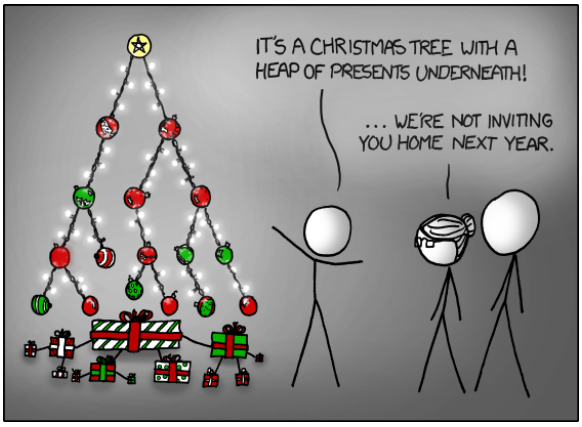
1 Introduction



xkcd webcomic by Randall Munroe (`xkcd.com/835/`). (XXX:Need to email to make sure we have permission per Creative Commons Attribution-NonCommercial 2.5 License.)

Imagine a world where people have no friends. Where roads never intersect. Where computers are not interconnected. This world without networks would be a very sad and boring place, where nothing happens — and even if something happened, nobody would know. Such a world is unimaginable, because our life is completely defined by networks: relationships, interactions, communications, and the Web. Biological networks governing the interactions between genes in our cells determine our development, neural networks in our brain make us think, information networks guide our knowledge and culture, transportation networks allow us to move, and social networks sustain our life.

Networks are a general yet powerful way to represent and study simple and complex interactions. This book explores the study of networks and how they help us understand the patterns of connections and relationships that shape our lives. In essence, a network is the simplest description of a set of interconnected entities, which we call *nodes*, and their connections, which we call *links*. The network representation is so general and powerful because it strips out many details of a particular system and focuses on the interactions among its elements. Networks are thus used to study widely diverse systems. Nodes can represent all sorts of entities: people, cities, computers, Web sites, concepts, cells, genes, species, and so on. Links represent relationships or interactions between these entities: friendships among people,

flights between airports, packets exchanged among computers on the Internet, links between Web pages, synapses between neurons, and so on.

1.1 Network Examples

Before we introduce the basic concepts, definitions, and nomenclature about networks, let us explore a few examples of social, infrastructure, information, and biological networks. Data for all the examples presented here is available on the book's Github repository.[1]

1.1.1 Social Networks

A social network is a group of people connected by some type of relationship. Friendship, collaboration, romance, or mere acquaintance are all examples of social relationships that connect pairs of people. When we talk about a social network, we typically think of a particular type of relationship. A person is represented by a node in the social network, and the relationship is represented by a link between two people. The network is therefore a representation of the relationship. It allows us to talk about the relationship, to describe it and analyze it at a level that goes beyond a pair of people.

There are many different types of social networks, and they are important to study. Health workers analyze networks of sexual relationships to find ways to combat the spread of sexually transmitted diseases. Economists study job referral networks to address inequality and segregation in labor markets. And scientists inspect coauthorship and citation networks in scholarly publications to identify influential thinkers and ideas.

These days we use online social networking sites to keep track of our social ties. Platforms like Facebook and Twitter allow us to keep in touch with many people — partners, friends, colleagues, and acquaintances, sometimes in the hundreds — and communicate with them conveniently, irrespective of distance. Figure 1.1 illustrates a familiar network, a portion of the Facebook social graph. In this network, nodes are people with a Facebook account at a US university, and connections may represent different types of relationship, from real friendship to mere acquaintance. Just looking at the network visualization reveals something about the underlying social structure. Some people have more connections; we represent this by making the corresponding nodes larger and darker. These might be popular students, teachers, or administrators. We also notice that the network is roughly divided into two parts. The data is anonymized so we cannot tell for sure, but a possible interpretation is that the larger sub-network comprises mostly undergraduate students, and the smaller one includes mostly graduate students. There are connections between

