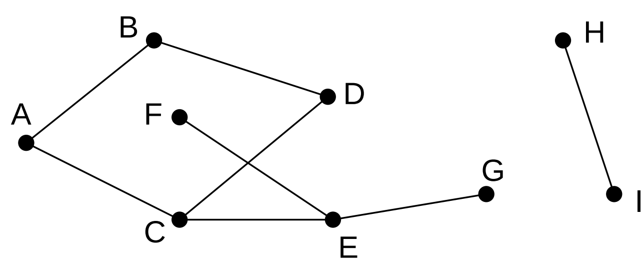
Graph Theory

[revised September 19, 2006, 1414 minutes]

1 Introduction

Informally, a ***graph*** is a bunch of dots connected by lines. Here is an example of a graph:



Sadly, this definition is not precise enough for mathematical discussion. Formally, a graph is a pair of sets (V, E), where:

- V is a set whose elements are called ***vertices***.
- E is a collection of two-element subsets of V called ***edges***.

The vertices correspond to the dots in the picture, and the edges correspond to the lines. Thus, the dots-and-lines diagram above is a pictorial representation of the graph (V, E) where:

$$V = \{A, B, C, D, E, F, G, H, I\}$$
$$E = \{\{A, B\}, \{A, C\}, \{B, D\}, \{C, D\}, \{C, E\}, \{E, F\}, \{E, G\}, \{H, I\}\}.$$

1.1 Definitions

A nuisance in first learning graph theory is that there are so many definitions. They all correspond to intuitive ideas, but can take a while to absorb. Some ideas have multiple names. For example, graphs are sometimes called ***networks***, vertices are sometimes called ***nodes***, and edges are sometimes called ***arcs***. Even worse, no one can agree on the exact meanings of terms. For example, in our definition of a graph, the set of vertices may be empty. But some authors require every graph to have at least one vertex. This is typical; everyone agrees more-or-less what each term means, but disagrees about weird special cases. So do not be alarmed if definitions here differ subtly from definitions you see elsewhere. Usually, these differences do not matter.

Hereafter, we use A—B to denote an edge between vertices A and B rather than the set notation $\{A, B\}$. Note that A—B and B—A are the same edge, just as $\{A, B\}$ and $\{B, A\}$ are the same set.

Two vertices in a graph are said to be ***adjacent*** if they are joined by an edge, and an edge is said to be ***incident*** to the vertices it joins. The number of edges incident to a vertex is called the ***degree*** of the vertex. For example, in the graph above, A is adjacent to B and B is adjacent to D, and the edge A—C is incident to vertices A and C. Vertex H has degree 1, D has degree 2, and E has degree 3.

Deleting some vertices and/or edges from a graph leaves a ***subgraph***. Formally, a subgraph of $G = (V, E)$ is a graph $G' = (V', E')$ where $V' \subseteq V$ and $E' \subseteq E$.

1.2 Sex in America

A 1994 University of Chicago study entitled *The Social Organization of Sexuality* found that on average men have 74% more opposite-gender partners than women.

Let's recast this obvservation in graph theoretic terms. Let $G = (V, E)$ be a graph where the set of vertices V consists of everyone in America. Now each vertex either represents either a man or a woman, so we can partition V into two subsets: M, which contains all the male vertices, and W, which contains all the female vertices. Let's draw all the M vertices on the left and the W vertices on the right.

Now, without getting into a lot of specifics, *sometimes an edge appears* between an M vertex and a W vertex:

Since we're only considering opposite-gender relationships, every edge connects an M vertex on the left to a W vertex on the right. (A graph, like this one, in which vertices can be partitioned into two types and edges join only vertices of different types is call ***bipartite***. Bipartite graphs arise in many contexts.) So the sum of the degrees of the M vertices must equal the sum of the degrees of the W vertices:

$$\sum_{x \in M} \deg(x) = \sum_{y \in W} \deg(y)$$

Now suppose we divide both sides of this equation by the product of the sizes of the two sets, $|M| \cdot |W|$:

$$\left(\frac{\sum_{x \in M} \deg(x)}{|M|} \right) \cdot \frac{1}{|W|} = \left(\frac{\sum_{y \in W} \deg(y)}{|W|} \right) \cdot \frac{1}{|M|}$$

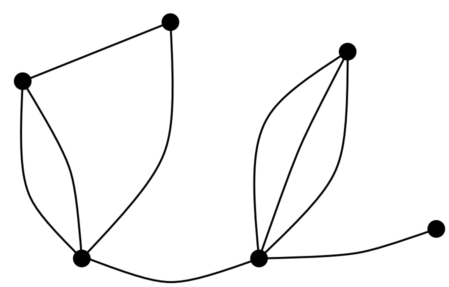
The terms above in parentheses are the *average degree of an M vertex* and the *average degree of a W vertex*. So we know:

$$\frac{\text{Avg. deg in } M}{|W|} = \frac{\text{Avg. deg in } W}{|M|}$$
$$\text{Avg. deg in } M = \frac{|W|}{|M|} \cdot \text{Avg. deg in } W$$

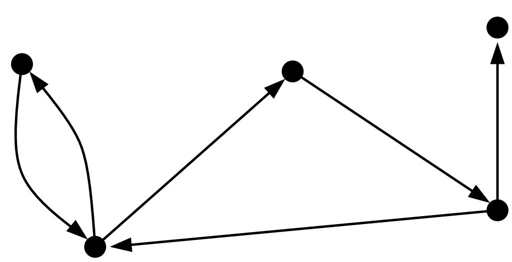
Now the Census Bureau reports that there are slightly more women than men in America; in particular $|W| / |M|$ is about 1.035. So— assuming the Census Bureau is correct— we've just proved that the University of Chicago study got bad data! On average, men have 3.5% more opposite-gender partners. Furthermore, this is totally unaffected by differences in sexual practices between men and women; rather, it is completely determined by the relative number of men and women!

