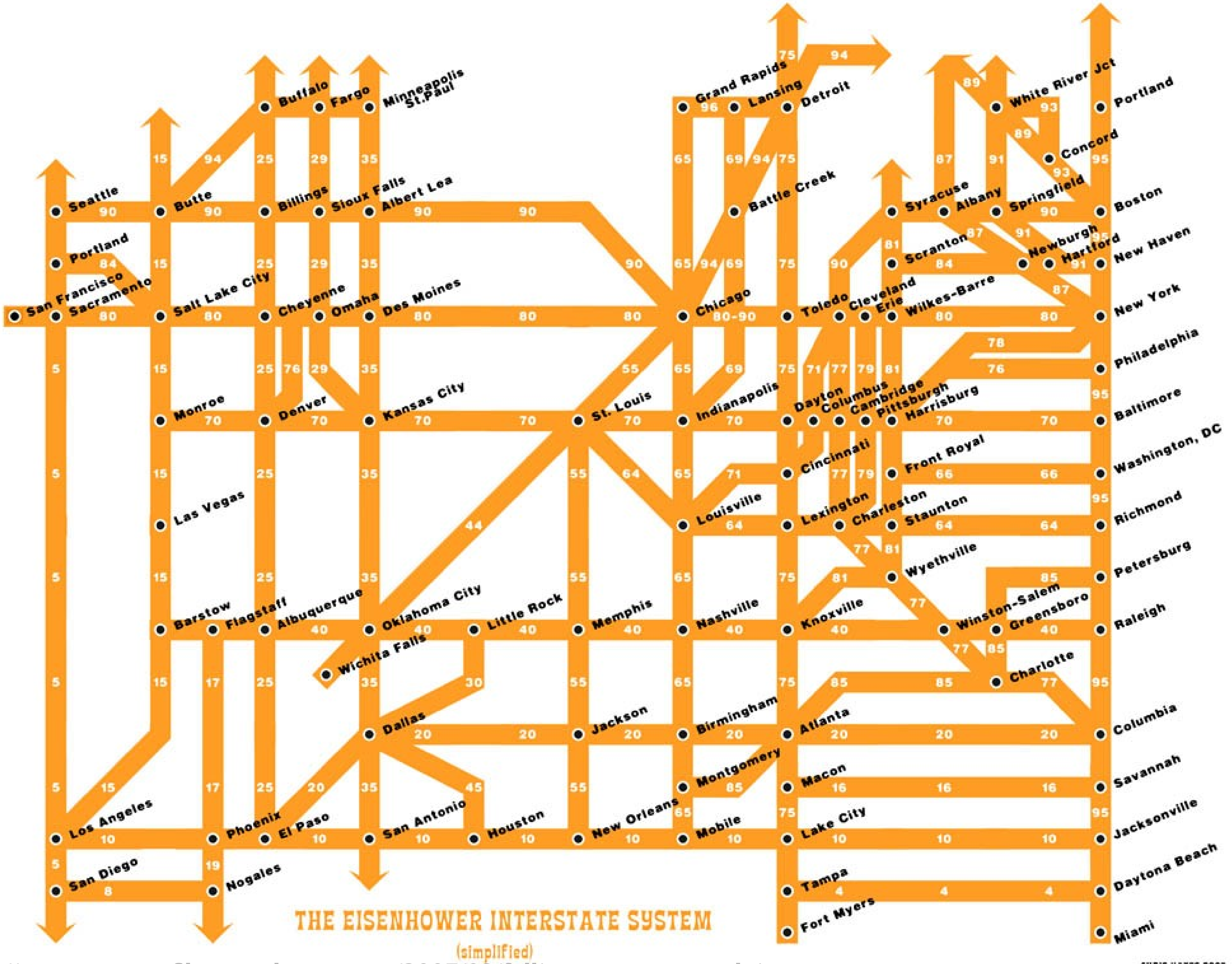
Graph Theory

Part One



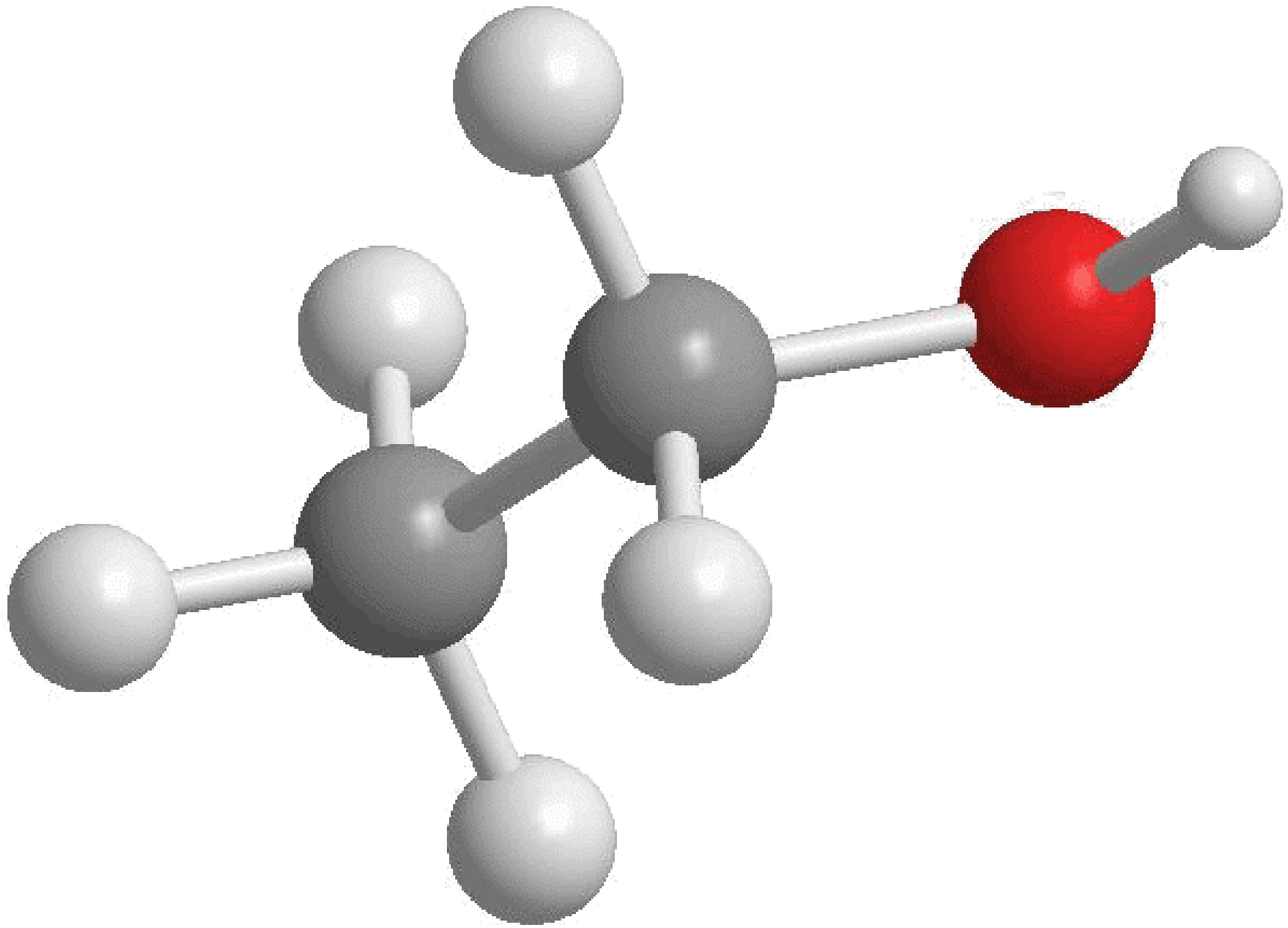
THE EISENHOWER INTERSTATE SYSTEM

(simplified)

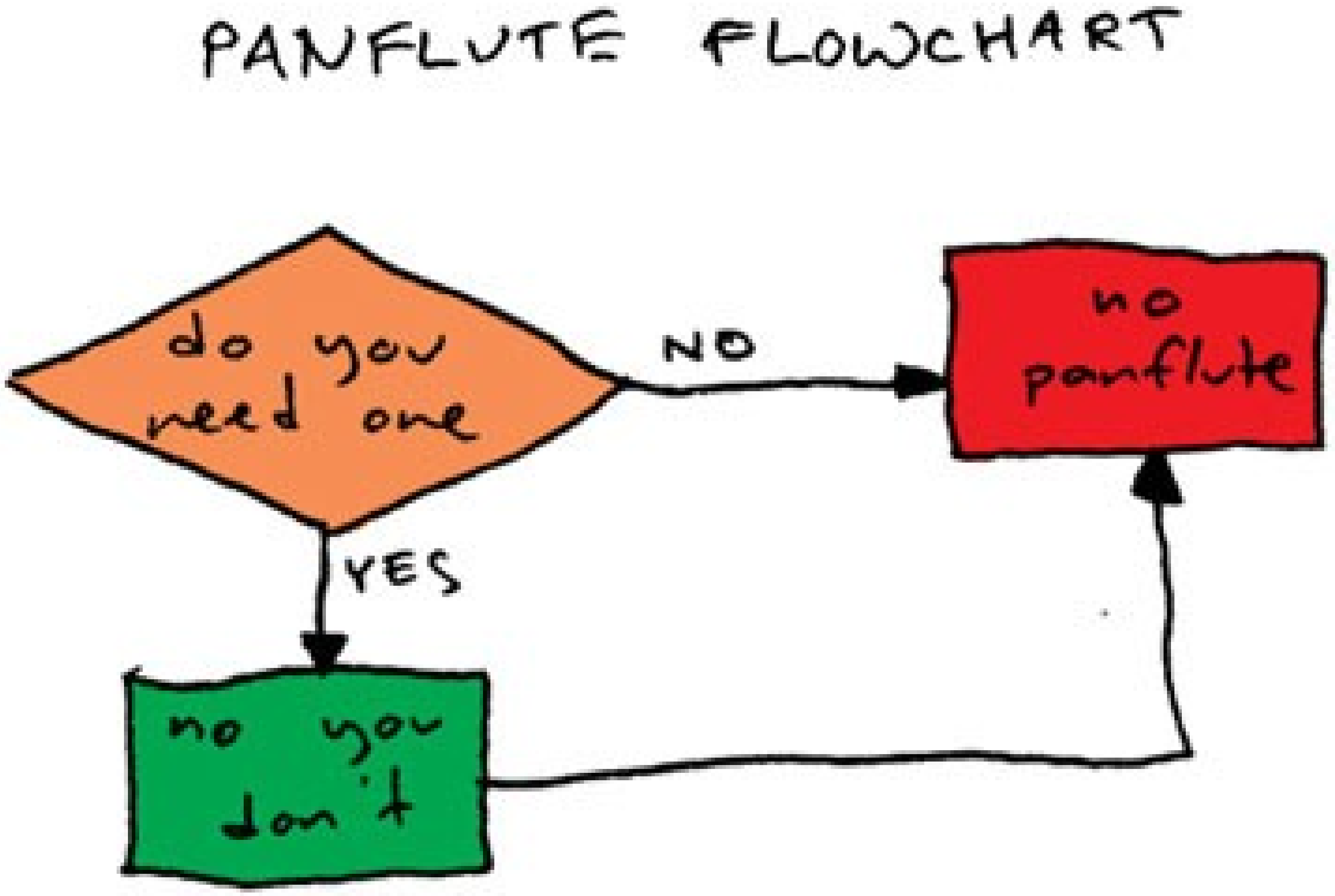
CHRIS YATES 2007

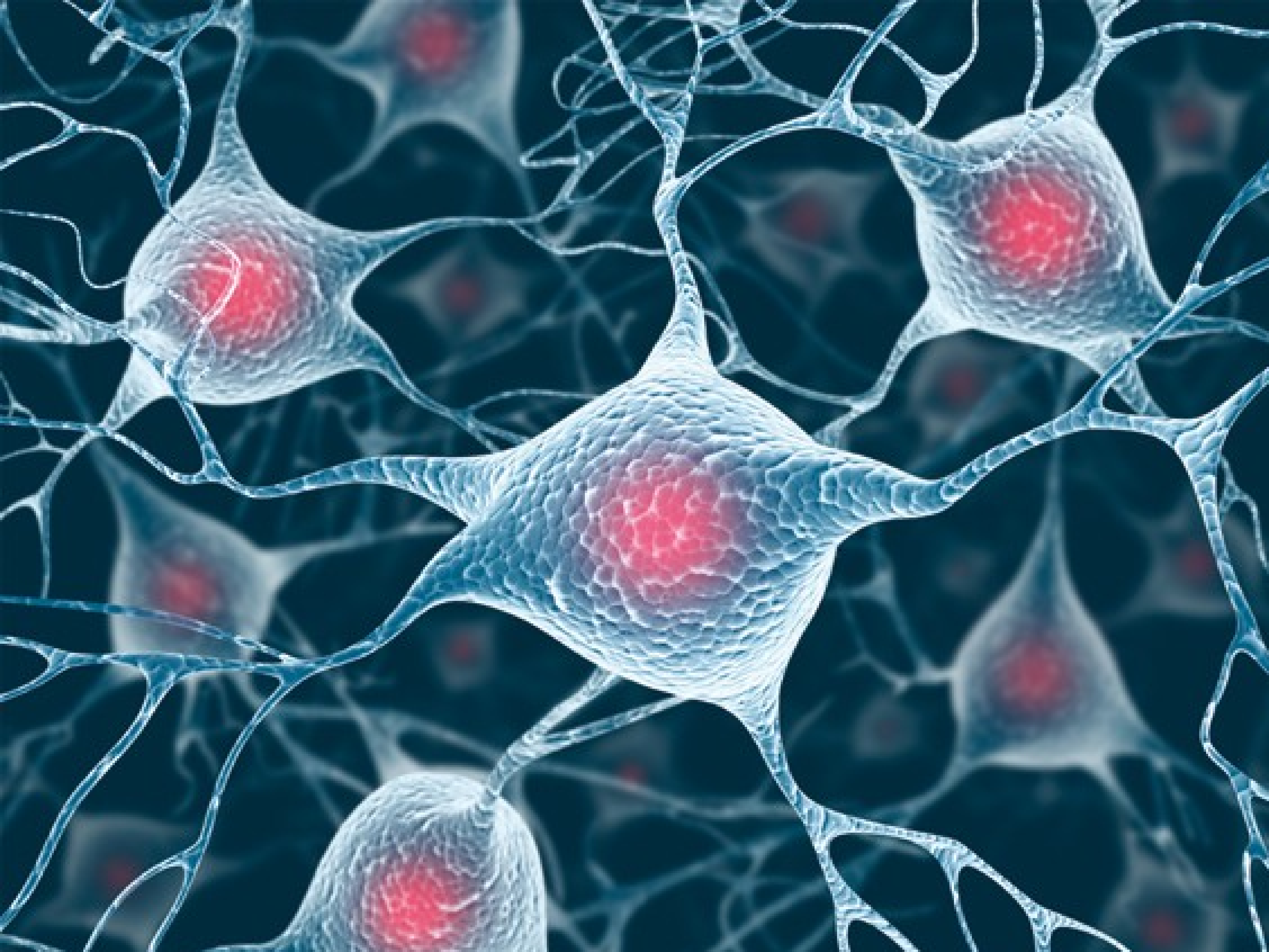
http://strangemaps.files.wordpress.com/2007/02/fullinterstatemap-web.jpg