[1] `github.com/CambridgeUniversityPress/IntroductoryNetworkScience`

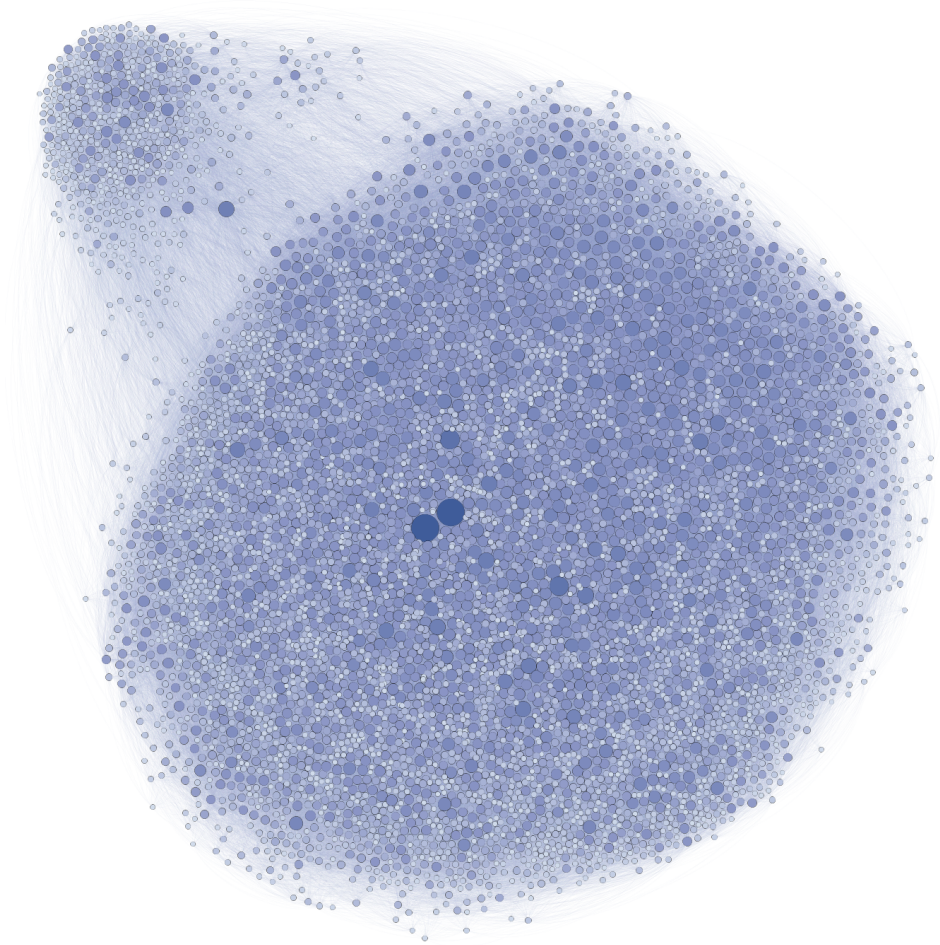


Fig. 1.1 Visualization of a network of Facebook users at Northwestern University. Nodes represent people, and links stand for Facebook friend connections.

nodes in the two groups, but not as many as among nodes within each group. In other words, undergraduate students are more likely to be friends with other undergrads than with grad students. Later we will introduce formal names for all these observations, which are typical of most social networks.

The availability of data from online social networks is very exciting for scientists. We can study human interactions at a scale and resolution that was never possible in the past: who befriends whom, who pays attention to what, who likes what, what gets recommended, and how this information propagates through the network. This data provides us with an unprecedented opportunity to discover, track, mine, and model what people do. Like the telescope gave us a first view of distant planets and stars, and the microscope allowed to peek into living tissue and micro-organisms, social media are enabling the study of social systems and human activity. However,