1.3 Graph Variations

There are many variations on the basic notion of a graph. Three particularly common variations are described below. In a *multigraph*, there may be more than one edge between a pair of vertices. Here is an example:

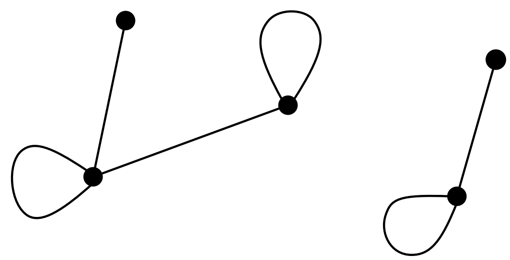


The edges in a ***directed graph*** are arrows pointing to one endpoint or the other. Here is an example:



Directed graphs are often called ***digraphs***. We denote an edge from vertex A to vertex B in a digraph by $A \longrightarrow B$. Formally, the edges in a directed graph are ordered pairs of vertices rather than sets of two vertices. The number of edges directed into a vertex is called the ***indegree*** of the vertex, and the number of edged directed out is called the ***outdegree***.

One can also allow ***self-loops***, edges with both endpoints at one vertex. Here is an example of a graph with self-loops:



Combinations of these variations are also possible; for example, one could work with directed multigraphs with self-loops.

1.4 Applications of Graphs

Graphs are the most useful mathematical objects in computer science. You can model an enormous number of real-world systems and phenomena using graphs. Once you've created such a model, you can tap the vast store of theorems about graphs to gain insight into the system you're modeling. Here are some practical situations where graphs arise:

Data Structures Each vertex represents a data object. There is a directed edge from one object to another if the first contains a pointer or reference to the second.

Attraction Each vertex represents a person, and each edge represents a romantic attraction. The graph could be directed to model the unfortunate asymmetries.

Airline Connections Each vertex represents an airport. If there is a direct flight between two airports, then there is an edge between the corresponding vertices. These graphs often appear in airline magazines.

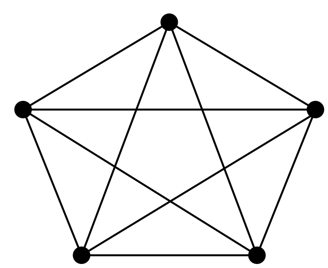
The Web Each vertex represents a web page. Directed edges between vertices represent hyperlinks.

Binary Relations A binary relation R on a set A actually ***is*** a directed graph (with self-loops, possibly) with vertex set $V = A$ and edge set $E = R$. The relation uRv holds if and only if the graph contains edge $u \longrightarrow v$.

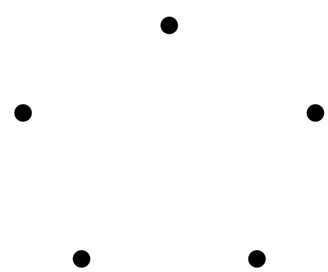
People often put numbers on the edges of a graph, put colors on the vertices, or add other ornaments that capture additional aspects of the phenomenon being modeled. For example, a graph of airline connections might have numbers on the edges to indicate the duration of the corresponding flight. The vertices in the attraction graph might be colored to indicate the person's gender.

1.5 Some Common Graphs

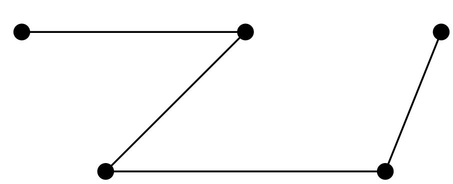
Some graphs come up so frequently that they have names. The ***complete graph*** on n vertices, also called K_n, has an edge between every pair of vertices. Here is K_5:



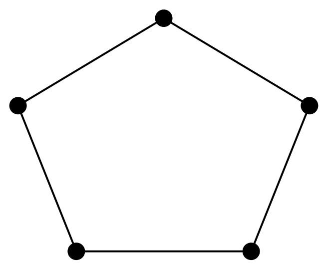
The ***empty graph*** has no edges at all. Here is the empty graph on 5 vertices:



Here is a ***path*** with 5 vertices:



And here is a ***cycle*** with 5 vertices, which is typically denoted C_5:



Paths and cycles are going to be particularly important, so let's define them precisely. A path is a graph $P = (V, E)$ of the form

$$V = \{v_1, v_2, \ldots, v_n\} \qquad E = \{v_1—v_2, v_2—v_3, \ldots, v_{n-1}—v_n\},$$

where $n \geq 1$ and vertices $v_1, \ldots, v_n$ are all distinct. Vertices v_1 and v_n are the ***endpoints*** of the path. Note that a path may consist of a single vertex, in which case both endpoints are the same. Similarly, a cycle is a graph $C = (V, E)$ of the form

$$V = \{v_1, v_2, \ldots, v_n\} \qquad E = \{v_1—v_2, v_2—v_3, \ldots, v_{n-1}—v_n, v_n—v_1\},$$

where $n \geq 3$ and $v_1, \ldots, v_n$ are all distinct. The ***length*** of a path or cycle is the number of edges it contains. For example, a path with 5 vertices has length 4, but a cycle with 5 vertices has length 5.

