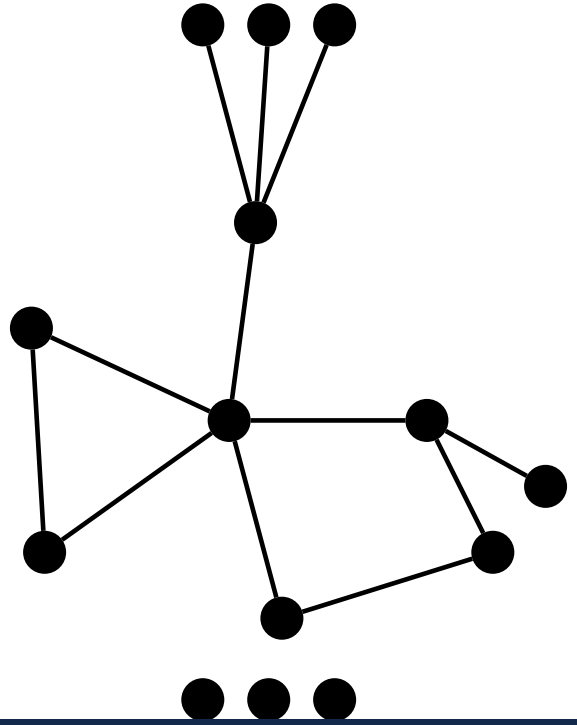
- **Directed cycle**: a directed path where the starting and ending nodes are the same.

A directed graph without any directed cycles is called acyclic.

ILLINOIS

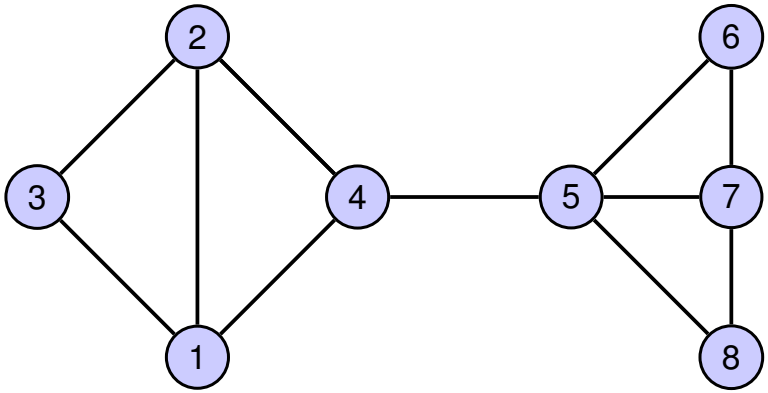
Neighborhoods and subgraphs

- The **degree** of a node is the number of edges the node is incident to. It can be further divided into **in-degree** and **out-degree** (for incoming and outgoing connections) for directed graphs.
- **Open neighborhood** of a node: $N(i) = \{j \in V : (i,j) \in E\}$.
- Nodes reachable within k hops from i: $N^k(i)$.
- **Closed neighborhood** of a node: $N[i] = N(i) \cup \{i\}$.
- **Induced subgraphs:** For $S \subset V$, $G[S]$ has a vertex set $V[G[S]] = S$ and an edge set $E[G[S]] = \{(i,j) \in E : i,j \in S\}$.
  - For $S \subset V$, $G[S]$ has a vertex set $V[G[S]] = S$ and an edge set $E[G[S]] = \{(i,j) \in E : i,j \in S\}$.

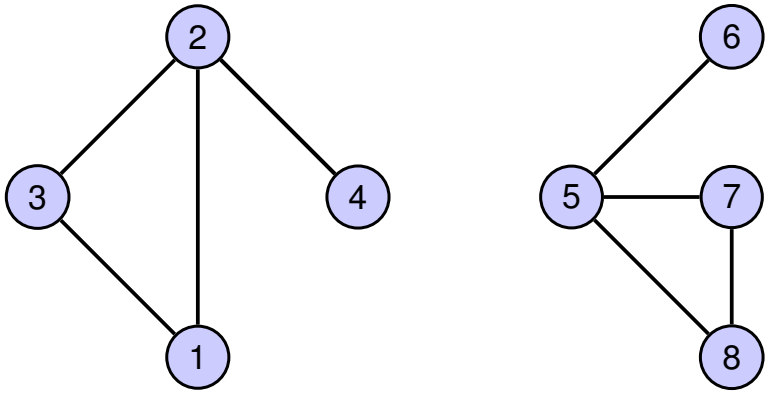


Connectivity

A graph is *connected* if there exists at least one path from every node to every other node.