Chemical Bonds



http://4.bp.blogspot.com/-xCtBJ8lKHqA/Tjm0BONWBRI/AAAAAAAAAK4/-mHrbAUOHHg/s1600/


PANFLUTE FLOWCHART
do you need one
NO
no panflute
YES
no you don't

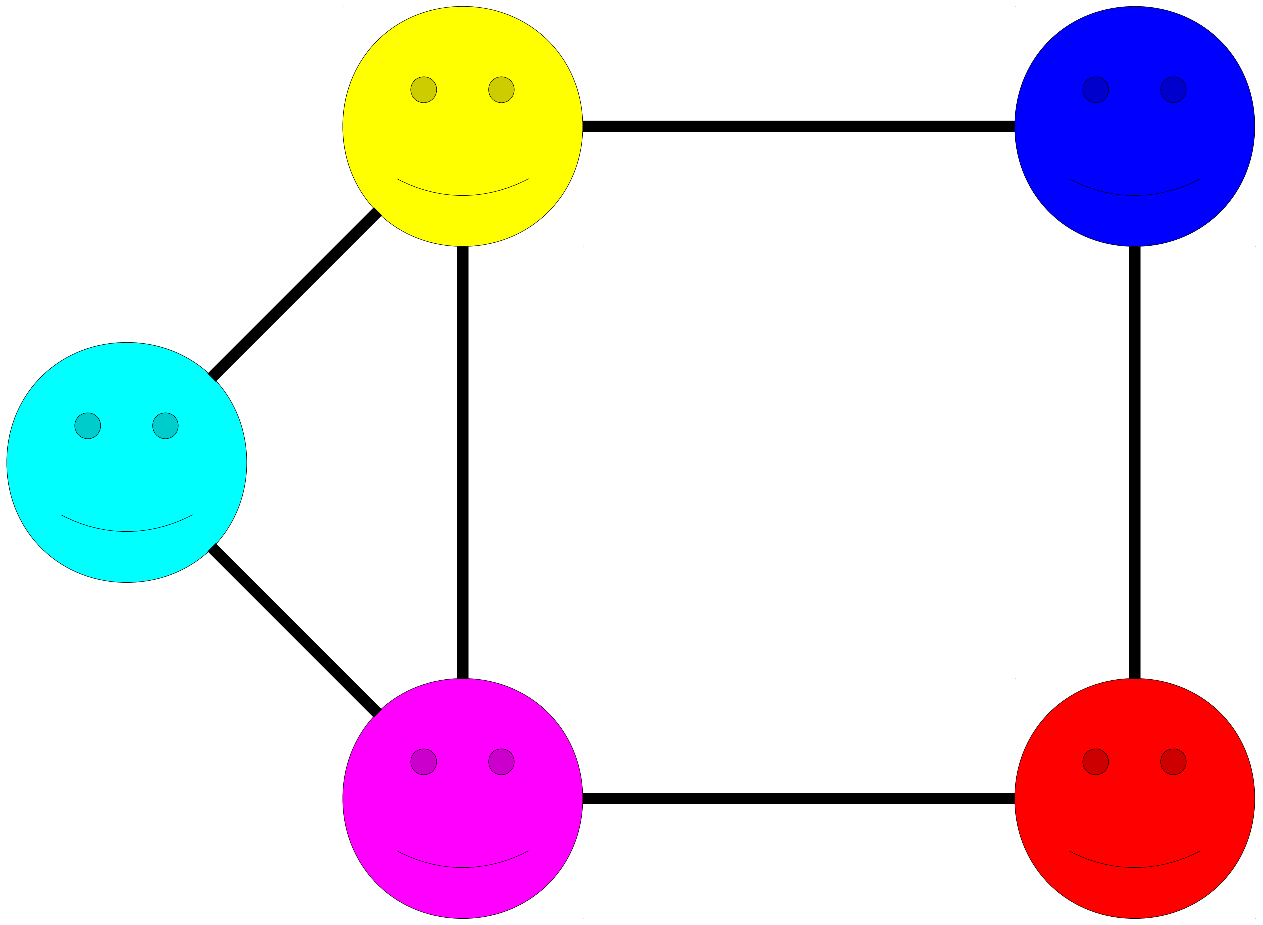


facebook®

What's in Common

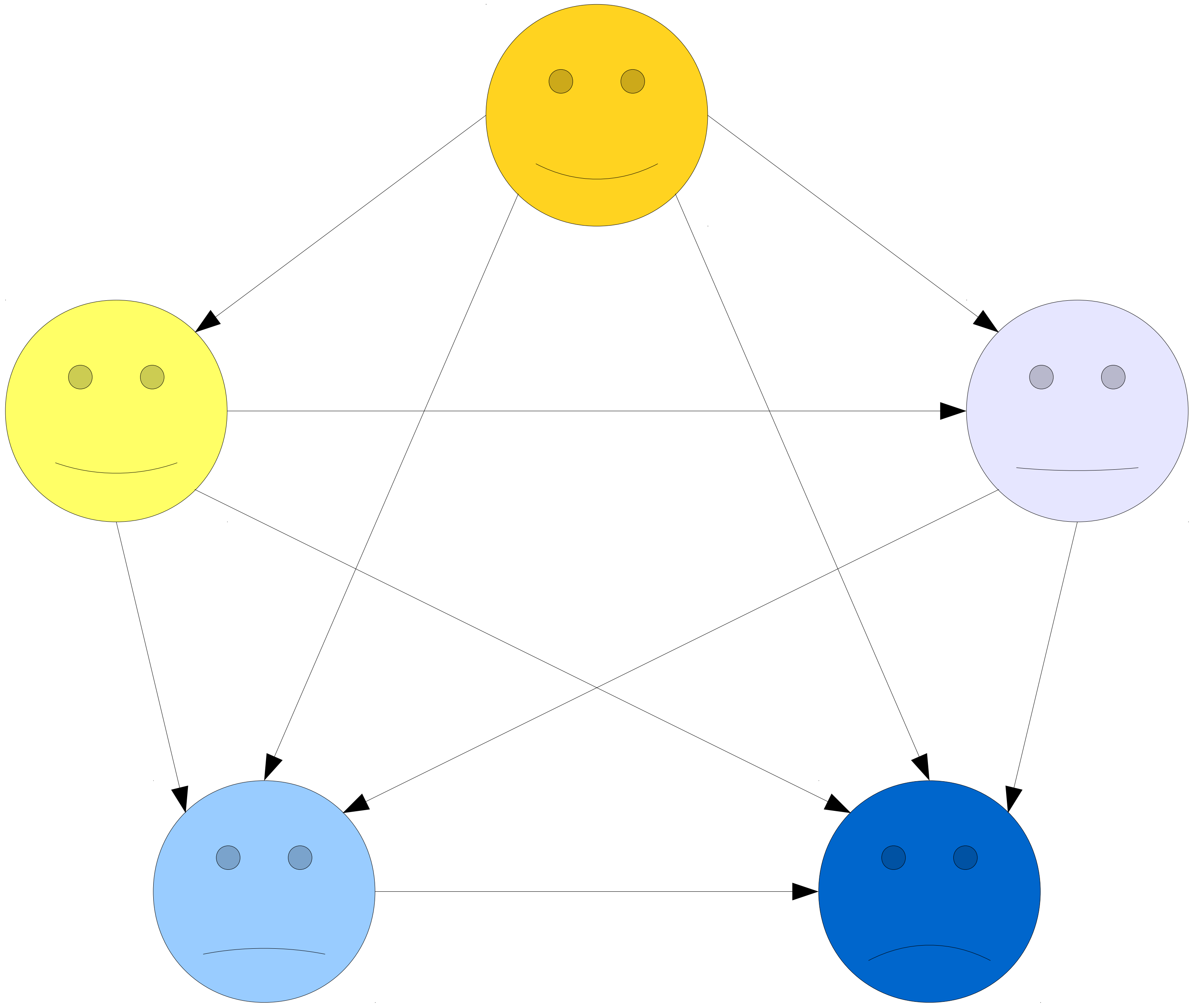
- Each of these structures consists of
  - a collection of objects and
  - links between those objects.
- ***Goal:*** find a general framework for describing these objects and their properties.

A ***graph*** is a mathematical structure for representing relationships.

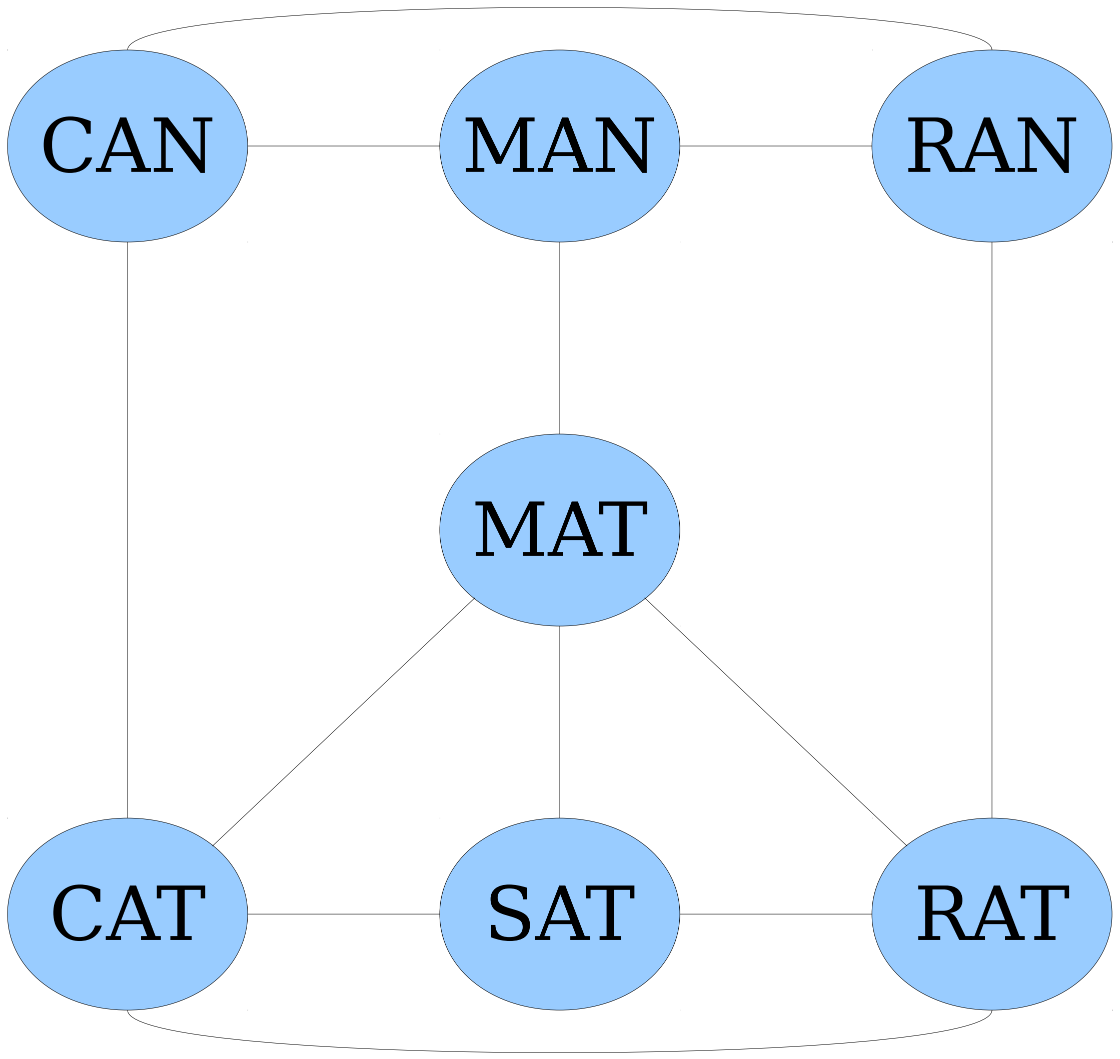


A graph consists of a set of ***nodes*** (or ***vertices***) connected by ***edges*** (or ***arcs***)

Some graphs are ***directed***.



Some graphs are ***undirected***.



Going forward, we're primarily going to focus on undirected graphs.

The term “graph” generally refers to undirected graphs with a finite number of nodes, unless specified otherwise.

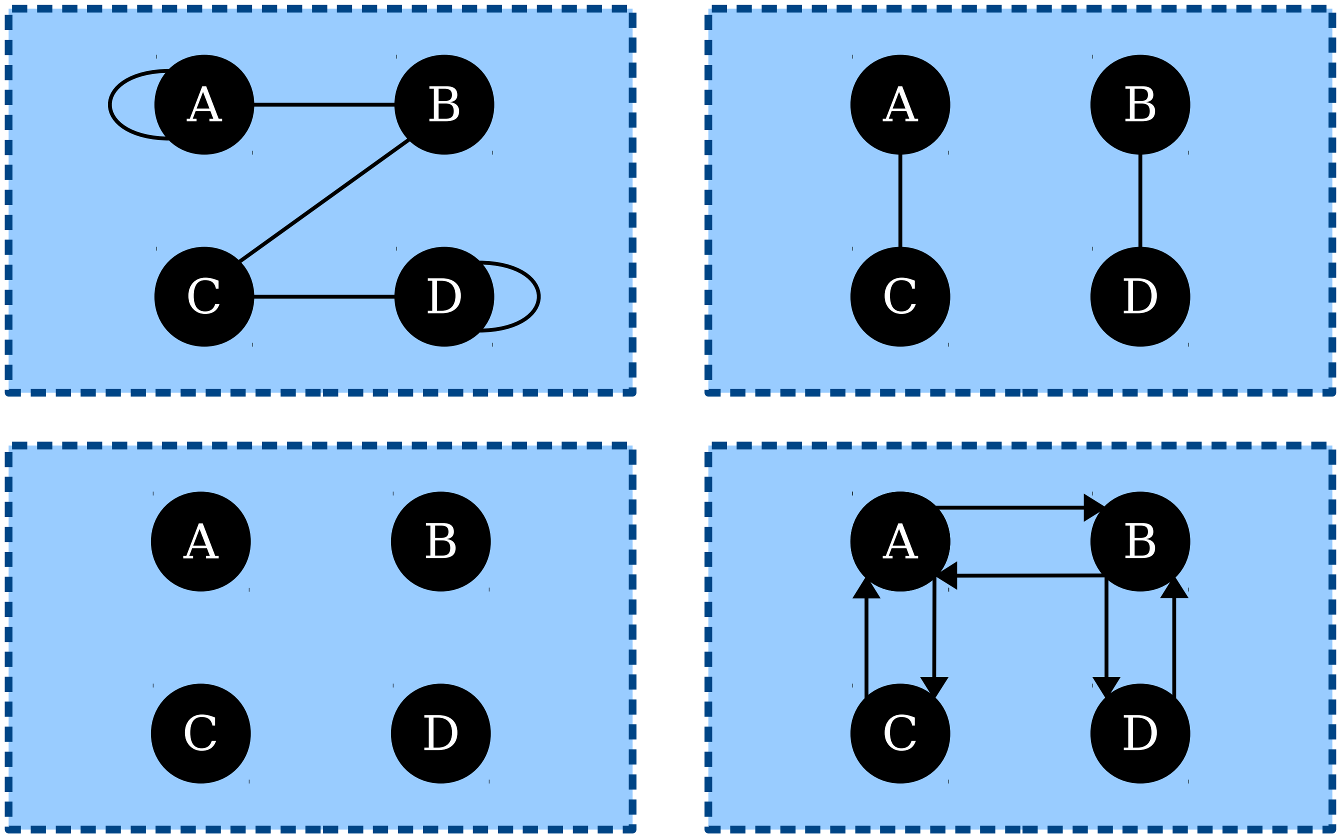
Formalizing Graphs

- How might we define a graph mathematically?
- We need to specify
  - what the nodes in the graph are, and
  - which edges are in the graph.
- The nodes can be pretty much anything.
- What about the edges?

Formalizing Graphs

- An ***unordered pair*** is a set $\{a, b\}$ of two elements $a \neq b$. (Remember that sets are unordered).
    - $\{0, 1\} = \{1, 0\}$
- An ***undirected graph*** is an ordered pair $G = (V, E)$, where
    - V is a set of nodes, which can be anything, and
    - E is a set of edges, which are unordered pairs of nodes drawn from V.
- A ***directed graph*** is an ordered pair $G = (V, E)$, where
    - V is a set of nodes, which can be anything, and
    - E is a set of edges, which are *ordered* pairs of nodes drawn from V.

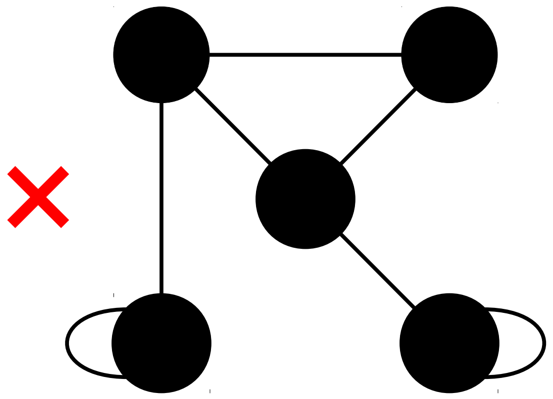
- An ***unordered pair*** is a set $\{a, b\}$ of two elements $a \neq b$.
- An ***undirected graph*** is an ordered pair $G = (V, E)$, where
  - V is a set of nodes, which can be anything, and
  - E is a set of edges, which are unordered pairs of nodes drawn from V.