1.6 Isomorphism

Two graphs that look the same might actually be different in a formal sense. For example, the two graphs below are both cycles with 4 vertices:

But one graph has vertex set $\{A, B, C, D\}$ while the other has vertex set $\{1, 2, 3, 4\}$. If so, then the graphs are different mathematical objects, strictly speaking. But this is a frustrating distinction; the graphs *look the same*!

Fortunately, we can neatly capture the idea of "looks the same" and use that as our main notion of equivalence between graphs. Graphs G_1 and G_2 are ***isomorphic*** if there exists a one-to-one correspondence between vertices in G_1 and vertices in G_2 such that there is an edge between two vertices in G_1 if and only if there is an edge between the two corresponding vertices in G_2. For example, take the following correspondence between vertices in the two graphs above:

$$\begin{array}{ll} A \text{ corresponds to } 1 & B \text{ corresponds to } 2 \\ D \text{ corresponds to } 4 & C \text{ corresponds to } 3. \end{array}$$

Now there is an edge between two vertices in the graph on the left if and only if there is an edge between the two corresponding vertices in the graph on the right. Therefore, the two graphs are isomorphic. The correspondence itself is called an ***isomorphism***.

Two isomorphic graphs may be drawn to look quite different. For example, here are two different ways of drawing C_5:

The computational problem of deciding whether two graphs are isomorphic has a curious status. The problem is usually easy in practice and is suspected *not* to be NP-complete.[1] However, no one has yet found a provably efficient way to determine whether two graphs are isomorphic.

2 Connectivity

In the diagram below, the graph on the left has two pieces, while the graph on the right has just one.

[1]We'll throw around the term *NP-complete* without defining it precisely. The essential point is that no one has ever found an efficient computational approach to any NP-complete problem. Finding such an approach or proving that none exists is the most famous open question in computer science.

Let's put this observation in rigorous terms. A graph is ***connected*** if for every pair of vertices u and v, the graph contains a path with endpoints u and v as a subgraph. The graph on the left is not connected because there is no path from any of the top three vertices to either of the bottom two vertices. However, the graph on the right is connected, because there is a path between every pair of vertices.

A maximal, connected subgraph is called a ***connected component***. The graph on the left has two connected components, the triangle and the single edge. The graph on the right is entirely connected and thus has a single connected component.

2.1 A Simple Connectivity Theorem

The following theorem says that a graph with few edges must have many connected components.

Theorem 1. *Every graph $G = (V, E)$ has at least $|V| - |E|$ connected components.*

Proof. We use induction on the number of edges. Let $P(n)$ be the proposition that every graph $G = (V, E)$ with $|E| = n$ has at least $|V| - n$ connected components.

Base case: In a graph with 0 edges, each vertex is itself a connected component, and so there are exactly $|V| - 0 = |V|$ connected components.

Inductive step: Now we assume that the induction hypothesis holds for every n-edge graph in order to prove that it holds for every $(n + 1)$-edge graph, where $n \geq 0$. Consider a graph $G = (V, E)$ with $n + 1$ edges. Remove an arbitrary edge u—v and call the resulting graph G'. By the induction assumption, G' has at least $|V| - n$ connected components. Now add back the edge u—v to obtain the original graph G. If u and v were in the same connected component of G', then G has the same number of connected components as G', which is at least $|V| - n$. Otherwise, if u and v were in different connected components of G', then these two components are merged into one in G, but all other components remain. Therefore, G has at least $|V| - n - 1 = |V| - (n + 1)$ connected components.

The theorem follows by induction. $\square$

Corollary 2. *Every connected graph with n vertices has at least $n - 1$ edges.*

A couple points about the proof of Theorem 1 are worth noting. First, notice that we used induction on the number of edges in the graph. This is very common in proofs involving graphs, and so is induction on the number of vertices. When you're presented with a graph problem, these two approachs should be among the first you consider. Don't try induction on other variables that crop up in the problem unless these two strategies seem hopeless.

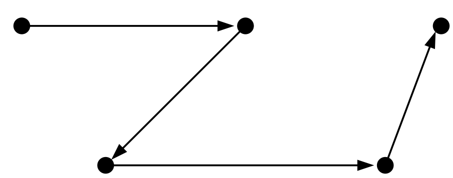
The second point is more subtle. Notice that in the inductive step, we took an arbitrary $(n + 1)$-edge graph, threw out an edge so that we could apply the induction assumption, and then put the edge back. You'll see this shrink-down, grow-back process very often in the inductive steps of proofs related to graphs. This might seem like needless effort; why not start with an n-edge graph and add one more to get an $n+1$ edge graph? That would work fine in this case, but can lead to a very nasty logical error in similar arguments. (You'll see an example in recitation.) Always use shink-down, grow-back arguments, and you'll never fall into this trap.