G_1: **connected**
1 component



G_2: **disconnected**
2 components

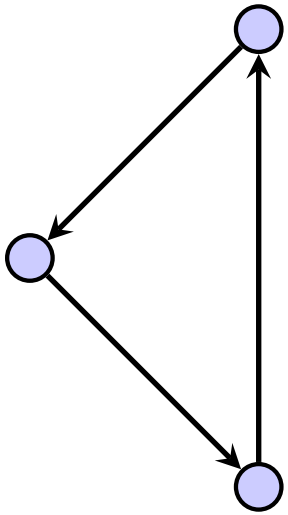
The maximal, connected subgraphs of a graph are its **components**.

- **maximal**: cannot be made any larger by adding more nodes.
  Mathematically: for any property Π, $S \subseteq \Omega$ is maximal if S satisfies Π but $\nexists i \in \Omega \setminus S$ such that $S \cup \{i\}$ satisfies Π.
- In G_2 $\{1, 2, 3, 4\}$ is maximal and connected – so it is a component.
- In G_2 $\{5, 6\}$ is not maximal – not a component.

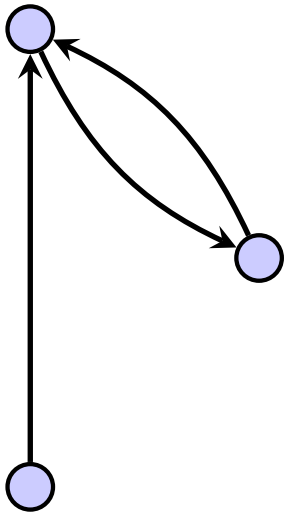
ILLINOIS

Connectivity for directed netwoks

- A directed graph is connected if there exists a path (no direction requirement) from every node to every other node.
- A directed graph is *strongly connected* if there exists a directed path from every node to every other node.



Strongly connected.



Not strongly connected.

Eulerian cycle

- **Eulerian path**: a path that uses every edge *exactly* once.
- **Eulerian cycle**: an Eulerian closed path (that starts and ends at the same node).

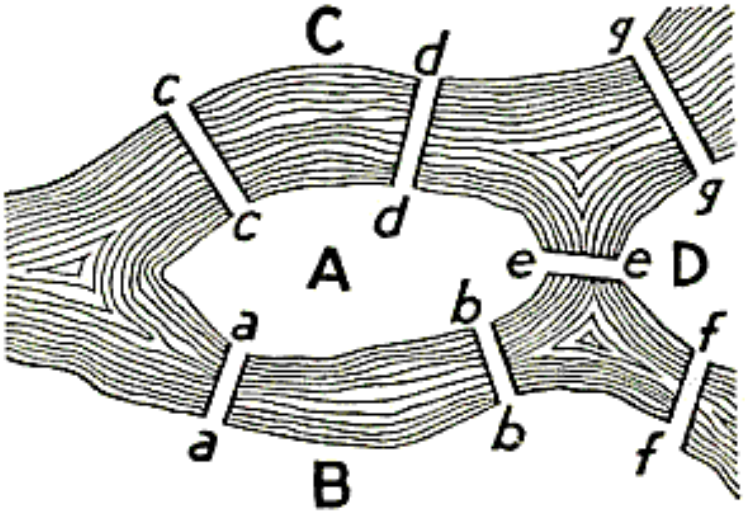
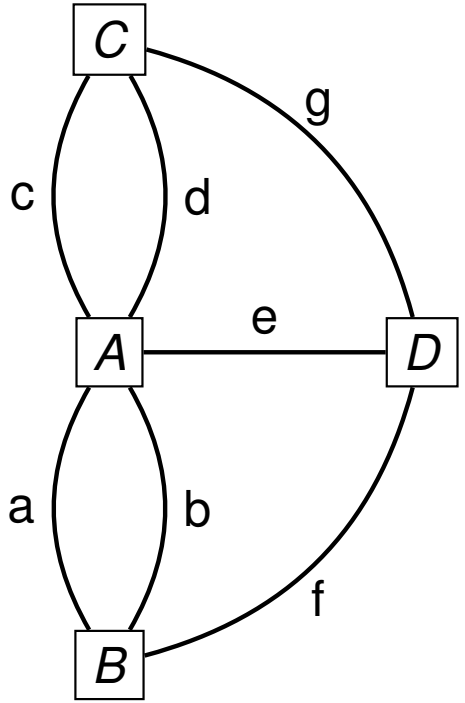