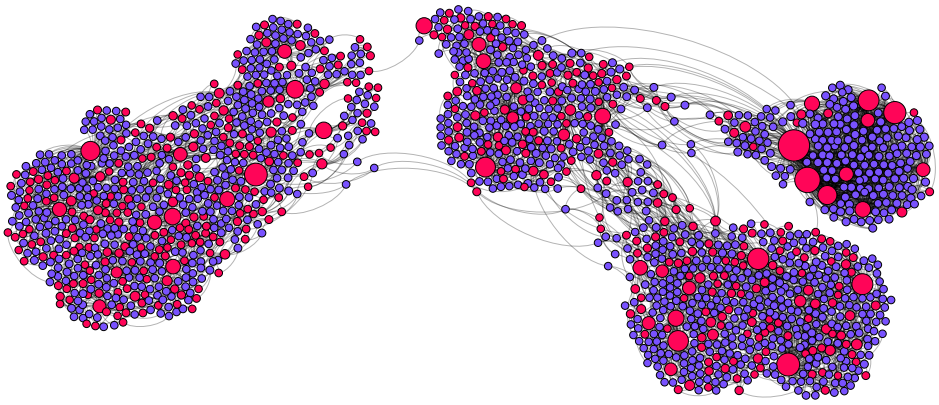


Fig. 1.2 A movie-star network, based on a small sample of movies, actors and actresses from the Internet Movie Database. Nodes represent movies (blue) or actors/actresses (red). A link connects an actor or actress to a movie in which they starred.

as exciting as these opportunities are to researchers, they don't come without risks of abuse. Online interactions expose our private personal information. We've all heard stories about employers finding embarrassing pictures of prospective employees, or scandals related to hackers and political organizations amassing data about millions of users. The dangers can be subtle. Knowing a little bit of information about a large number of people can reveal a lot more than intended. Using data from Facebook, two MIT students found that just by looking at the gender and sexuality of a person's online friends, they could predict whether the person was gay. Online social networks also make impersonation easy to do and hard to detect. Social phishing is the technique of impersonating a victim's friend (as inferred from an online social network) to induce the victim to disclose sensitive information. Two Indiana University students demonstrated that they were able to obtain the secret passwords of 72% of victims in this way.

Data about a social network can be extracted from many sources. If we want to map human mobility patterns to improve urban transportation networks, we can collect call data from cell phones. If we want to map coauthorship among scientists, we can extract the names from a database of scientific publications. (This is not a trivial exercise, because several scientists have common names.) If we want to map the collaboration among movie stars, we can extract movie credits data from the Internet Movie Database (`IMDB.com`). Figure 1.2 illustrates such a network. Here, there are actually two kinds of nodes: movies and actors/actresses. We draw a link between an actress and a movie in which she has starred. Although the depicted network captures only a tiny portion of the movie database, we again notice some clear patterns. Larger nodes have more connections, representing stars who acted in many movies. We also see that the network is structured into several dense groups, likely associated with periods, languages, or film genres.