2.2 Connectivity in Directed Graphs

Connectivity is a bit more complicated in directed graphs. A ***directed path*** from vertex v_1 to vertex v_n is a directed graph of the form

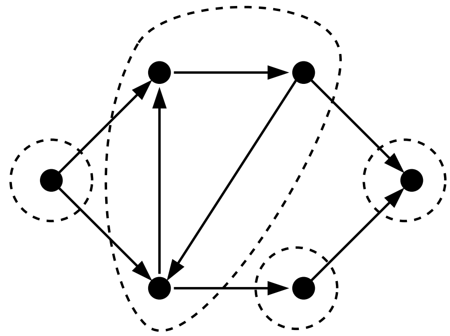
$$V = \{v_1, v_2, \ldots, v_n\} \qquad E = \{v_1 \longrightarrow v_2, v_2 \longrightarrow v_3, \ldots, v_{n-1} \longrightarrow v_n\},$$

where $n \geq 1$ and $v_1, \ldots, v_n$ are all distinct. Here is an example of a directed path:



A directed graph is ***strongly connected*** if for every pair of vertices u and v, the graph contains a directed path from u to v as a subgraph. In the diagram below, the graph on the left is strongly connected, but the graph on the right is not.

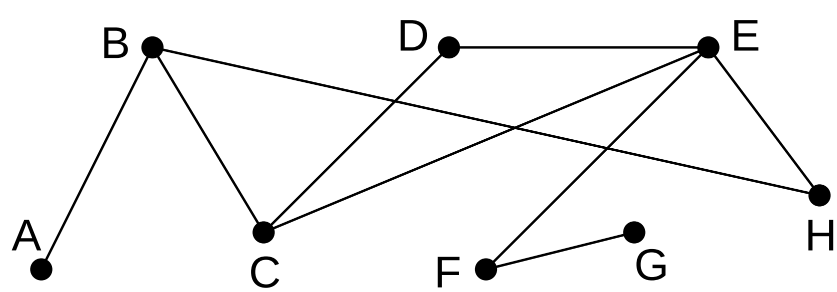
A maximal, strongly-connected subgraph of a directed graph is called a ***strongly-connected component***. The four strongly-connected components in the directed graph below are circled with dashed lines.



Suppose that we change a directed graph into an undirected graph by replacing arrows with lines. A directed graph is ***weakly connected*** if the corresponding undirected graph is connected. Similarly, vertices u and v lie in the same ***weakly-connected component*** of a directed graph if and only if u and v lie in the same connected component of the corresponding undirected graph.

2.3 Distance and Diameter

The ***distance*** between two vertices in a graph is the length of the shortest path between them. For example, the distance between two vertices in a graph of airline connections is the minimum number of flights required to travel between two cities.



In this graph, the distance between C and H is 2, the distance between G and C is 3, and the distance between A and itself is 0. If there is *no* path between two vertices, then the distance between them is said to be "infinity".

The ***diameter*** of a graph is the distance between the two vertices that are farthest apart. The diameter of the graph above is 5. The most-distant vertices are A and G, which are at distance 5 from one another.

2.3.1 Six Degrees of Separation

There is an old claim that the world has only "six degrees of separation". In other words, if you pick any two people on the planet— say a hermit in Montana and a random person off the street in Beijing— then the hermit knows someone who knows someone who knows someone who knows the Chinese pedestrian, where the word "knows" appears at most six times.

We can recast this in graph-theoretic terms. Consider a graph where the vertices are all the people on the planet, and there is an edge between two people if and only if they know each other. Then the "six degrees of separation" claim amounts to the assertion that the diameter of this graph is at most 6.

There is little hope of proving or disproving the claim, since people are constantly being born, meeting one another, and dying and no one can keep track of who-knows-who. However, precise data does exist for something similar. The University of Virginia maintains the *Oracle of Bacon* website. This is based on a graph where the vertices are actors and actresses, and there is an edge between two performers if they appeared in a movie together. The website reports that everyone is within distance 8 of Kevin Bacon. This allows us to at least obtain an upper bound on the diameter of the acting graph.

Theorem 3. *If every vertex in a graph is within distance d of some vertex v, then the diameter of the graph is at most $2d$.*

Proof. Let x and y be arbitrary vertices in the graph. There exists a path of length at most d from x to v and a path of length at most d from v to y. Therefore, there exists a path of length at most $2d$ between x and y. Thus, every pair of vertices is within distance $2d$ of one another. $\square$

2.4 Walks

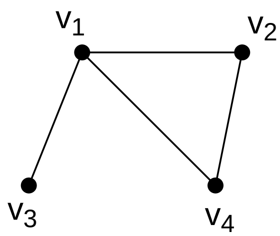
A ***walk*** in a graph G is an alternating sequence of vertices and edges of the form:

$$v_0 \; v_0 — v_1 \; v_1 \; v_1 — v_2 \; v_2 \; \ldots \; v_{n-1} \; v_{n-1} — v_n \; v_n$$

If $v_0 = v_n$, then the walk is ***closed***. Walks are similar to paths. However, a walk can cross itself, traverse the same edge multiple times, etc. There is a walk between two vertices if and only if there is a path between the vertices.