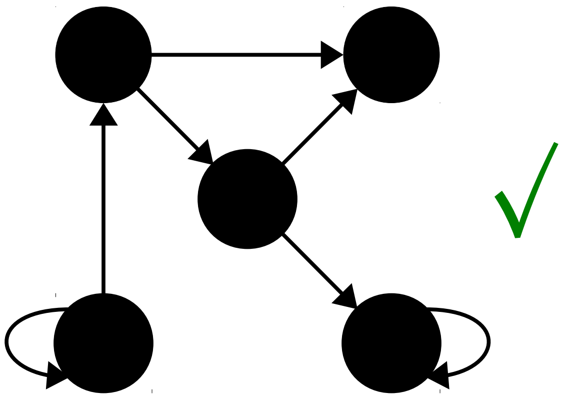


How many of these drawings are of valid undirected graphs?

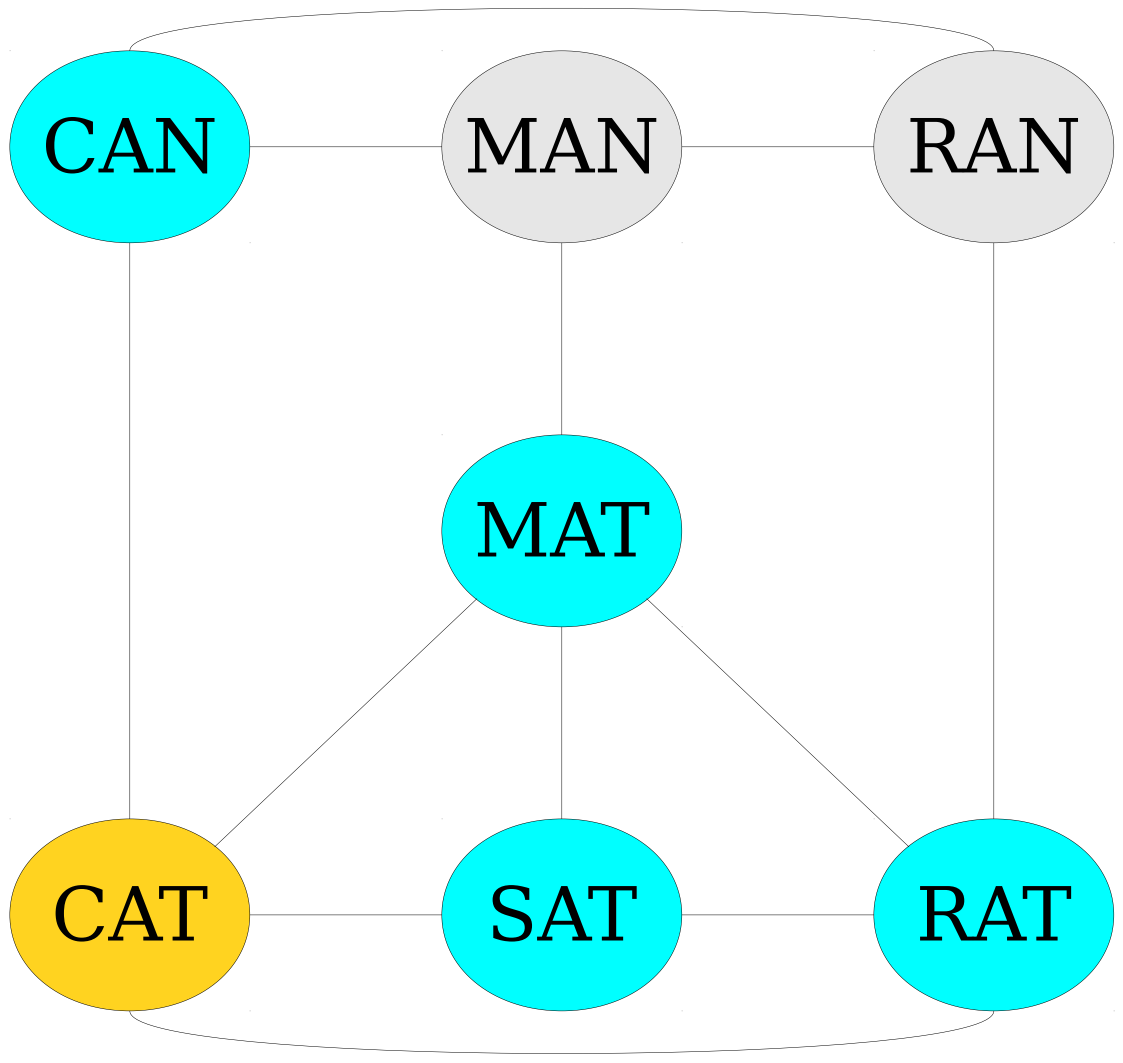
Self-Loops

- An edge from a node to itself is called a ***self-loop***.
- In undirected graphs, self-loops are generally not allowed.
  - Can you see how this follows from the definition?
- In directed graphs, self-loops are generally allowed unless specified otherwise.





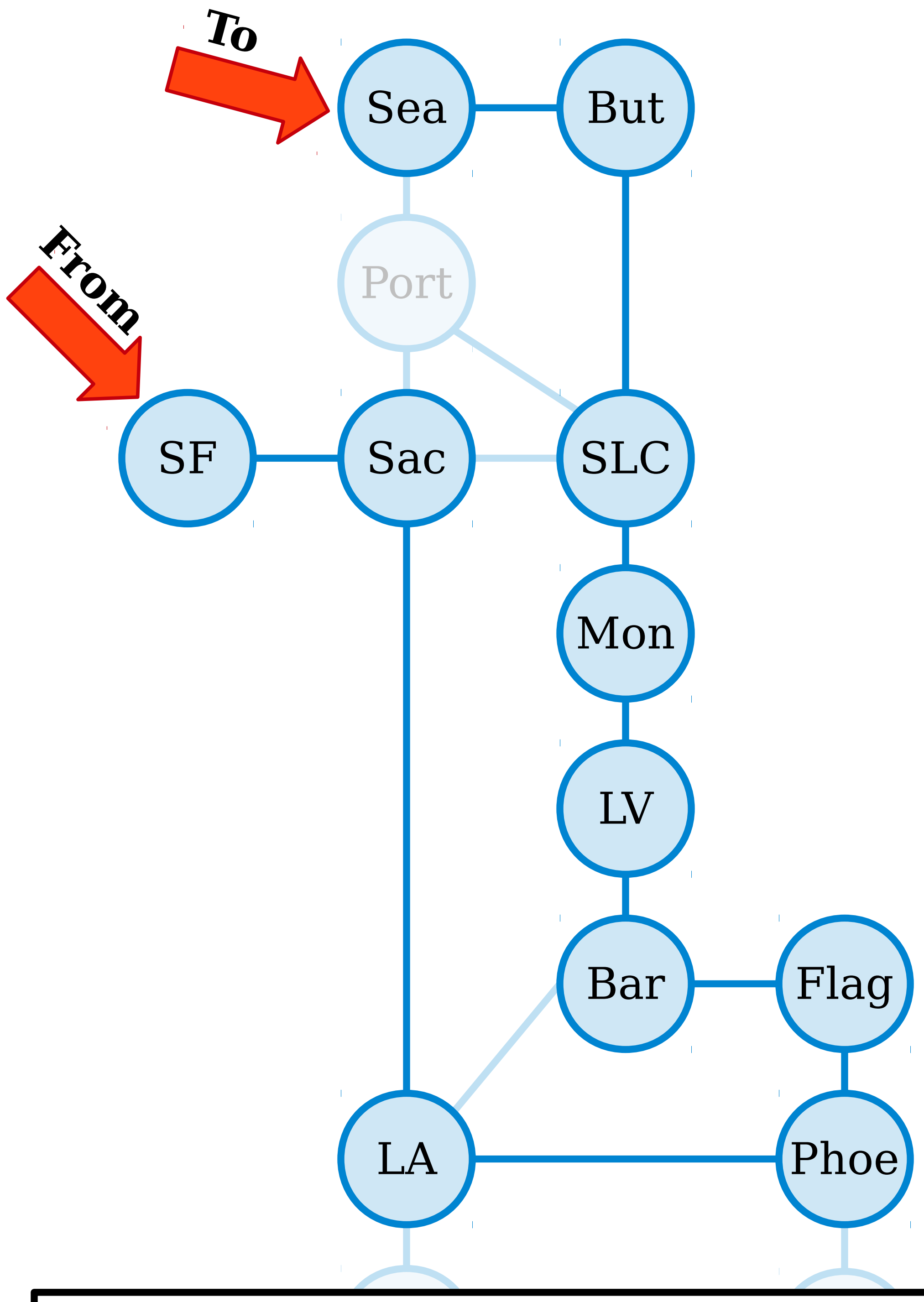
Standard Graph Terminology



Two nodes are called ***adjacent*** if there is an edge between them.

Using our Formalisms

- Let $G = (V, E)$ be a graph.
- Intuitively, two nodes are adjacent if they're linked by an edge.
- Formally speaking, we say that two nodes $u, v \in V$ are ***adjacent*** if $\{u, v\} \in E$.

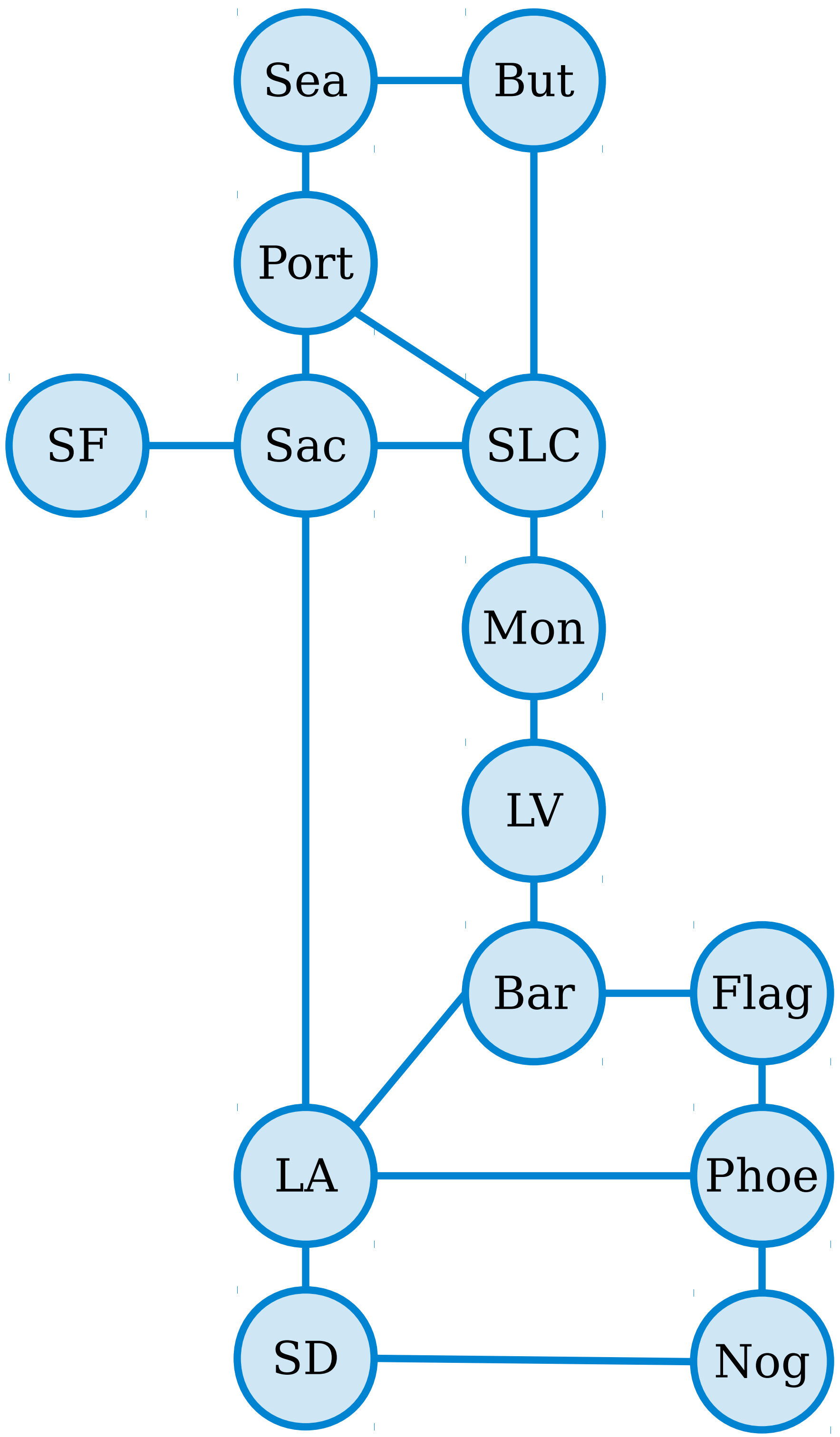


A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.

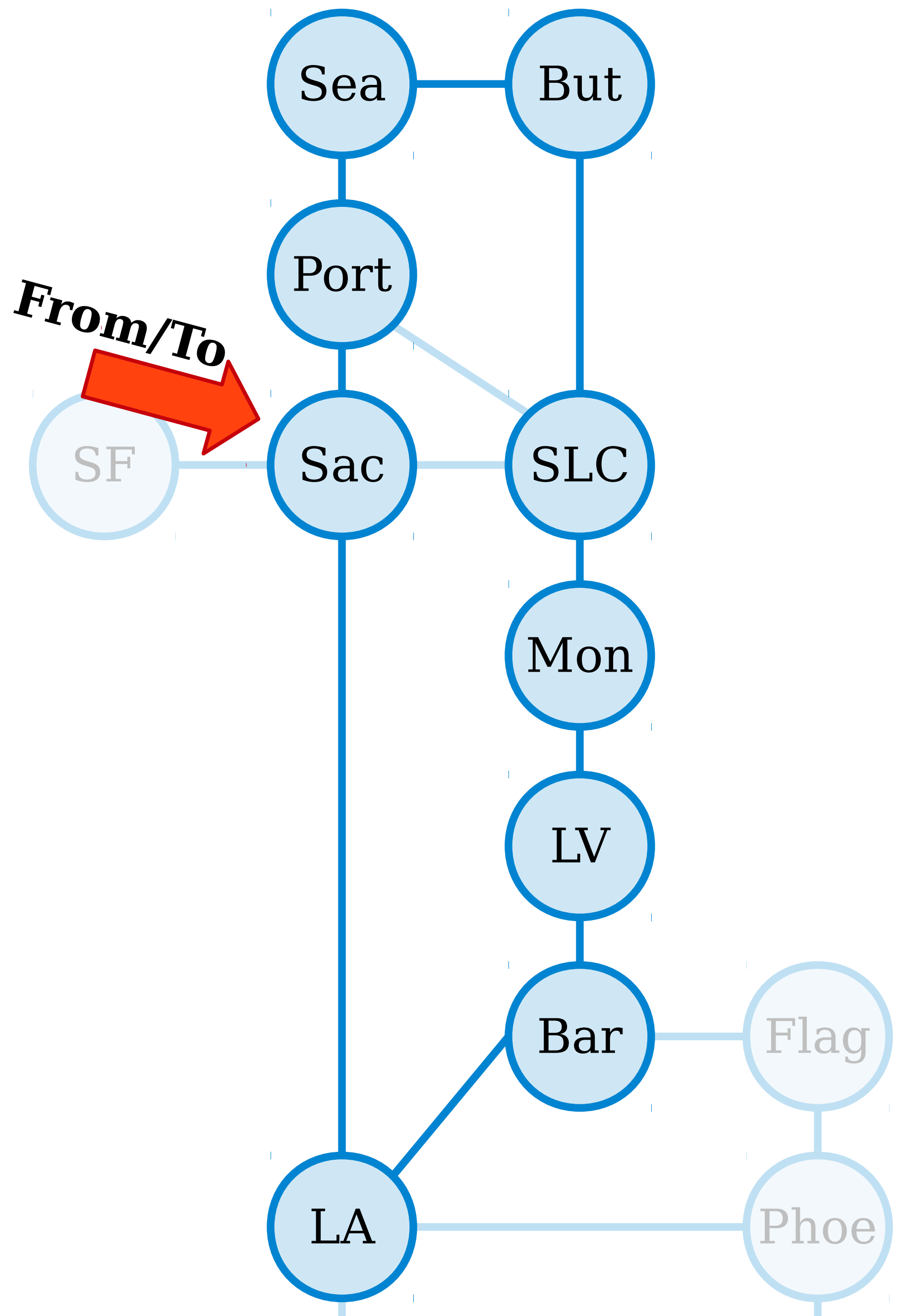
(This path has length 10, but visits 11 cities.)