FIGURE 98. *Geographic Map: The Königsberg Bridges.*

$\Rightarrow$



Question: Given a graph $G(V, E)$, does there exist an Eulerian cycle?

Question: If there is one, can we find it?

We similarly define a **Hamiltonian path/cycle**, where we want to use every node exactly once.

Theorem (Eulerian cycle existence)

A graph $G(V, E)$ has an Eulerian cycle if and only if the graph is connected and every node has an even degree.

Proof:

- $\Rightarrow$: easier to show.
  - Assume G is not connected. Then no cycle that uses all edges exists (even allowing for repetitions).
  - Assume G has a node with an odd degree. That node cannot be part of a cycle.
- $\Leftarrow$: constructive proof (due to Fleury, 1883).
  1. Start at any node.
  2. Choose any edge so long as removing it does not disconnect the graph. If no such edge exists, then pick the remaining edge.
  3. Traverse that edge and remove it from the graph.
  4. When done, if there are no edges left, this is an Euler cycle; otherwise, there is no Euler cycle.

Notes:

a) Correctness?

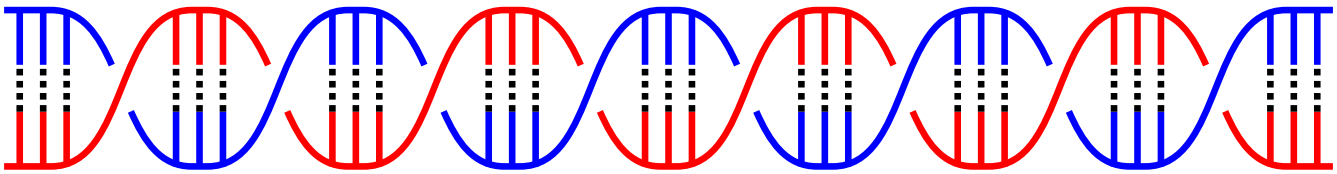
b) Complexity?

c) Any *better* algorithms?

From the theorem, we also get that

- an Eulerian path exists if and only if the graph is connected and only two nodes have an odd degree.
- in a directed graph an Eulerian cycle exists if and only if the graph is connected and for every node its in-degree is equal to the out-degree.

Application: DNA sequencing



Remember that DNA consists of 4 nucleotides A, T, G, C in succession.

The process: getting multiple copies of a genome, and then obtaining "reads", i.e., cutting a bigger sequence of DNA into smaller subsequences, each of length k.

$k = 4$**:** $ATGCCAT \implies \{ATGC, TGCC, GCCA, CCAT\}$.

$k = 3$**:** $ATGCCAT \implies \{ATG, TGC, GCC, CCA, CAT\}$.

$k = 2$**:** $ATGCCAT \implies \{AT, TG, GC, CC, CA, AT\}$.

Think shredding a book and then trying to reconstruct the pages.

DNA sequencing: reconstructing the full sequence of length n from the available set of $n - k + 1$ subsequences of length k each.