3 Adjacency Matrices

A graph can be represented by an ***adjacency matrix***. In particular, if a graph has vertices $v_1, \ldots, v_n$, then the adjacency matrix is $n \times n$. The entry in row i, column j is 1 if there is an edge v_i—v_j and is 0 if there is no such edge. For example, here is a graph and its adjacency matrix:



$$\begin{bmatrix} 0\,1\,1\,1 \\ 1\,0\,0\,1 \\ 1\,0\,0\,0 \\ 1\,1\,0\,0 \end{bmatrix}$$

The adjacency matrix of an undirected graph is always symmetric about the diagonal line running from the upper left entry to the lower right. The adjacency matrix of a directed graph need not be symmetric, however. Entries on the diagonal of an adjacency matrix are nonzero only if the graph contains self-loops.

Adjacency matrices are useful for two reasons. First, they provide one way to represent a graph in computer memory. Second, by mapping graphs to the world of matrices, one can bring all the machinery of linear algebra to bear on the study of graphs. (For example, one can analyze a highly-prized quality of graphs called *expansion* by looking at eigenvalues of the adjacency matrix.)

Here we prove a simpler theorem in this vein. If M is a matrix, then M_{ij} denotes the entry in row i, column j. Let M^k denote the k-th power of M. As a special case, M^0 is the identity matrix.

Theorem 4. *Let G be a digraph (possibly with self-loops) with vertices $v_1, \ldots, v_n$. Let M be the adjacency matrix of G. Then M^k_{ij} is equal to the number of length-k walks from v_i to v_j.*

Proof. We use induction on k. The induction hypothesis is that M^k_{ij} is equal to the number of length-k walks from v_i to v_j, for all i, j.

Each vertex has a length-0 walk only to itself. Since $M^0_{ij} = 1$ if and only if $i = j$, the hypothesis holds for $k = 0$.

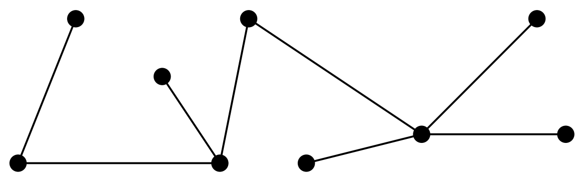
Now suppose that the hypothesis holds for some $k \geq 0$. We prove that it also holds for $k + 1$. Every length-$(k+1)$ walk from v_i to v_j consists of a length k walk from v_i to some intermediate vertex v_m followed by an edge v_m—v_j. Thus, the number of length-$(k+1)$ walks from v_i to v_j is equal to:

$$M^k_{iv_1} M_{v_1 j} + M^k_{iv_2} M_{v_2 j} + \ldots + M^k_{iv_n} M_{v_n j}$$

This is precisely the value of M^{k+1}_{ij}, so the hypothesis holds for $k + 1$ as well. The theorem follows by induction. $\square$

4 Trees

A connected, acyclic graph is called a ***tree***. (A graph is *acyclic* if no subgraph is a cycle.) Here is an example of a tree:



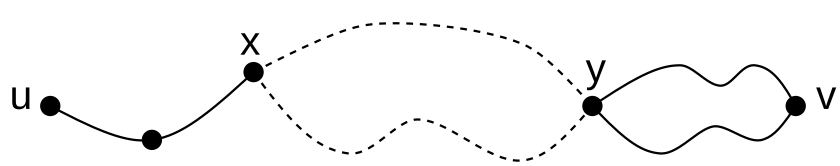
A vertex of degree one is called a ***leaf***. In this example, there are 5 leaves.

The graph shown above would no longer be a tree if any edge were removed, because it would no longer be connected. The graph would also not remain a tree if any edge were added, because then it would contain a cycle. Furthermore, note that there is a unique path between every pair of vertices. These features of the example tree are actually common to all trees.

Theorem 5. *Every tree $T = (V, E)$ has the following properties:*

1. *There is a unique path between every pair of vertices.*
2. *Adding any edge creates a cycle.*
3. *Removing any edge disconnects the graph.*
4. $|V| = |E| + 1$.

Proof. 1. There is at least one path between every pair of vertices, because the graph is connected. Suppose that there are two different paths between vertices u and v. Beginning at u, let x be the first vertex where the paths diverge, and let y be the next vertex they share. Then there are two paths from x to y with no common edges, which defines a cycle. This is a contradiction, since trees are acyclic. Therefore, there is exactly one path between every pair of vertices.



2. An additional edge u—v together with the unique path between u and v forms a cycle.

3. Suppose that we remove edge u—v. Since a tree contained a unique path between u and v, that path must have been u—v. Therefore, when that edge is removed, no path remains, and so the graph is not connected.

4. We use induction on $|V|$. For a tree with a single vertex, the claim holds since $|E| + 1 = 0 + 1 = 1$. Now suppose that the claim holds for all n-vertex trees and consider an $(n + 1)$-vertex tree T. Let $v_1, \ldots, v_m$ be the sequence of vertices on the longest path in T.