SF, Sac, LA, Phoe, Flag, Bar, LV, Mon, SLC, But, Sea



A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.



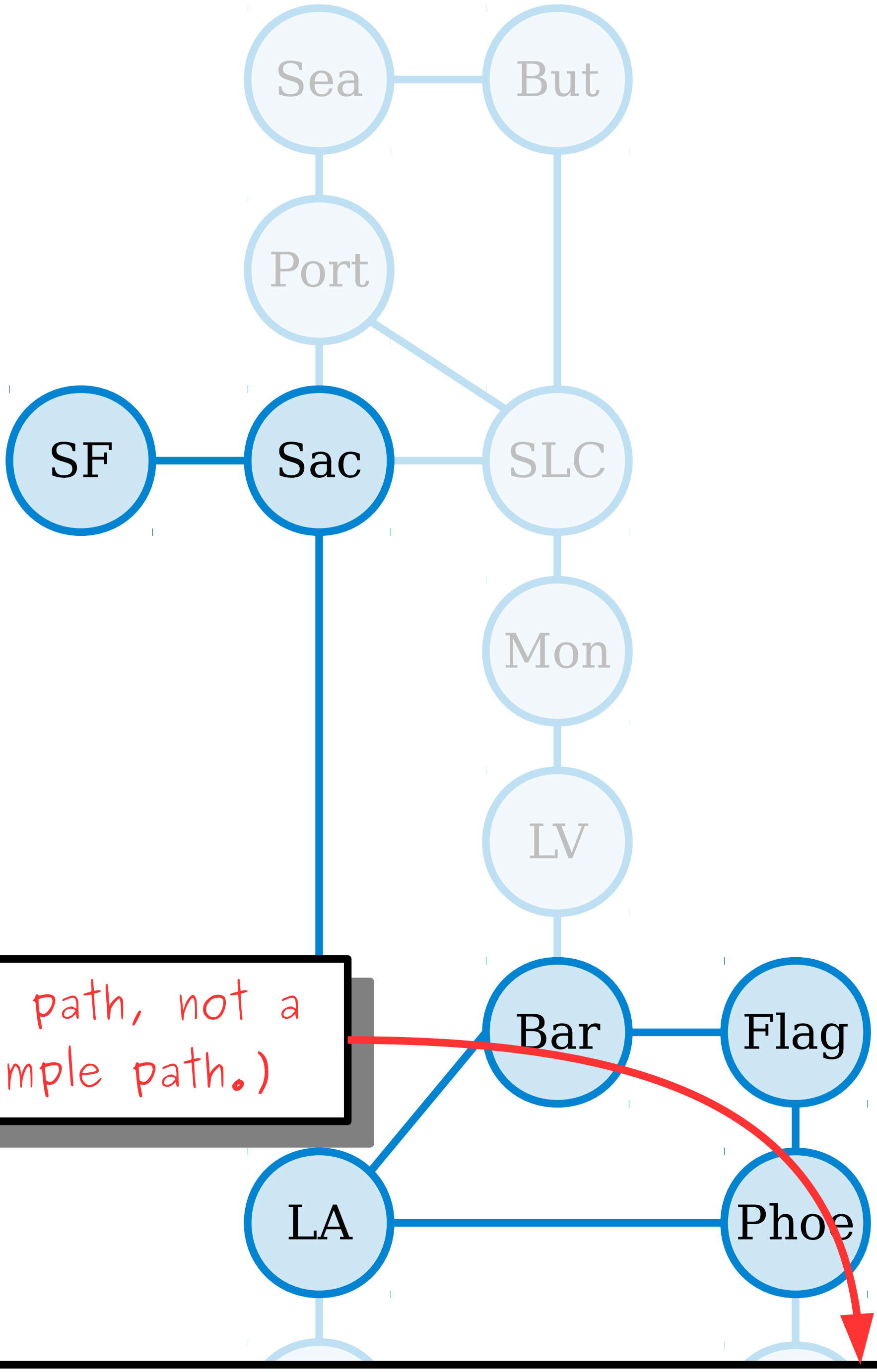
A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.

A ***cycle*** in a graph is a path from a node back to itself. (By convention, a cycle cannot have length zero.)

(This cycle has length nine and visits nine different cities.)

Sac, Port, Sea, But, SLC, Mon, LV, Bar, LA, Sac



A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

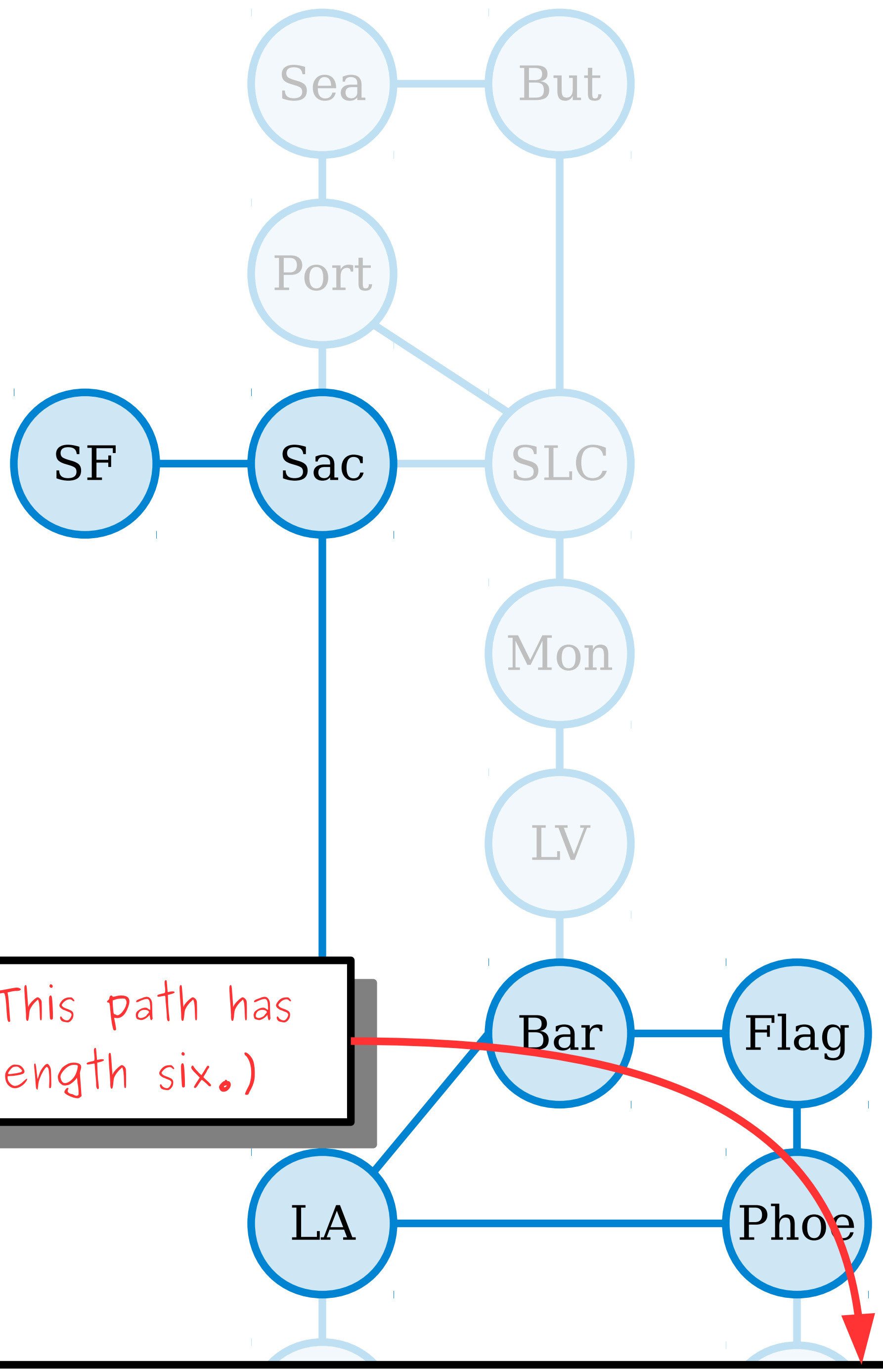
The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.

A ***cycle*** in a graph is a path from a node back to itself. (By convention, a cycle cannot have length zero.)

A ***simple path*** in a graph is path that does not repeat any nodes or edges.

(A path, not a simple path.)

SF, Sac, LA, Phoe, Flag, Bar, LA



(This path has length six.)

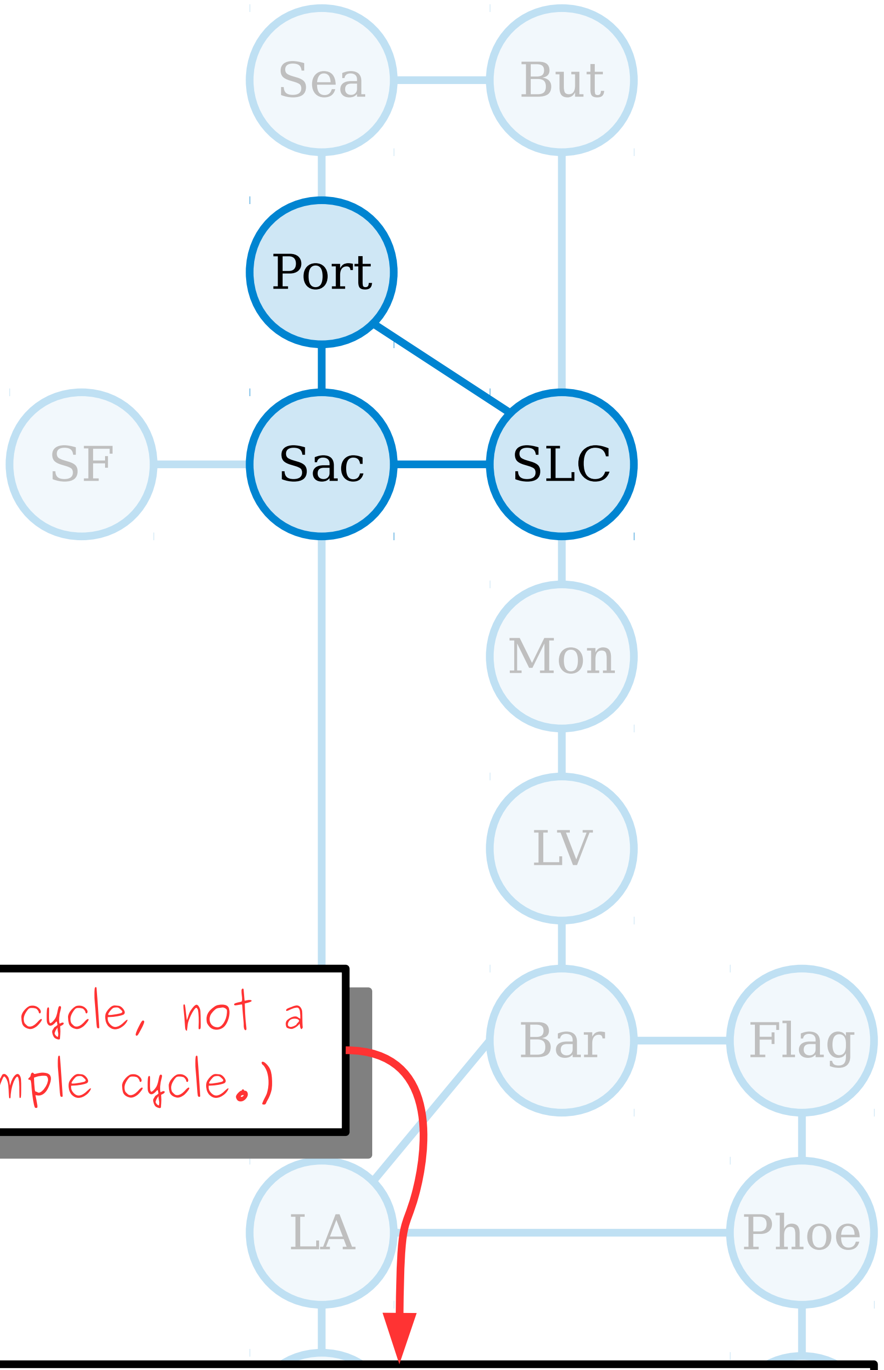
A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.

A ***cycle*** in a graph is a path from a node back to itself. (By convention, a cycle cannot have length zero.)

A ***simple path*** in a graph is path that does not repeat any nodes or edges.

SF, Sac, LA, Phoe, Flag, Bar, LA



(A cycle, not a simple cycle.)

Sac, SLC, Port, Sac, SLC, Port, Sac

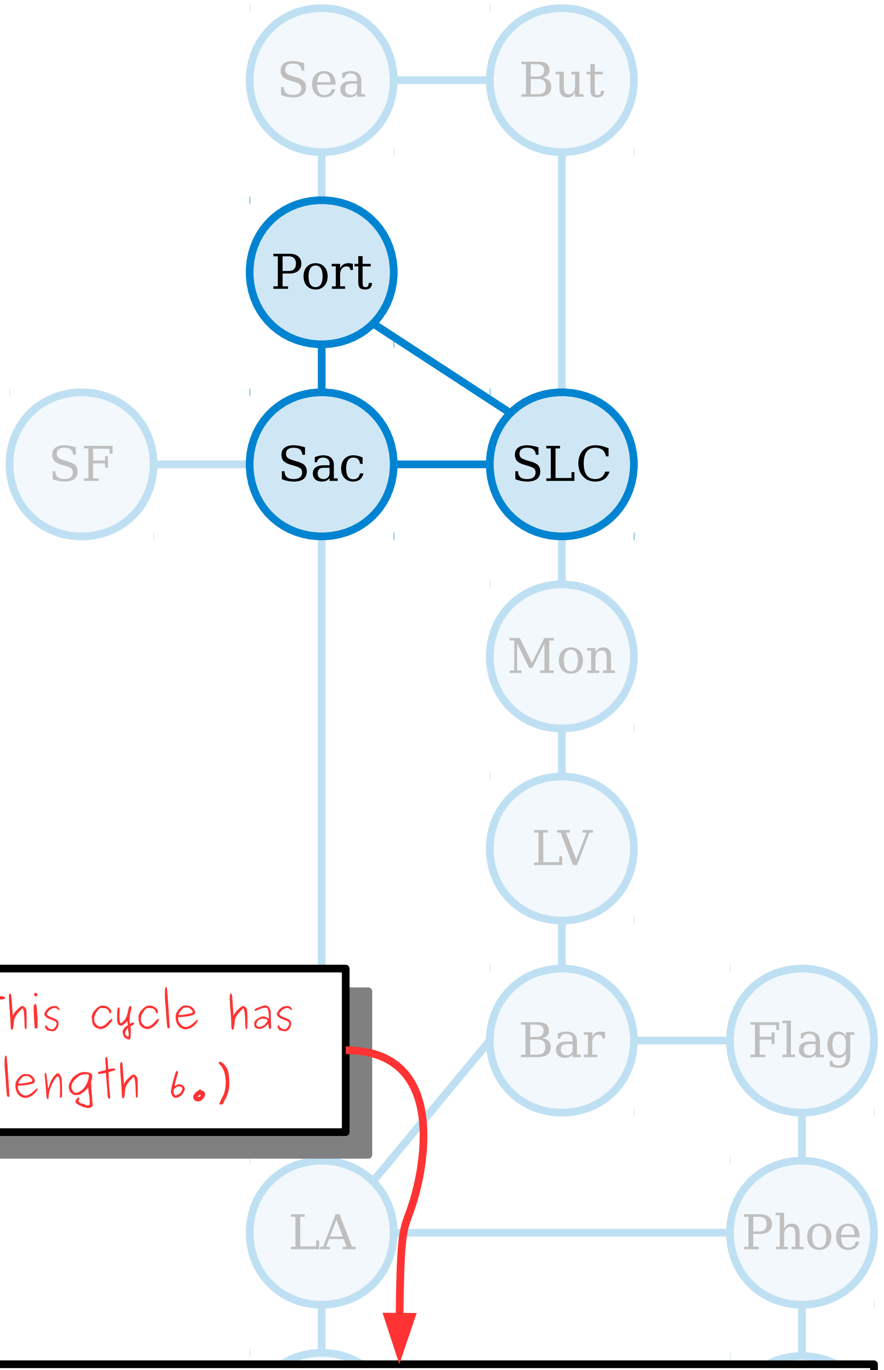
A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.