ILLINOIS

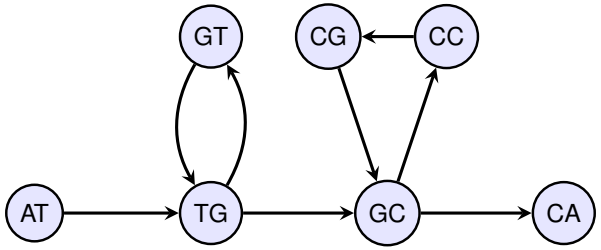
DNA sequencing and Eulerian paths

Let s be the set of subsequences of length k. We construct a graph as follows:

- nodes are all possible length $k-1$ subsequences obtained from s.
- edges between two nodes if the two of them together produce one of the subsequences in s.

The claim? An Eulerian path is a valid, feasible full DNA sequence.

For example, say we have obtained the following subsequences: $\{ATG, CCG, CGC, GCA, GCC, GTG, TGC, TGT\}$. The graph would be:



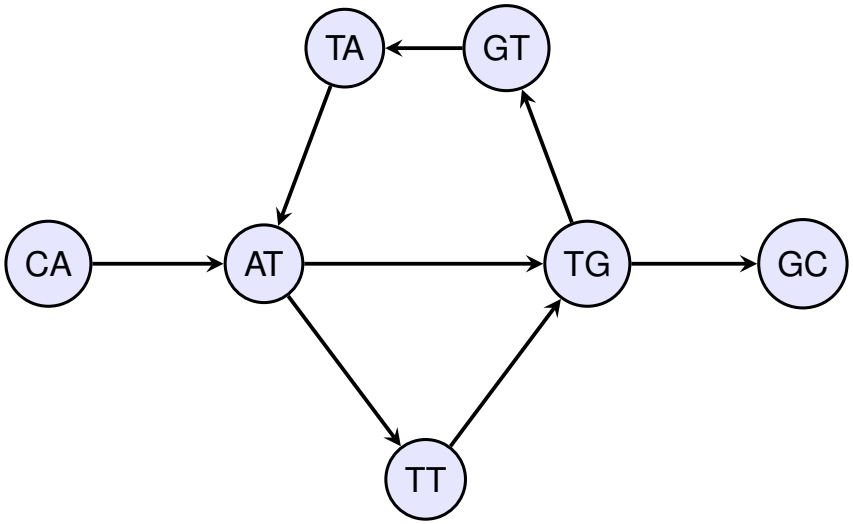
With an Eulerian path of AT→TG→GT→TG→GC→CG→CC→GC→CA, leading to a sequence of ATGTGCCGCA.

DNA sequencing and Eulerian paths

- The Eulerian path is not always unique.
- Actually, the opposite is typically the case.

For instance, consider $s = \{CAT, ATT, ATG, TAT, TTG, TGC, TGT, GTA\}$.

The graph would be:



We may now show that there are two Eulerian paths:

1. CA→AT→TT→TG→GT→TA→AT→TG→GC ⇒ **CATTGTATGC**.
2. CA→AT→TG→GT→TA→AT→TT→TG→GC ⇒ **CATGTATTGC**.

Hence, we have a 50% chance of getting the correct original sequence.

Hamiltonian path/cycle

- *Hamiltonian path*: a path that includes all nodes *exactly* once.
- *Hamiltonian cycle*: a Hamiltonian closed path.

Contrary to Eulerian cycles (which are "easier"), the question whether a graph has a Hamiltonian cycle is $\mathcal{NP}$-complete (technical term meaning that it *cannot* (?) be answered efficiently).

Does a graph have a Hamiltonian cycle?

1952. **Dirac's theorem**: Let $G(V, E)$ be a graph with $|V| = n \geq 3$ and each node i has degree $d_i \geq n/2$. Then, G has a Hamiltonian cycle.

1962. **Ore's theorem**: Let $G(V, E)$ be a graph with $|V| = n \geq 3$ and any two non-adjacent nodes i, j have degrees such that $d_i + d_j \geq n$. Then, G has a Hamiltonian cycle.