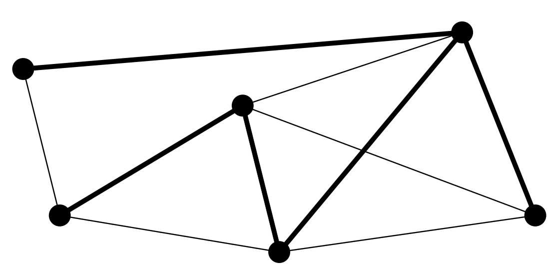
   There can not be an edge v_1—v_i for $2 < i \leq m$; otherwise, vertices $v_1, \ldots, v_i$ would from a cycle. Furthermore, there can not be an edge v_1—u where u is not on the path; otherwise, we could make the path longer. Therefore, the only edge incident to v_1 is v_1—v_2. Deleting v_1 and this edge gives a smaller tree for which the equation $|V| = |E| + 1$ holds by induction. If we add back the vertex v_1 and the edge v_1—v_2, then the equation still holds. Thus, the claim holds for T and, by induction, for all trees.

□

Many subsets of the four properties above, together with connectedness and lack of cycles, are sufficient to characterize all trees. For example, a connected graph that satisfies $|V| = |E| + 1$ is necessarily a tree.

4.1 Spanning Trees

Trees are everywhere. In fact, every connected graph $G = (V, E)$ contains a ***spanning tree*** $T = (V, E')$ as a subgraph. (Note that G and T have the same set of vertices.) For example, here is a connected graph with a spanning tree highlighted.



Theorem 6. *Every connected graph $G = (V, E)$ contains a spanning tree.*

Proof. Let $T = (V, E')$ be a connected subgraph of G with the smallest number of edges. We show that T is acyclic by contradiction. So suppose that T has a cycle with the following edges:

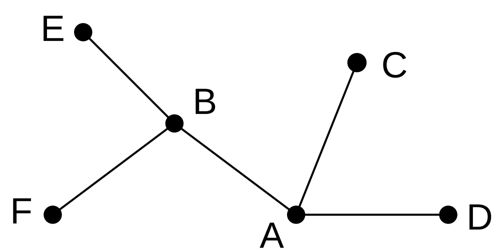
$$v_0 \text{—} v_1, v_1 \text{—} v_2, \ldots, v_n \text{—} v_0$$

Suppose that we remove the last edge, v_n—v_0. If a pair of vertices x and y was joined by a path not containing v_n—v_0, then they remain joined by that path. On the other hand, if x and y were joined by a path containing v_n—v_0, then they remain joined by a path containing the remainder of the cycle. This is a contradiction, since T was defined to be a connected subgraph of G with the smallest number of edges. Therefore, T is acyclic. $\square$

4.2 Tree Variations

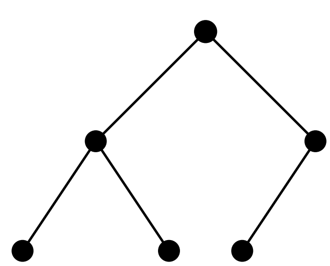
Trees come up often in computer science. For example, information is often stored in tree-like data structures and the execution of many recursive programs can be regarded as a traversal of a tree.

There are many varieties of trees. For example, a ***rooted tree*** is a tree with one vertex identified as the ***root***. Let u—v be an edge in a rooted tree such that u is closer to the root than v. Then u is the ***parent*** of v, and v is the ***child*** of u.

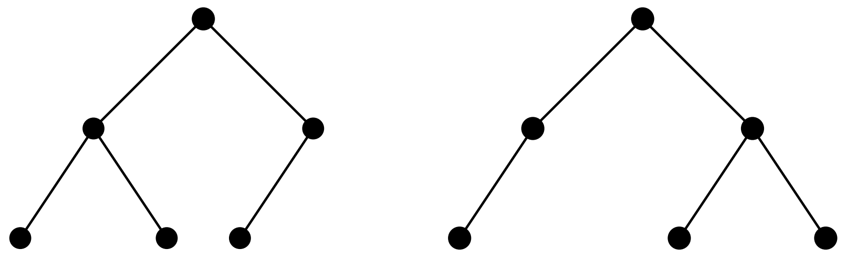


In the tree above, suppose that we regard vertex A as the root. Then E and F are the children of B, and A is the parent of B, C, and D.

A ***binary*** tree is a rooted tree in which every vertex has at most two children. Here is an example, where the topmost vertex is the root.



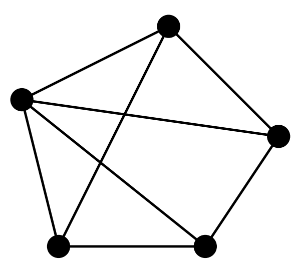
In an ***ordered, binary*** tree, the children of a vertex v are distinguished. One is called the ***left child*** of v, and the other is called the ***right child***. For example, if we regard the two binary trees below as unordered, then they are equivalent. However, if we regard these trees as ordered, then they are different.



5 Coloring Graphs

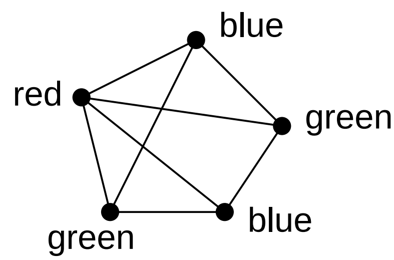
Each term, the MIT Schedules Office must assign a time slot for each final exam. This is not easy, because some students are taking several classes with finals, and a student can take only one test during a particular time slot. The Schedules Office wants to avoid all conflicts, but make the exam period as short as possible.