A ***cycle*** in a graph is a path from a node back to itself. (By convention, a cycle cannot have length zero.)

A ***simple path*** in a graph is path that does not repeat any nodes or edges.

A ***simple cycle*** in a graph is cycle that does not repeat any nodes or edges except the first/last node.



(This cycle has length 6.)

Sac, SLC, Port, Sac, SLC, Port, Sac

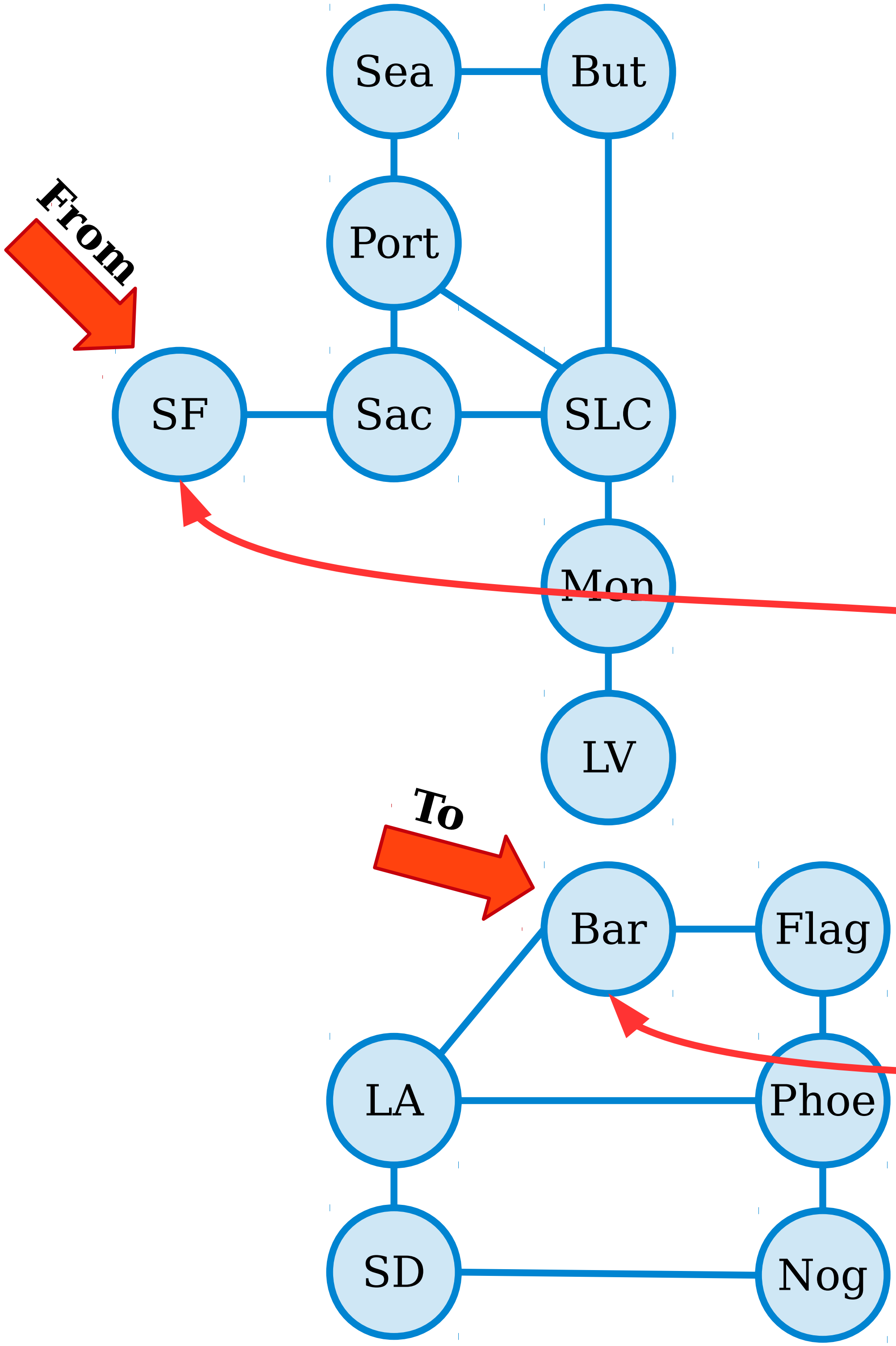
A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

The ***length*** of the path $v_1, \ldots, v_n$ is $n - 1$.

A ***cycle*** in a graph is a path from a node back to itself. (By convention, a cycle cannot have length zero.)

A ***simple path*** in a graph is path that does not repeat any nodes or edges.

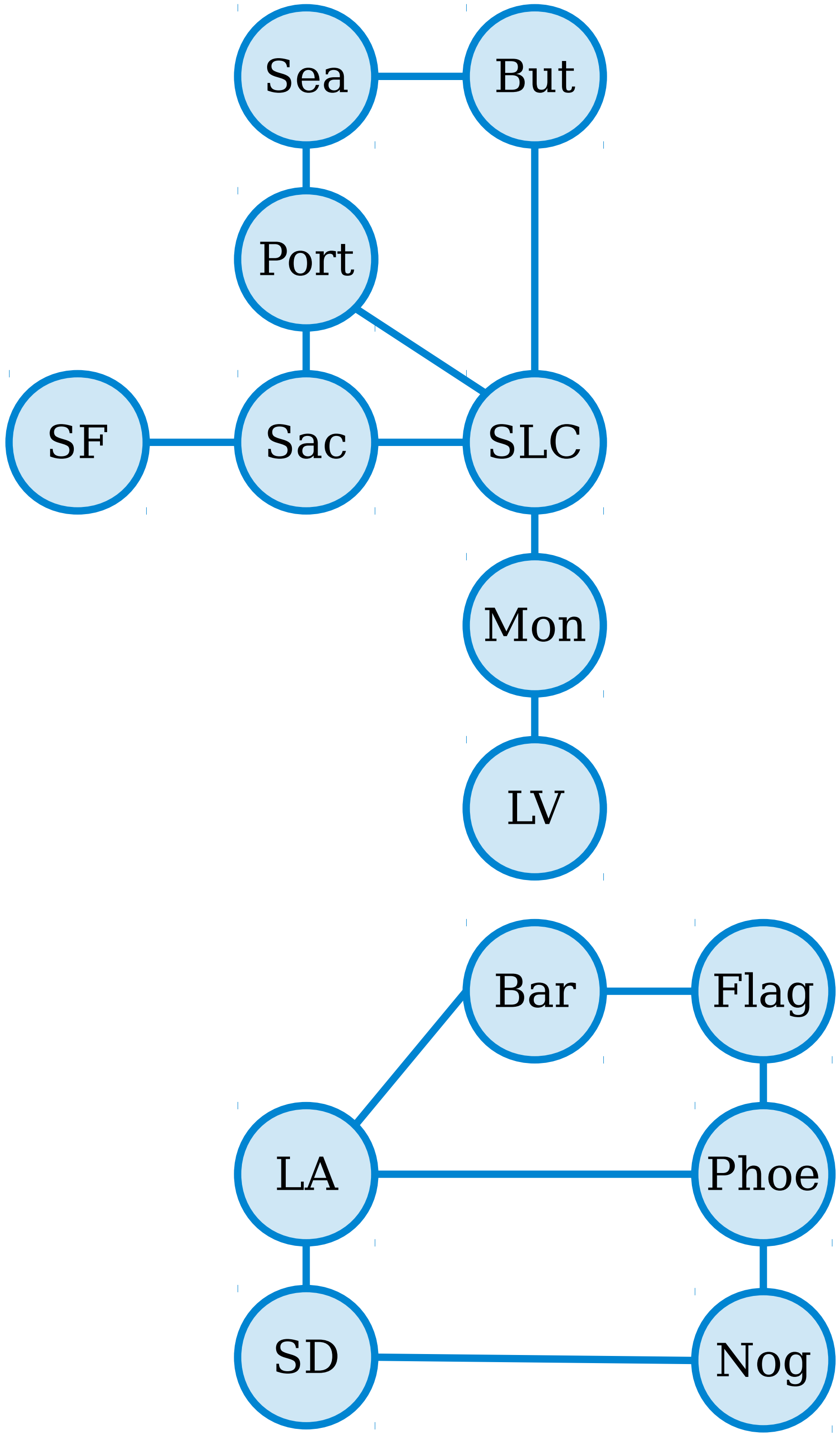
A ***simple cycle*** in a graph is cycle that does not repeat any nodes or edges except the first/last node.



A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

Two nodes in a graph are called ***connected*** if there is a path between them.

(These nodes are not connected. No Grand Canyon for you.)



A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

Two nodes in a graph are called ***connected*** if there is a path between them.

A graph G as a whole is called ***connected*** if all pairs of nodes in G are connected.

(This graph is not connected.)

Time-Out for Announcements!

Apply to Section Lead!

- Interested in section leading for the CS106 courses? Apply online using this link:

  https://cs198.stanford.edu/cs198/Apply.aspx
- Already completed CS106B/X or CS107? Be sure to apply by tomorrow evening.
- This is an incredible program – many SLs cite the program as the highlight of their time here at Stanford.

What?

Illustrated presentation on the fascinating Enigma machine story, and hands-on demonstration when the audience get to play with and photograph an original, iconic Enigma machine.

Who?

Talk by Dr Mark Baldwin (aka Dr Enigma), world expert on the Enigma machine and professional speaker with 20 years' experience and over 700 presentations around the world given at universities, conferences, clubs and companies

Where:

Y2E2, Room 111
Tuesday, October 23
1:30pm-2:50pm

https://drenigma.org

DR ENIGMA

Midterm Exam Logistics

- Our first midterm exam is next ***Monday, October 22nd***, from ***7:00PM – 10:00PM***. Locations are divvied up by last (family) name:
  - `A - L`: Go to Bishop Auditorium.
  - `M - Z`: Go to Cubberley Auditorium.
- You're responsible for Lectures 00 – 05 and topics covered in PS0 – PS2. Later lectures (relations forward) and problem sets (PS3 onward) won't be tested here.
- The exam is closed-book, closed-computer, and limited-note. You can bring a double-sided, 8.5" × 11" sheet of notes with you to the exam, decorated however you'd like.
- Students with OAE accommodations: you should have heard back from us with an alternate exam time. If you didn't hear back – or that alternate time doesn't work for you – please contact us as soon as possible.

Practice Midterm Exam

- We will be holding a practice midterm exam ***tonight*** in 320-105 from 7PM – 10PM.
- ***You should go to this exam unless you have a hard conflict***. It's the most realistic practice you can get.
- Can't make it? No worries! We'll post the exam and solutions up on the course website.

More Practice

- Want more practice? We’ve posted online
  - Extra Practice Problems 1;
  - three practice midterms, with solutions; and
  - the CS103A materials for the past few weeks.
- We’ll also post the fourth and final practice midterm (the one we’re giving out tonight).

Problem Sets

- PS3 checkpoints have been graded and returned. Please take some time to review them.
- ***Great question to ponder:*** Why *isn't* this relation transitive?

$$\begin{tikzcd}
x \arrow[rr, shift left] \arrow[dr, shift left] & & y \arrow[ll, shift left] \arrow[dl, shift left] \\
& z \arrow[ul, shift left] \arrow[ur, shift left] &
\end{tikzcd}$$

- You should hear back with feedback on PS2 later this evening.

Your Questions

“How will the math we learn in this class make us stronger computer scientists/be applicable in the real world beyond research?”

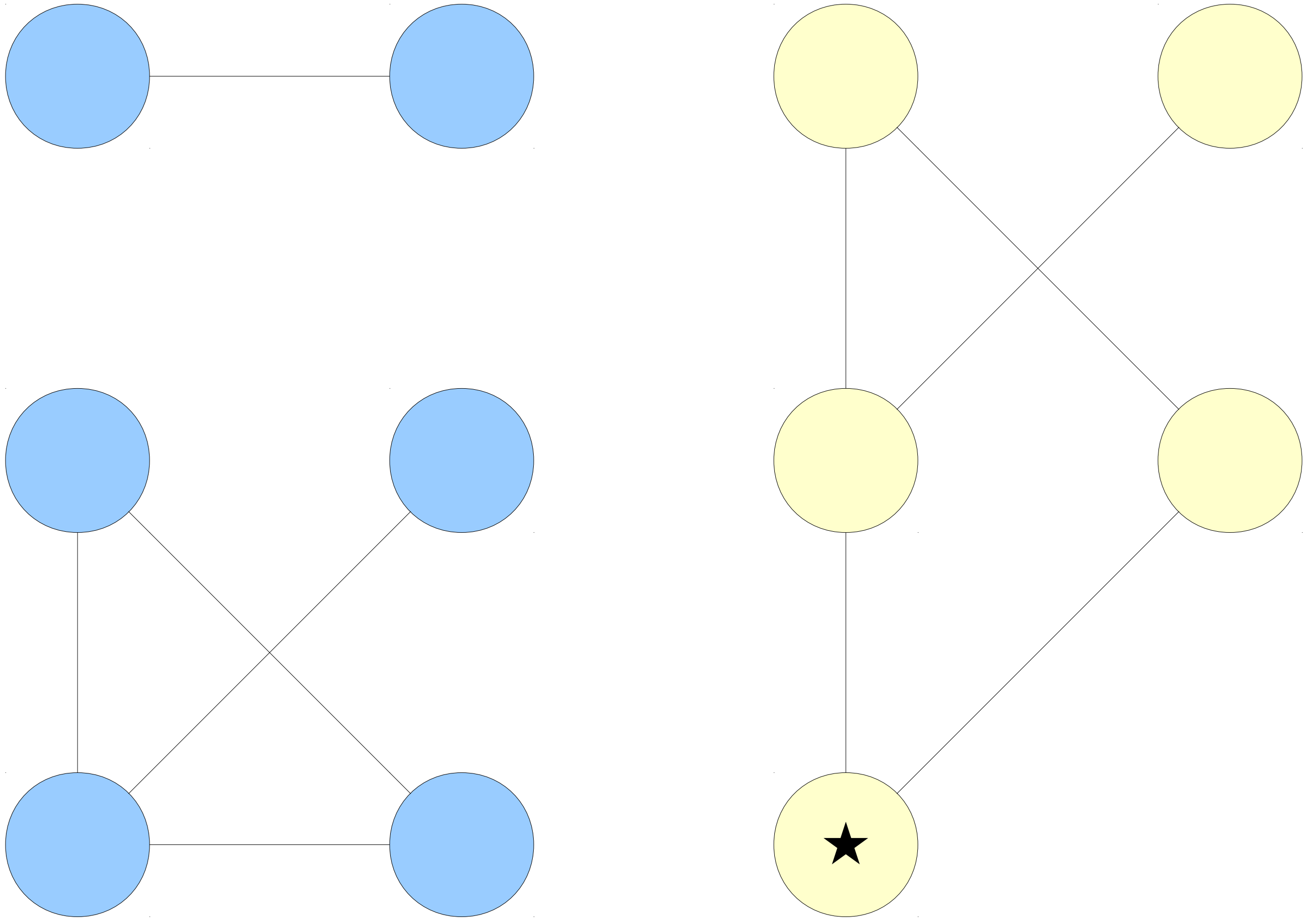
Math gives you an incredible set of tools for describing and reasoning about systems in a way that makes them easier to understand and, well, better! Allow me relate a few stories!

“You motivate us to keep trying, but you seem to have had a strong background coming to Stanford. Can you share one of your challenging moments at Stanford?”

I actually learned how to write proofs and to do math by taking CS103. I hadn't done anything like this before coming here, and it was a ton of work! And I felt totally outclassed by my classmates. But I put in a lot of effort, made tons of careless mistakes, learned all sorts of ways to mess up a proof, and eventually ended up getting a decent handle on things.

Back to CS103!