1976. **Bondy-Chvátal theorem**: A graph $G(V, E)$ is Hamiltonian if and only if its closure, $cl(G)$, is Hamiltonian. The closure operator is as follows: keep adding edges (i, j) for any two non-adjacent nodes i, j such that $d_i + d_j \geq n$.

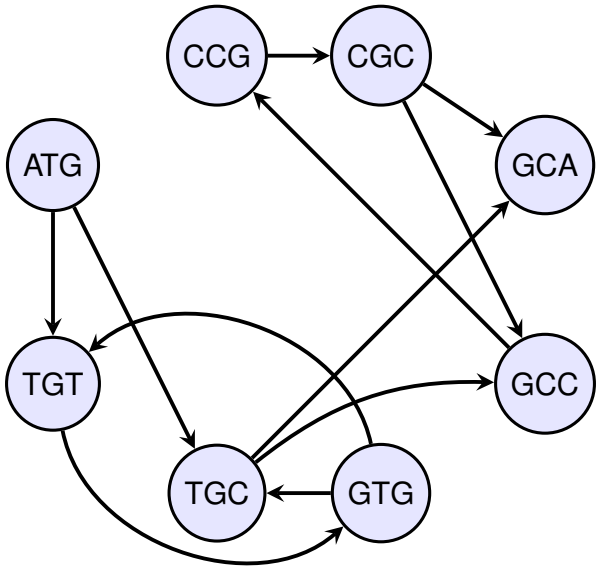
ILLINOIS

DNA sequencing and Hamiltonian paths

Again, we use s for the set of subsequences of length k. We now construct a graph as follows:

- nodes are the elements in s.
- edges between two nodes if the two of them "share" $k-1$ nucleotides.
  - That is, if the last $k-1$ nucleotides of the first node match the first $k-1$ nucleotides of the second one.

The claim? A **Hamiltonian** path is a valid, feasible full DNA sequence.
Using the earlier example ($\{ATG, CCG, CGC, GCA, GCC, GTG, TGC, TGT\}$), the resultant graph is:



With a Hamiltonian path of ATG→TGT→GTG→TGC→GCC→CCG→CGC→GCA, leading to the sequence of ATGTGCCGCA.

Palmer's algorithm

Ore's observation leads to a pretty efficient algorithm[2] in **dense graphs** (i.e., when Ore's theorem holds):

1. Arrange all nodes of a graph in a cycle (no matter whether consecutive nodes are adjacent or not for now).
2. For the first pair of nodes v_i, v_{i+1} that are **not** adjacent, identify "crossing chords" to some nodes v_j, v_{j+1} such that v_i is adjacent to v_j and v_{j+1} is adjacent to v_{j+1}.
   - Why does such a pair exist?
3. Finally, re-arrange the cycle nodes as v_i, v_j and v_{j+1}, v_{i+1}.

[2] Palmer, E. M., The hidden algorithm of Ore's theorem on Hamiltonian cycles, *Computers & Mathematics with Applications*, Vol. 34 no. 11, pp. 113–119, doi:10.1016/S0898-1221(97)00225-3.

Overview of IE 532

This lecture today is a nice example of what we'll do in IE 532:

- See many interesting **network problems**.
- Discuss **algorithms** to solve them.
  - correctness, complexity, implementations.

Some topics:

- Shortest paths.
- Network flows.
- Spanning trees.
- Integer programming techniques.
- Network structures.
- Centrality.
- Communities, modularity.
- Graph spectral properties.

ILLINOIS

Next time

1. Coding for the Eulerian path problem (using `networkx`).
2. Coding for the Hamiltonian cycle problem (implementing Palmer's).
   - Will help us get accustomed to network problems and using Python to solve them.
3. Matching in graphs.

4. Installing necessary material, including:
   - `networkx`.
   - `Gurobi`.

- Next week (Lectures 3–4): we will see an overview of optimization and mathematical modeling (with examples).

ILLINOIS

Before you head out

These are the *Kaliningrad* bridges today.

Still no Eulerian cycle! But, there now exists an Eulerian path!