We can recast this scheduling problem as a question about coloring the vertices of a graph. Create a vertex for each course with a final exam. Put an edge between two vertices if some student is taking both courses. For example, the scheduling graph might look like this:



Next, identify each time slot with a color. For example, Monday morning is red, Monday afternoon is blue, Tuesday morning is green, etc.

Assigning an exam to a time slot is now equivalent to coloring the corresponding vertex. The main constraint is that adjacent vertices must get different colors; otherwise, some student has two exams at the same time. Furthermore, in order to keep the exam period short, we should try to color all the vertices using as few different colors as possible. For our example graph, three colors suffice:



This coloring corresponds to giving one final on Monday morning (red), two Monday afternoon (blue), and two Tuesday morning (green).

5.1 k-Coloring

Many other resource allocation problems boil down to coloring some graph. In general, a graph G is k-***colorable*** if each vertex can be assigned one of k colors so that adjacent vertices get different colors. The smallest sufficient number of colors is called the ***chromatic number*** of G. The chromatic number of a graph is generally difficult to compute, but the following theorem provides an upper bound:

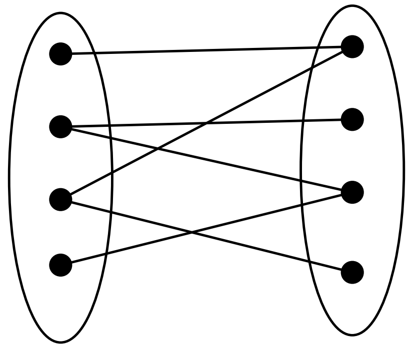
Theorem 7. *A graph with maximum degree at most k is $(k+1)$-colorable.*

Proof. We use induction on the number of vertices in the graph, which we denote by n. Let $P(n)$ be the proposition that an n-vertex graph with maximum degree at most k is $(k+1)$-colorable. A 1-vertex graph has maximum degree 0 and is 1-colorable, so $P(1)$ is true.

Now assume that $P(n)$ is true, and let G be an $(n+1)$-vertex graph with maximum degree at most k. Remove a vertex v, leaving an n-vertex graph G'. The maximum degree of G' is at most k, and so G' is $(k+1)$-colorable by our assumption $P(n)$. Now add back vertex v. We can assign v a color different from all adjacent vertices, since v has degree at most k and $k+1$ colors are available. Therefore, G is $(k+1)$-colorable. The theorem follows by induction. $\square$

5.2 Bipartite Graphs

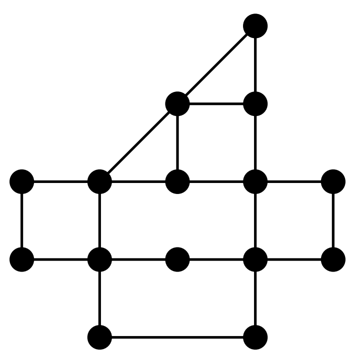
The 2-colorable graphs are important enough to merit a special name; they are called ***bipartite graphs***. Suppose that G is bipartite. Then we can color every vertex in G either black or white so that adjacent vertices get different colors. Then we can put all the black vertices in a clump on the left and all the white vertices in a clump on the right. Since every edge joins differently-colored vertices, every edge must run between the two clumps. Therefore, every bipartite graph looks something like this:



Bipartite graphs are both useful and common. For example, every path, every tree, and every even-length cycle is bipartite.

6 Traversing a Graph

Can you walk every hallway in the Museum of Fine Arts *exactly once*? If we represent hallways and intersections with edges and vertices, then this reduces to a question about graphs. For example, could you visit every hallway exactly once in a museum with this floorplan?



The entire field of graph theory began when Euler asked a similar question about the seven bridges of Königsberg. In his honor, an ***Euler walk*** is a walk that traverses every edge in a graph exactly once. Similarly, an ***Euler tour*** is an Euler walk that starts and finishes at the same vertex. Graphs with Euler tours and Euler walks both have simple characterizations.

Theorem 8. *A connected graph has an Euler tour if and only if every vertex has even degree.*

Proof. If a graph has an Euler tour, then every vertex must have even degree; in particular, a vertex visited k times on an Euler tour must have degree $2k$.

Now suppose every vertex in graph G has even degree. Let W be the longest walk in G that traverses every edge *at most* once:

$$W = v_0\ v_0\text{—}v_1\ v_1\ v_1\text{—}v_2\ v_2\ \ldots\ v_{n-1}\ v_{n-1}\text{—}v_n\ v_n$$

The walk W must traverse every edge incident to v_n; otherwise, the walk could be extended. In particular, the walk traverses two of these edges each time it passes through v_n and traverses v_{n-1}—v_n at the end of the walk. This accounts for an odd number of edges, but the degree of v_n is even by assumption. Therefore, the walk must also begin at v_n; that is, $v_0 = v_n$.

Suppose that W is not an Euler tour. Because G is a connected graph, we can find an edge not in W but incident to some vertex in W. Call this edge u—v_i. But then we can construct a longer walk:

$$u\ u\text{—}v_i\ v_i\ v_i\text{—}v_{i+1}\ \ldots\ v_{n-1}\text{—}v_n\ v_n\ v_0\text{—}v_1\ \ldots\ v_{i-1}\text{—}v_i\ v_i$$

This contradicts the definition of W, so W must be an Euler tour after all. □

Corollary 9. *A connected graph has an Euler walk if and only if either 0 or 2 vertices have odd degree.*

A similar theorem holds for directed walks on directed graphs.