Connected Components



Connected Components

- Let $G = (V, E)$ be a graph. For each $v \in V$, the ***connected component*** containing v is the set

$$[v] = \{ x \in V \mid v \text{ is connected to } x \}$$

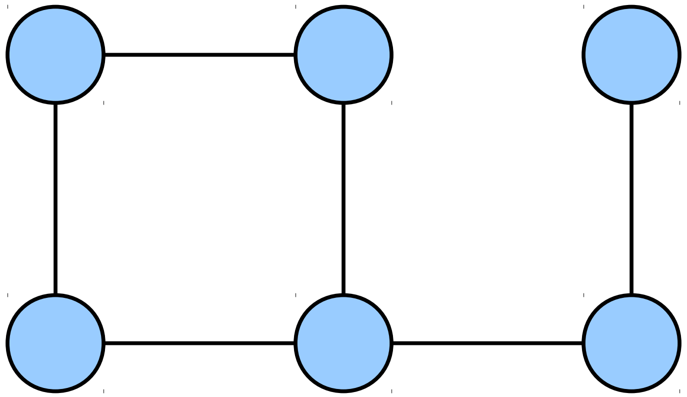
- Intuitively, a connected component is a "piece" of a graph in the sense we just talked about.
- ***Question:*** How do we know that this particular definition of a "piece" of a graph is a good one?
- ***Goal:*** Prove that any graph can be broken apart into different connected components.

We’re trying to reason about some way of partitioning the nodes in a graph into different groups.

What structure have we studied that captures the idea of a partition?

Connectivity

- ***Claim:*** For any graph G, the "is connected to" relation is an equivalence relation.
  - Is it reflexive?
  - Is it symmetric?
  - Is it transitive?



Connectivity

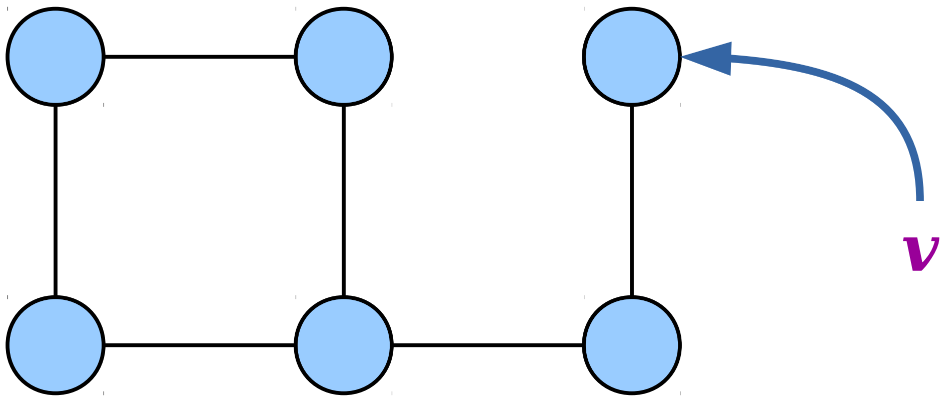
Claim: For any graph G, the "is connected to" relation is an equivalence relation.

- Is it reflexive?

  Is it symmetric?

  Is it transitive?

$$\forall v \in V.\ \mathit{Conn}(v, v)$$



A ***path*** in a graph $G = (V, E)$ is a sequence of one or more nodes $v_1, v_2, v_3, \ldots, v_n$ such that any two consecutive nodes in the sequence are adjacent.

Connectivity

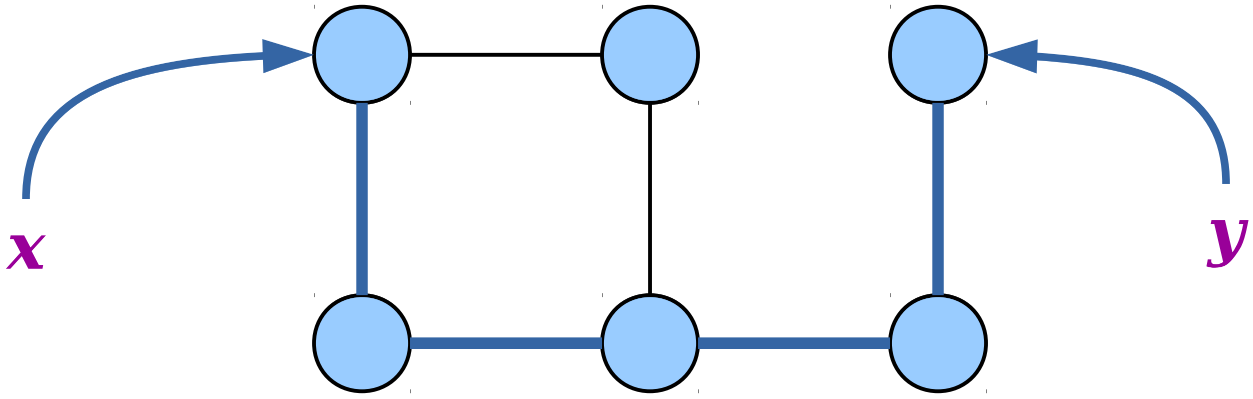
Claim: E
connecte
relation.

$\forall x \in V.\ \forall y \in V.\ (Conn(x, y) \to Conn(y, x))$

Is it reflexive?

- Is it symmetric?

Is it transitive?



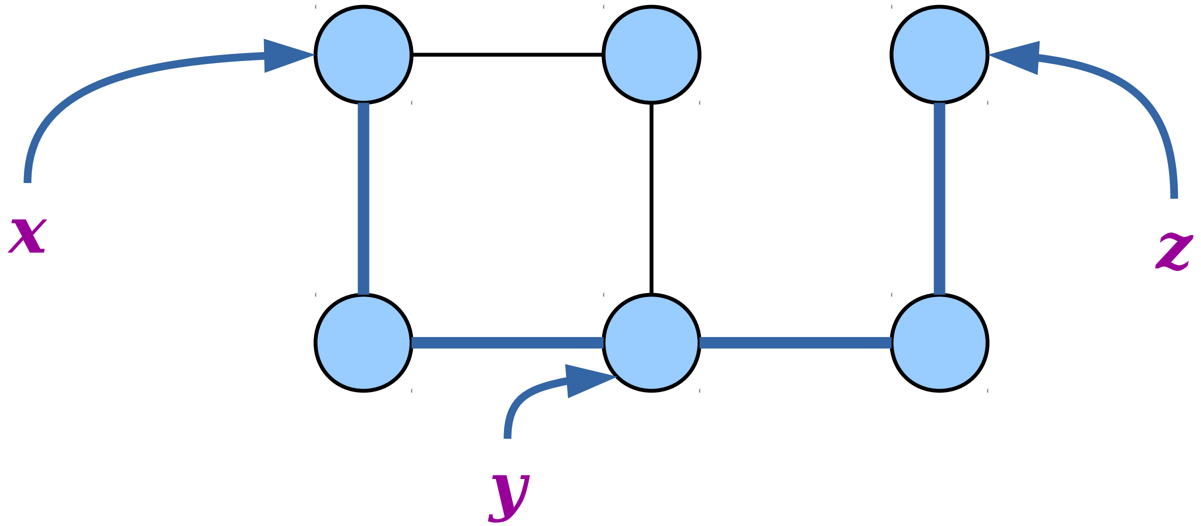
Connectivity

$$\forall x \in V.\ \forall y \in V.\ \forall z \in V.\ (Conn(x, y) \land Conn(y, z) \to Conn(x, z))$$

Is it reflexive?

Is it symmetric?

- Is it transitive?



Theorem: Let $G = (V, E)$ be a graph. Then the connectivity relation over V is an equivalence relation.

Proof: Consider an arbitrary graph $G = (V, E)$. We will prove that the connectivity relation over V is reflexive, symmetric, and transitive.

To show that connectivity is reflexive, consider any $v \in V$. Then the singleton path v is a path from v to itself. Therefore, v is connected to itself, as required.

To show that connectivity is symmetric, consider any $x, y \in V$ where x is connected to y. We need to show that y is connected to x. Since x is connected to y, there is some path $x, v_1, \ldots, v_n, y$ from x to y. Then $y, v_n, \ldots, v_1, x$ is a path from y back to x, so y is connected to x.

Finally, to show that connectivity is transitive, let $x, y, z \in V$ be arbitrary nodes where x is connected to y and y is connected to z. We will prove that x is connected to z. Since x is connected to y, there is a path $x, u_1, \ldots, u_n, y$ from x to y. Since y is connected to z, there is a path $y, v_1, \ldots, v_k, z$ from y to z. Then the path $x, u_1, \ldots, u_n, y, v_1, \ldots, v_k, z$ goes from x to z. Thus x is connected to z, as required. ■

Putting Things Together

- Earlier, we defined the connected component of a node *v* to be

$$[v] = \{ \; x \in V \mid v \text{ is connected to } x \; \}$$

- Connectivity is an equivalence relation! So what's the equivalence class of a node *v* with respect to connectivity?

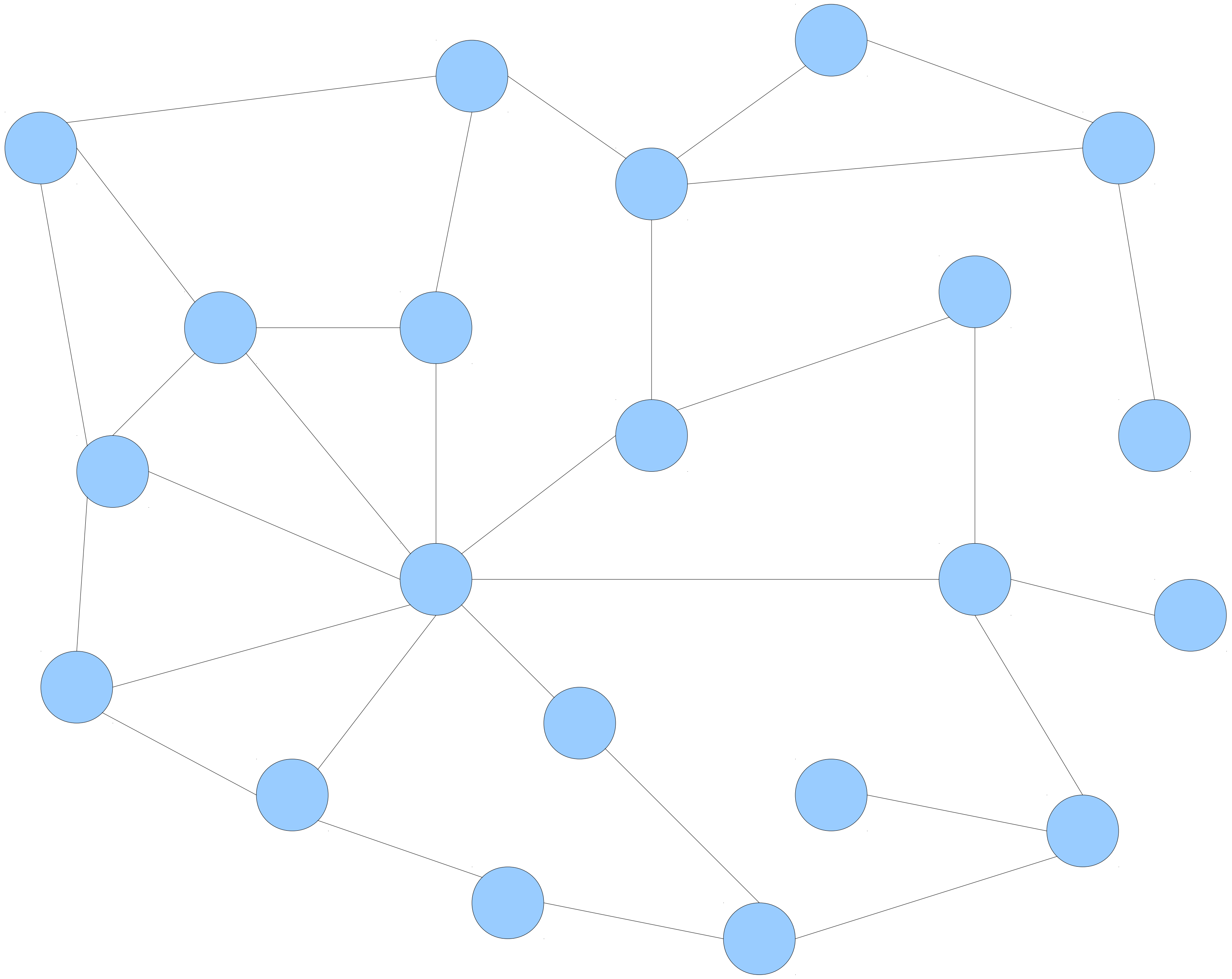
$$[v]_{conn} = \{ \; x \in V \mid v \text{ is connected to } x \; \}$$

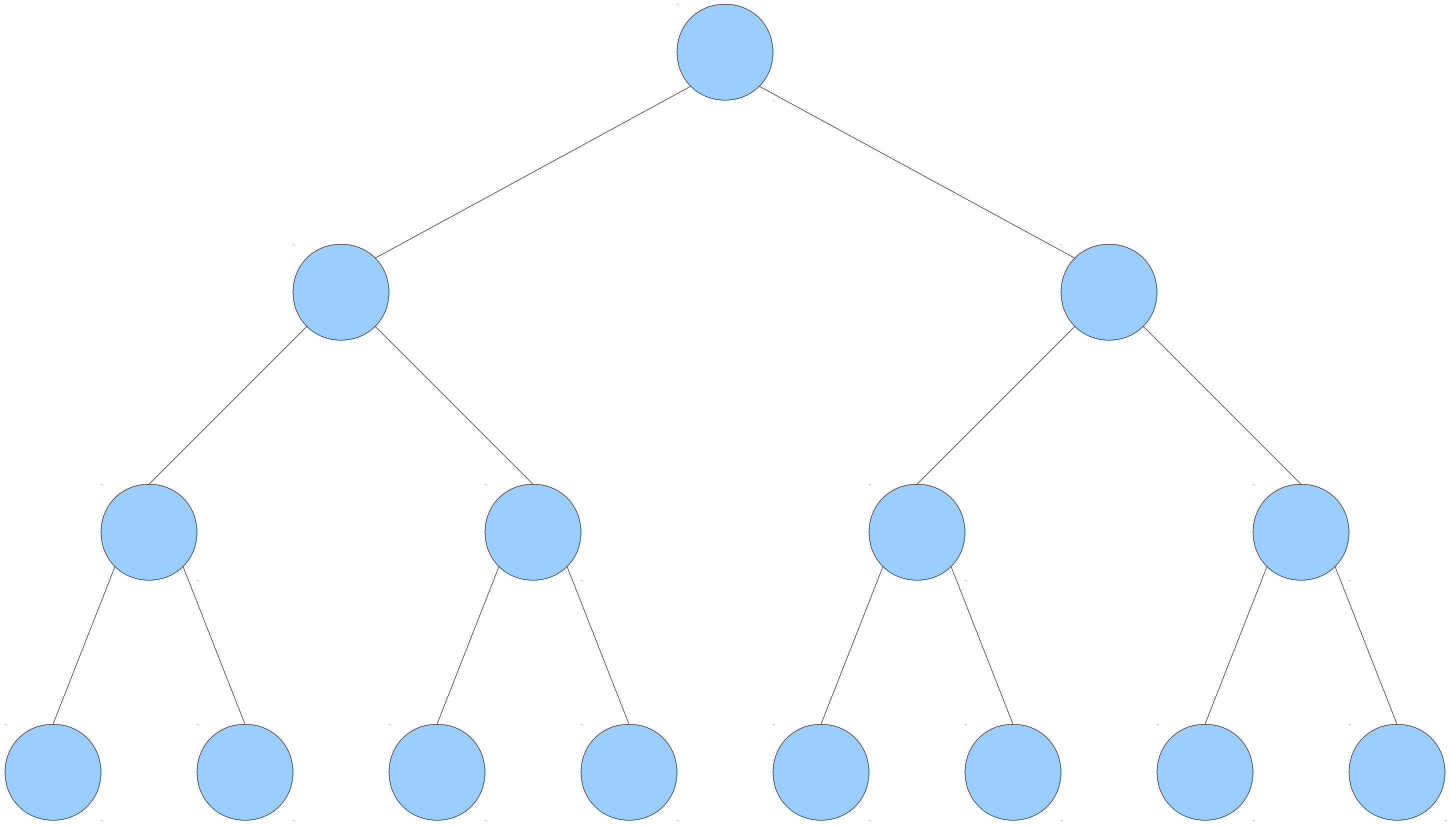
- ***Connected components are equivalence classes of the connectivity relation!***

Theorem: If $G = (V, E)$ is a graph, then every node in G belongs to exactly one connected component of G.