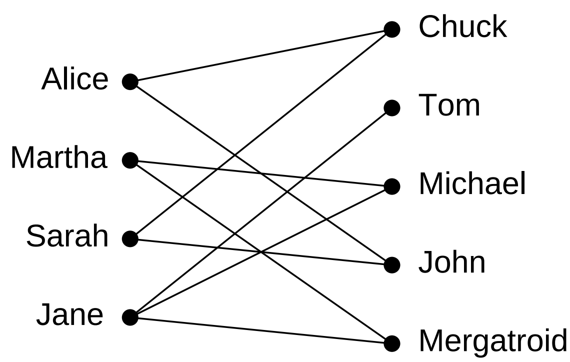
Theorem 10. *A weakly-connected directed graph has an Euler tour if and only if the indegree of every vertex is equal to the outdegree.*

Hamiltonian cycles are the unruly cousins of Euler tours. A ***Hamiltonian cycle*** visits every *vertex* in a graph exactly once. There is no simple characterization of all graphs with a Hamiltonian cycle. In fact, determining whether a given graph has a Hamiltonian cycle is NP-complete.

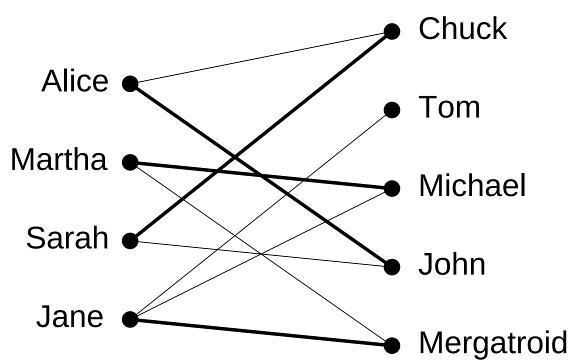
7 Hall's Marriage Theorem

A class contains some girls and some boys. Each girl likes some boys and does not like others. Under what conditions can each girl be paired up with a boy that she likes?

We can model the situation with a bipartite graph. Create a vertex on the left for each girl and a vertex on the right for each boy. If a girl likes a boy, put an edge between them. For example, we might obtain the following graph:



In graph terms, our goal is to find a ***matching*** for the girls; that is, a subset of edges such that exactly one edge is incident to each girl and at most one edge is incident to each boy. For example, here is one possible matching for the girls:



Hall's Marriage Theorem states necessary and sufficient conditions for the existence of a matching in a bipartite graph. Hall's Theorem is a remarkably useful mathematical tool, a hammer that bashes many problems. Moreover, it is the tip of a conceptual iceberg, a special case of the "max-flow, min-cut theorem", which is in turn a byproduct of "linear programming duality", one of the central ideas of algorithmic theory.

We'll state and prove Hall's Theorem using girl-likes-boy terminology. Define *the set of boys liked by a given set of girls* to consist of all boys liked by at least one of those girls. For example, the set of boys liked by Martha and Jane consists of Tom, Michael, and Mergatroid.

For us to have any chance at all of matching up the girls, the following ***marriage condition*** must hold:

Every subset of girls likes at least as large a set of boys.

For example, we can not find a matching if some 4 girls like only 3 boys. Hall's Theorem says that this necessary condition is actually sufficient; if the marriage condition holds, then a matching exists.

Theorem 11. *A matching for a set of girls G with a set of boys B can be found if and only if the marriage condition holds.*

Proof. First, let's suppose that a matching exists and show that the marriage condition holds. Consider an arbitrary subset of girls. Each girl likes at least the boy she is matched with. Therefore, every subset of girls likes at least as large a set of boys. Thus, the marriage condition holds.

Next, let's suppose that the marriage condition holds and show that a matching exists. We use strong induction on $|G|$, the number of girls. If $|G| = 1$, then the marriage condition implies that the lone girl likes at least one boy, and so a matching exists. Now suppose that $|G| \geq 2$. There are two possibilities:

1. Every proper subset of girls likes a *strictly larger* set of boys. In this case, we have some latitude: we pair an arbitrary girl with a boy she likes and send them both away. The marriage condition still holds for the remaining boys and girls, so we can match the rest of the girls by induction.

2. Some proper subset of girls $X \subset G$ likes an *equal-size* set of boys $Y \subset B$. We match the girls in X with the boys in Y by induction and send them all away. We will show that the marriage condition holds for the remaining boys and girls, and so we can match the rest of the girls by induction as well.

   To that end, consider an arbitrary subset of the remaining girls $X' \subseteq G - X$, and let Y' be the set of remaining boys that they like. We must show that $|X'| \leq |Y'|$. Originally, the combined set of girls $X \cup X'$ liked the set of boys $Y \cup Y'$. So, by the marriage condition, we know:

   $$|X \cup X'| \quad \leq \quad |Y \cup Y'|$$

   We sent away $|X|$ girls from the set on the left (leaving X') and sent away an equal number of boys from the set on the right (leaving Y'). Therefore, it must be that $|X'| \leq |Y'|$ as claimed.

In both cases, there is a matching for the girls. The theorem follows by induction. □

There is an efficient algorithm for finding a matching in a bipartite graph, if one exists. Thus, if a problem can be reduced to finding a matching, the problem is essentially solved from a computational perspective.