Proof: Let $G = (V, E)$ be an arbitrary graph and let $v \in V$ be any node in G. The connected components of G are just the equivalence classes of the connectivity relation in G. The Fundamental Theorem of Equivalence Relations guarantees that v belongs to exactly one equivalence class of the connectivity relation. Therefore, v belongs to exactly one connected component in G. ■

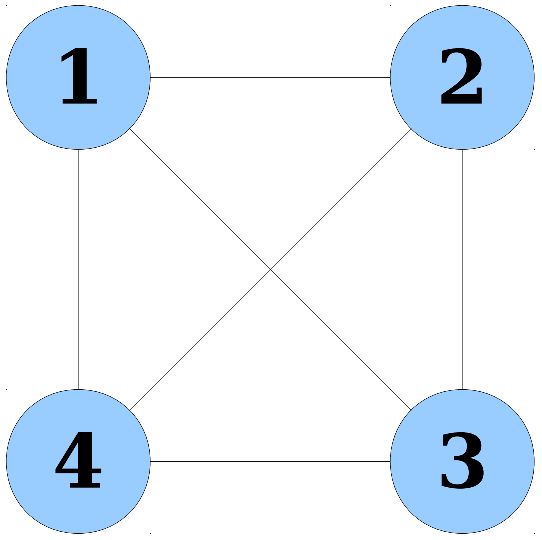
Planar Graphs



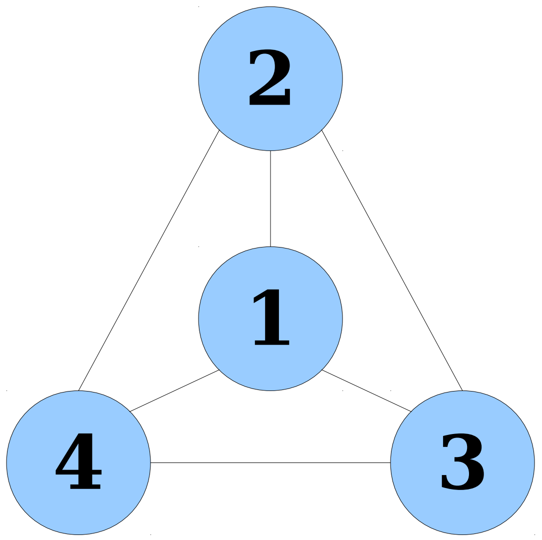


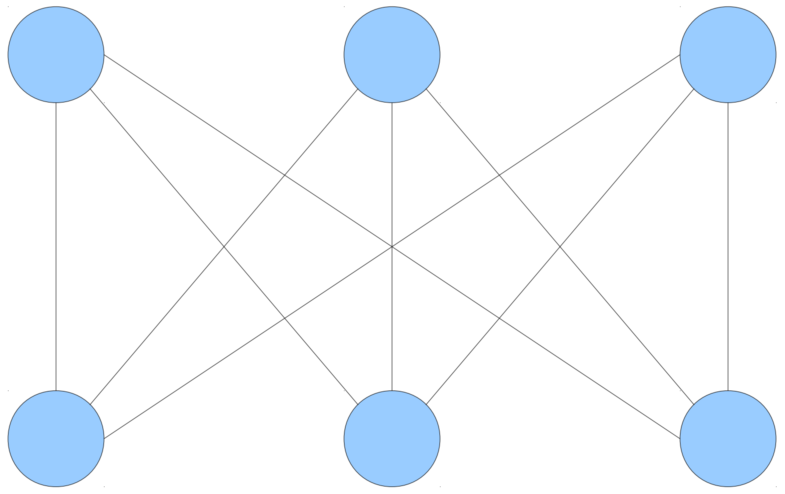
A graph is called a ***planar graph*** if there is some way to draw it in a 2D plane without any of the edges crossing.

A graph is called a ***planar graph*** if there is some way to draw it in a 2D plane without any of the edges crossing.



A graph is called a ***planar graph*** if there is some way to draw it in a 2D plane without any of the edges crossing.

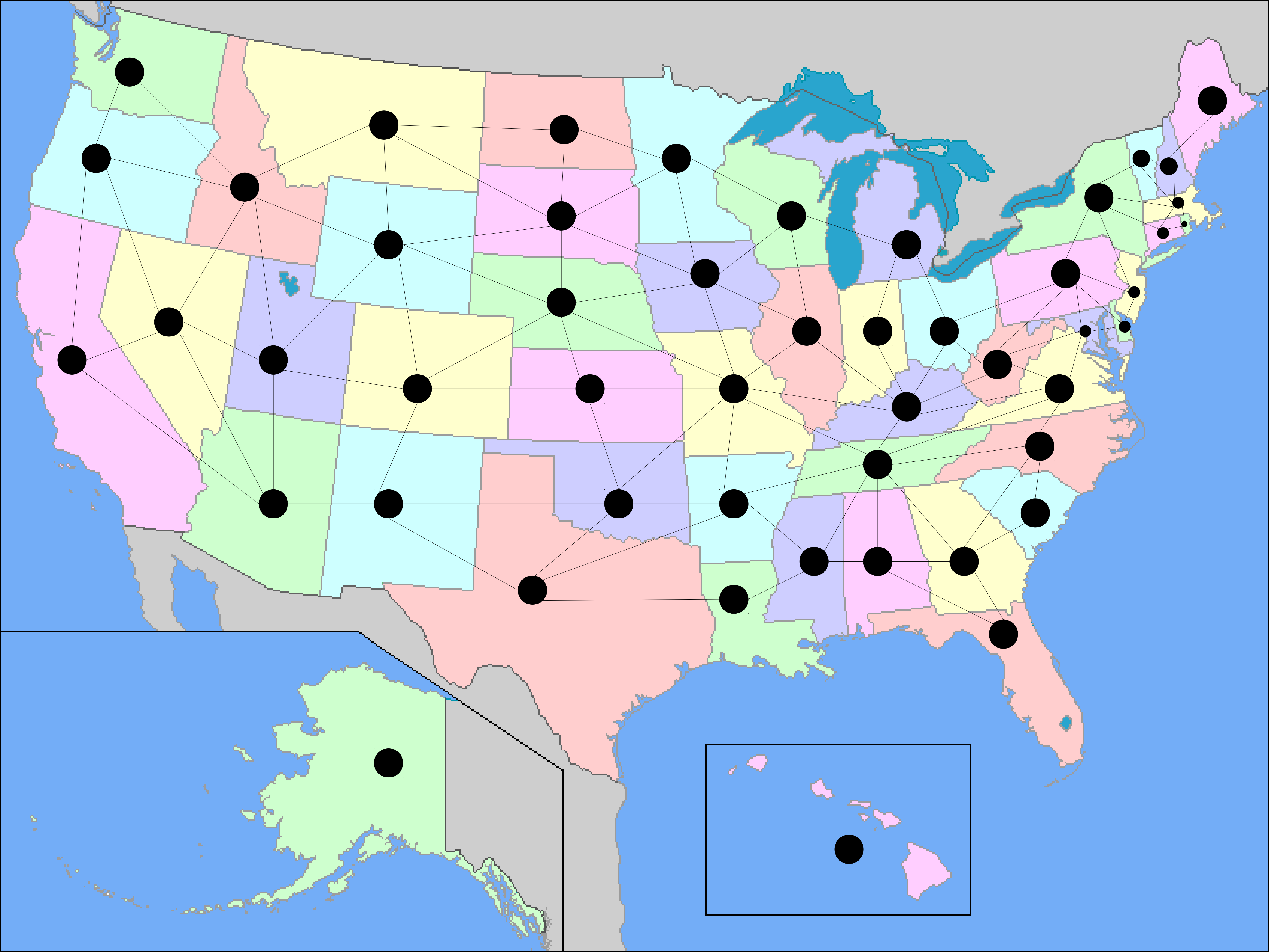


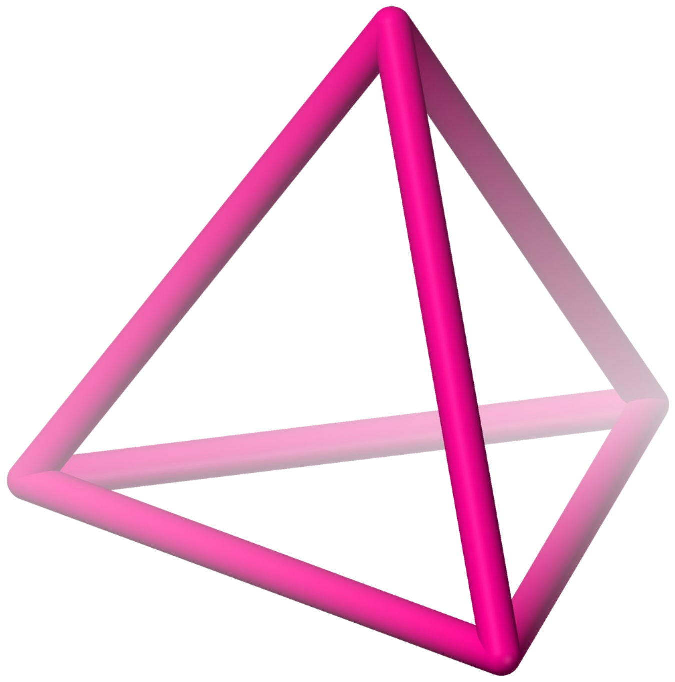


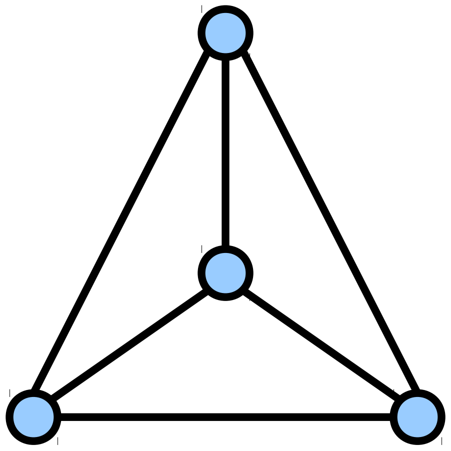
This graph is called the ***utility graph***. There is no way to draw it in the plane without edges crossing. Check out ***this video*** for an explanation!

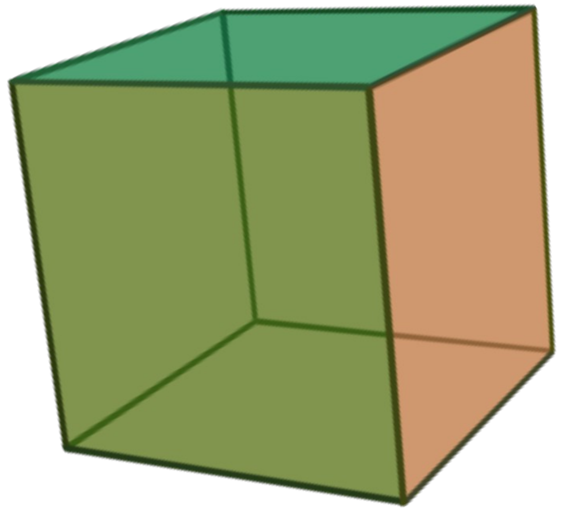
A fun game by a former CS103er:

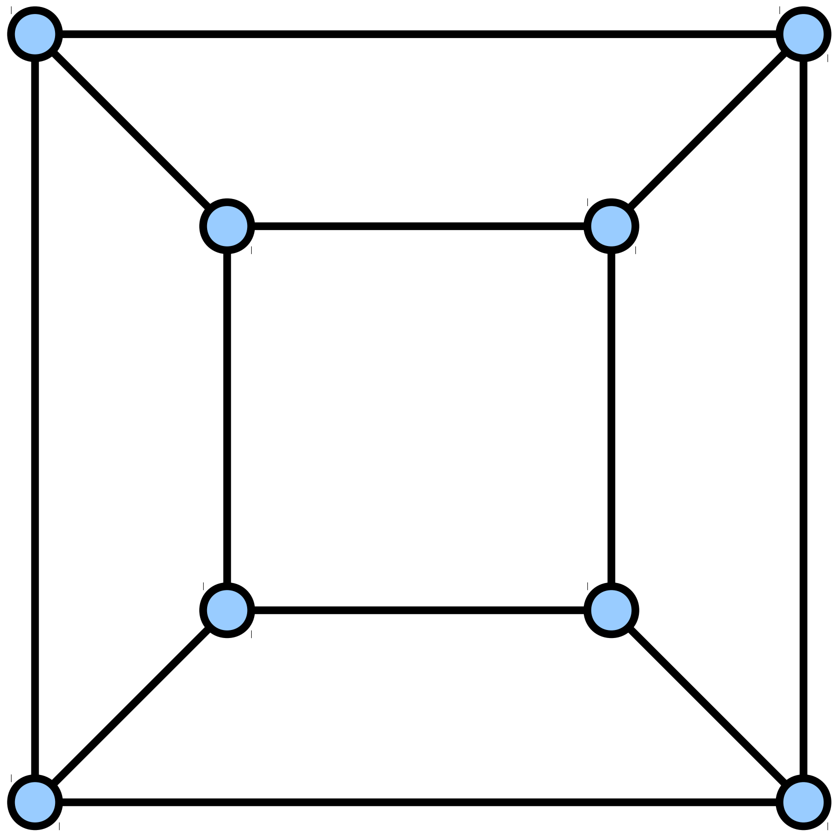
http://www.nkhem.com/planarity-knot/

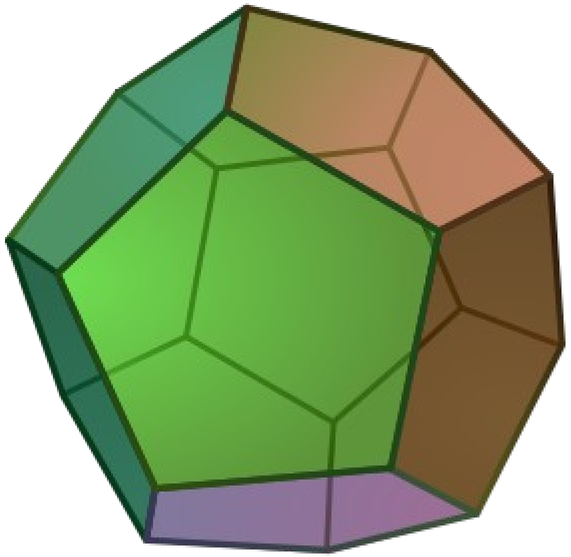
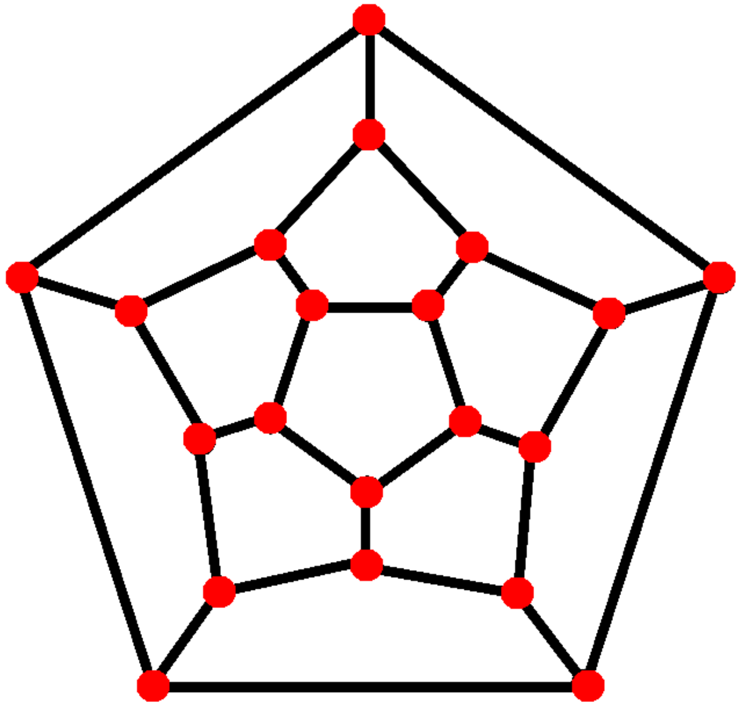










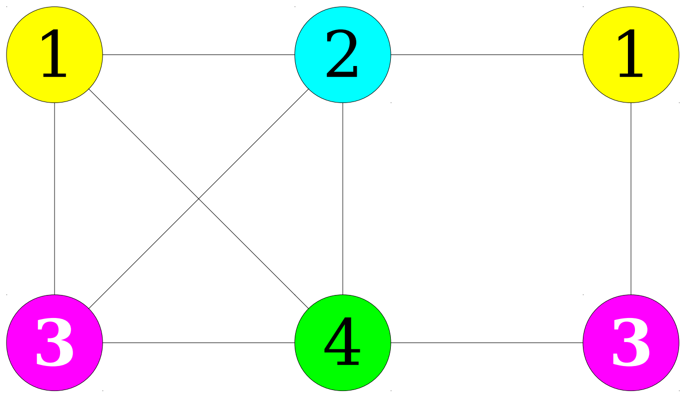
Graph Coloring

- Intuitively, a ***k-vertex-coloring*** of a graph $G = (V, E)$ is a way to color each node in V one of k different colors such that no two adjacent nodes in V are the same color.
- A ***k-vertex-coloring*** of a graph $G = (V, E)$ is a function

$$f : V \to \{1, 2, \ldots, k\}$$

such that

$$\forall u \in V.\ \forall v \in V.\ (\{u, v\} \in E \to f(u) \neq f(v))$$



Graph Coloring

- Intuitively, a ***k*-vertex-coloring** of a graph $G = (V, E)$ is a way to color each node in V one of k different colors such that no two adjacent nodes in V are the same color.
- A ***k*-vertex-coloring** of a graph $G = (V, E)$ is a function

$$f : V \to \{1, 2, \ldots, k\}$$

such that

$$\forall u \in V.\ \forall v \in V.\ (\{u, v\} \in E \to f(u) \neq f(v))$$

Although this is the formal definition of a *k*-vertex-coloring, you rarely see it used in proofs. It's more common to just talk about assigning colors to nodes. However, this definition is super useful if you want to write programs to reason about graph colorings!

Graph Coloring

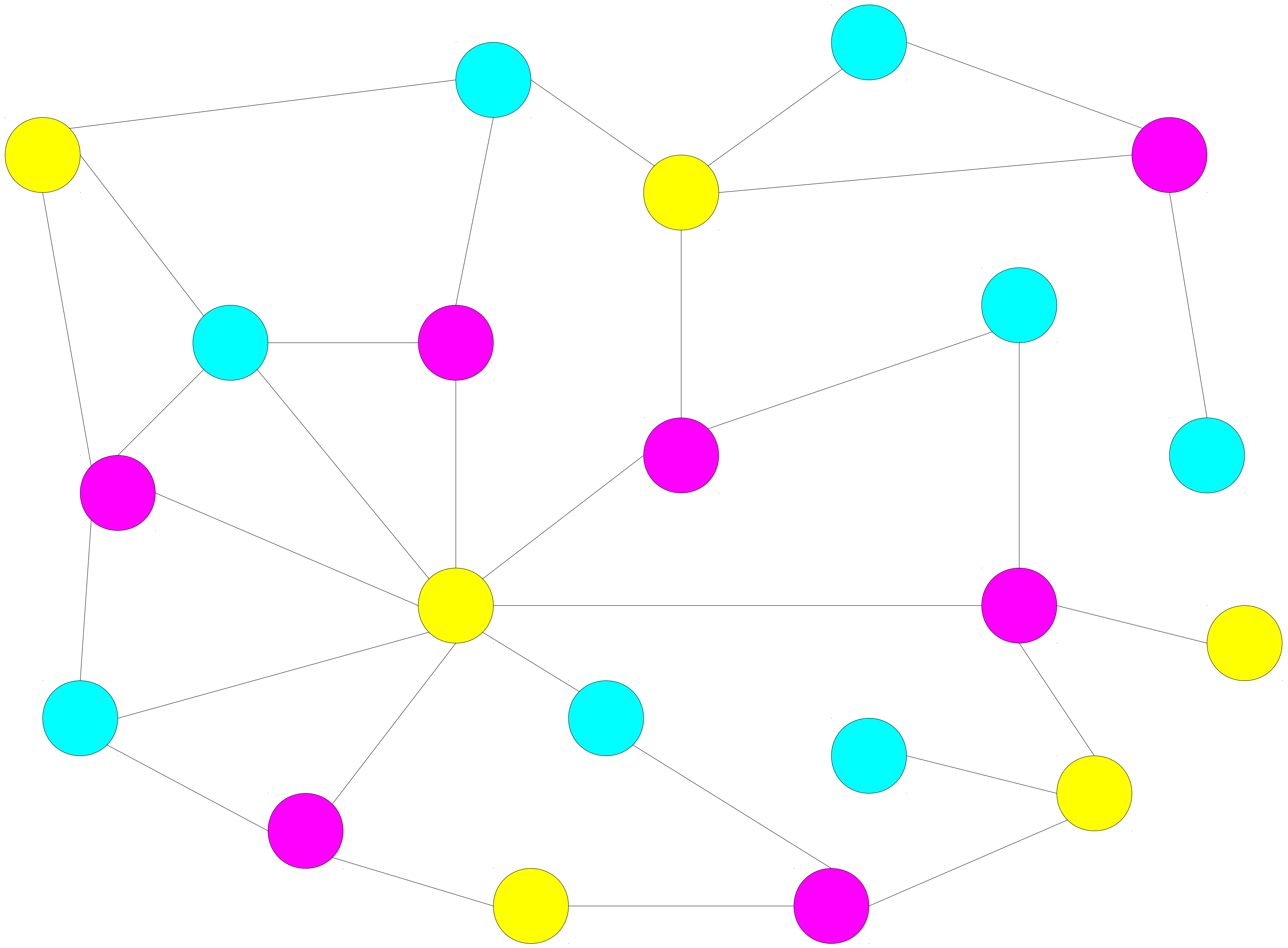
- Intuitively, a ***k-vertex-coloring*** of a graph $G = (V, E)$ is a way to color each node in V one of k different colors such that no two adjacent nodes in V are the same color.
- A ***k-vertex-coloring*** of a graph $G = (V, E)$ is a function

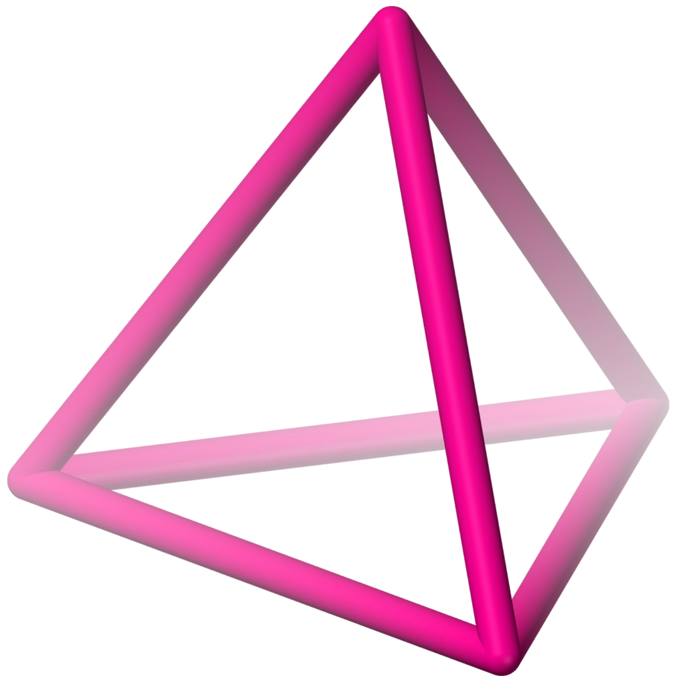
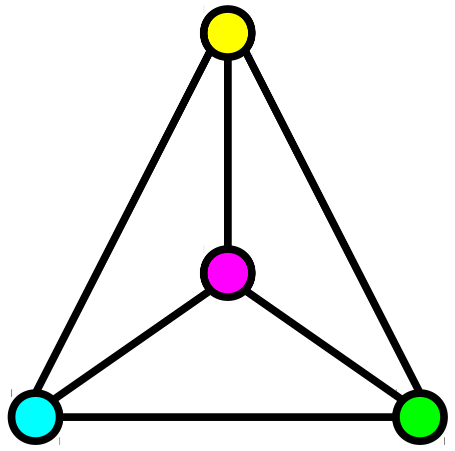
  $$f : V \to \{1, 2, \ldots, k\}$$

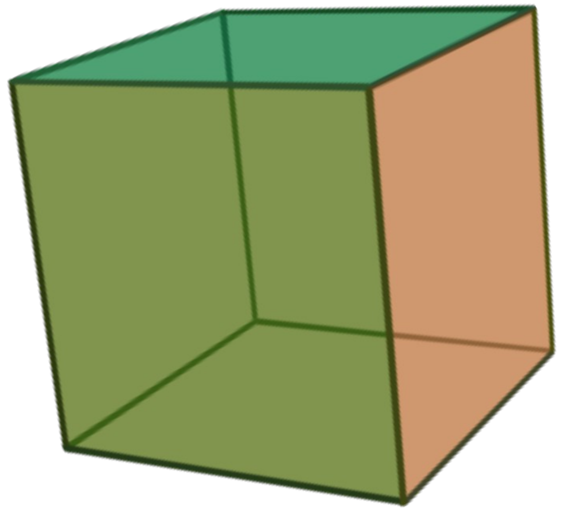
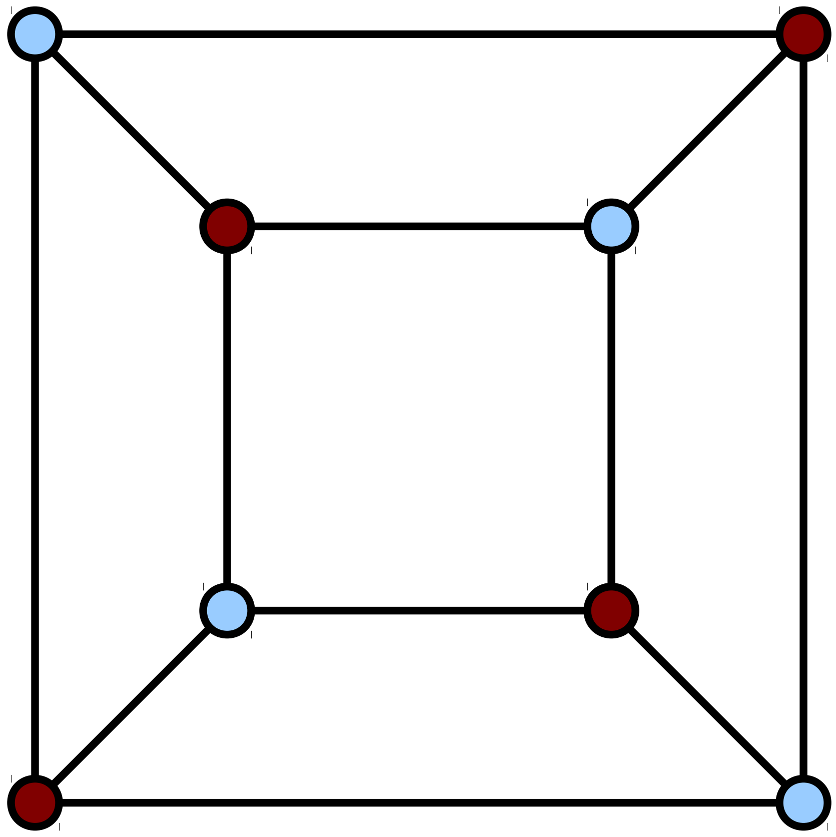
  such that

  $$\forall u \in V.\ \forall v \in V.\ (\{u, v\} \in E \to f(u) \neq f(v))$$
- A graph G is ***k-colorable*** if a k-vertex coloring of G exists.
- The smallest k for which G is k-colorable is its ***chromatic number***.
  - The chromatic number of a graph G is denoted $\chi(G)$, from the Greek *χρώμα*, meaning “color.”

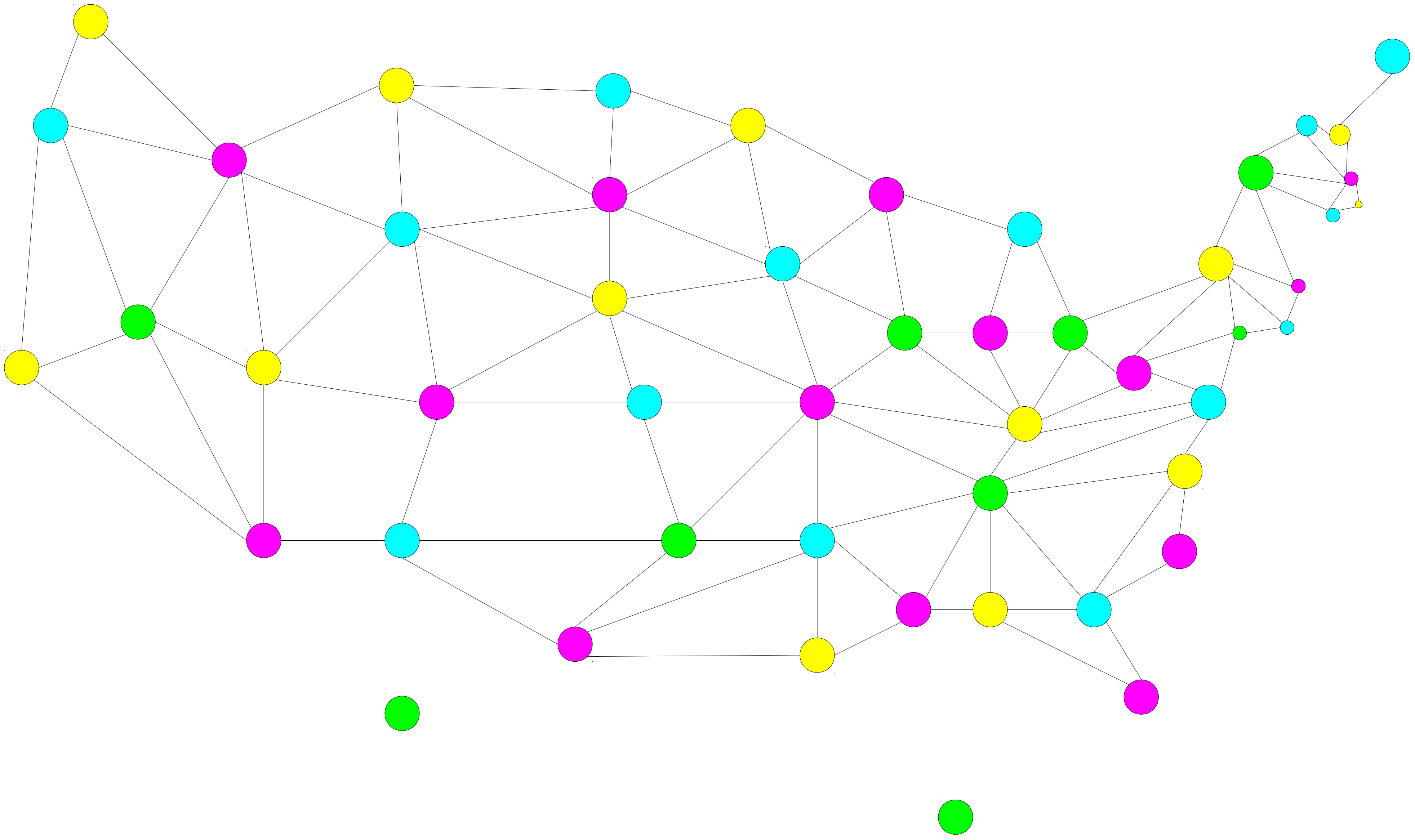
Graph Coloring



Graph Coloring



Theorem (Four-Color Theorem): Every planar graph is 4-colorable.

- **1850s:** Four-Color Conjecture posed.
- **1879:** Kempe proves the Four-Color Theorem.
- **1890:** Heawood finds a flaw in Kempe's proof.
- **1976:** Appel and Haken design a computer program that proves the Four-Color Theorem. The program checked 1,936 specific cases that are “minimal counterexamples;” any counterexample to the theorem must contain one of the 1,936 specific cases.
- **1980s:** Doubts rise about the validity of the proof due to errors in the software.
- **1989:** Appel and Haken revise their proof and show it is indeed correct. They publish a book including a 400-page appendix of all the cases to check.
- **1996:** Roberts, Sanders, Seymour, and Thomas reduce the number of cases to check down to 633.
- **2005:** Werner and Gonthier repeat the proof using an established automatic theorem prover (Coq), improving confidence in the truth of the theorem.

Philosophical Question: Is a theorem true if no human has ever read the proof?

A Fantastic Video on a Cool Theorem:

https://youtu.be/-9OUyo8NFZg

Next Time

- ***The Pigeonhole Principle***
  - A simple, powerful, versatile theorem.
- ***Graph Theory Party Tricks***
  - Applying math to graphs of people!