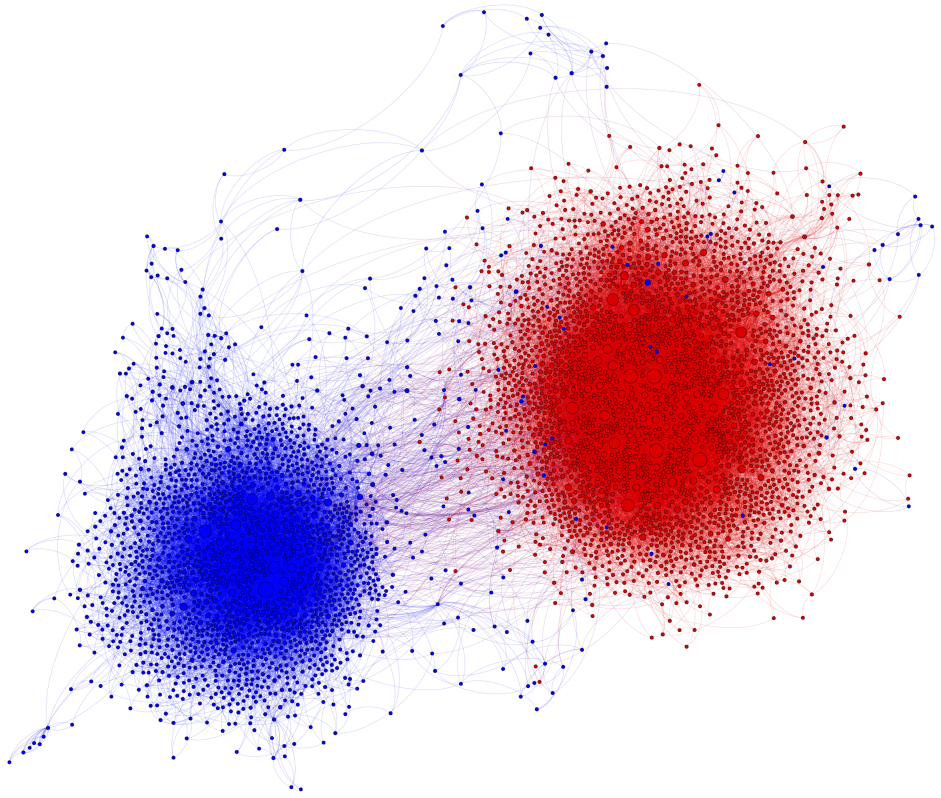


Fig. 1.3 A retweet network on Twitter, among people sharing posts about US politics. Links represent retweets of posts that used hashtags such as `#tcot` and `#p2`, associated with conservative and progressive messages respectively, around the 2010 US midterm election. When Bob retweets Alice, we draw a directed link from Alice to Bob to indicate that a message has propagated from her to him. The direction of the links is not shown.

1.1.2 Communication Networks

In the Facebook and movie networks, links are reciprocal: you cannot friend someone on Facebook unless they agree, and you cannot star in a movie without being listed in the credits. Not all social networks have reciprocal links, however. For example, Twitter is a popular social network with links that are not necessarily reciprocal: Alice can follow Bob without Bob necessarily following Alice back. As a result, the relationships captured by the Twitter network is not friendship; you follow someone to see what they post. When you retweet a post, your followers see it. This is a good way to share information broadly, so Twitter is a social network mainly aimed at spreading information — a communication network. The retweet network in Figure 1.3 illustrates the spread of political messages during a US

election. Larger nodes are those with more outgoing links, because how many times users are retweeted by others is a way to measure their influence. You probably noticed immediately a more striking pattern: conservative users (red nodes) mostly retweet messages from other conservatives, while progressive users (blue nodes) similarly share progressive content. In fact, such preferential patterns of the social connections allows us to guess a person's political leaning with high accuracy.

Networks like Twitter let us trace the diffusion of hashtags and news, observing how ideas and cultural concepts spread from person to person. But social media are also used to spread misinformation, which is unknowingly passed on by gullible users. Using fake news sites and automated or semi-automated accounts known as "social bots," a malicious entity can cheaply and effectively generate and amplify a disinformation campaign, either for political purposes or to monetize traffic through ads. In recent years we have observed a sharp increase in these types of manipulation of social media on a global scale. If one can control what information people see online, one can manipulate their opinions. This is a threat to democracy in many countries, because without well-informed voters one cannot have free elections. Academic researchers and industry engineers are working hard to develop countermeasures. Understanding the structure and dynamics of information diffusion networks in a critical component of these efforts.

The social links in Twitter are in place before a user generates a post, which is typically broadcast to all of the user's followers. In email, just like in social networks, nodes are people. However, each message is intended for one or more specific recipients. Links are based on the messages exchanged. Email does not depend on a particular platform; the protocol is open and distributed, so that no single organization controls all of the traffic. As a result, email is still among the most widely used communication networks. Figure 1.4 illustrates an example of an email network. Again, links are directed from the sender to the receiver of an email, indicated by arrows. Node size and color represent two different features: number of incoming and outgoing links, respectively: a larger node receives emails from more people, and a darker node sends emails to more people. The fact that larger nodes tend to be darker and vice versa tells us that there is a correlation between sending and receiving emails.

1.1.3 The Web and Wikipedia

The Web is the largest information network. While it is now used to provide all kinds of services, it was originally just a network of documents (pages) connected by "hyperlinks," or clickable links. In the early 1990s, Tim Berners-Lee wanted to simplify access by scientists to information about high-energy physics experiments at the European Organization for Nuclear Research (CERN) near Geneva. He came up with three key ideas: (1) a naming system for pages, the Uniform Resource Locator (URL); (2) a simple language for writing documents, called HyperText Markup Language (HTML), including hyperlinks from one page to another; and (3) a simple protocol called HyperText Transfer Protocol (HTTP) for clients (browsers)

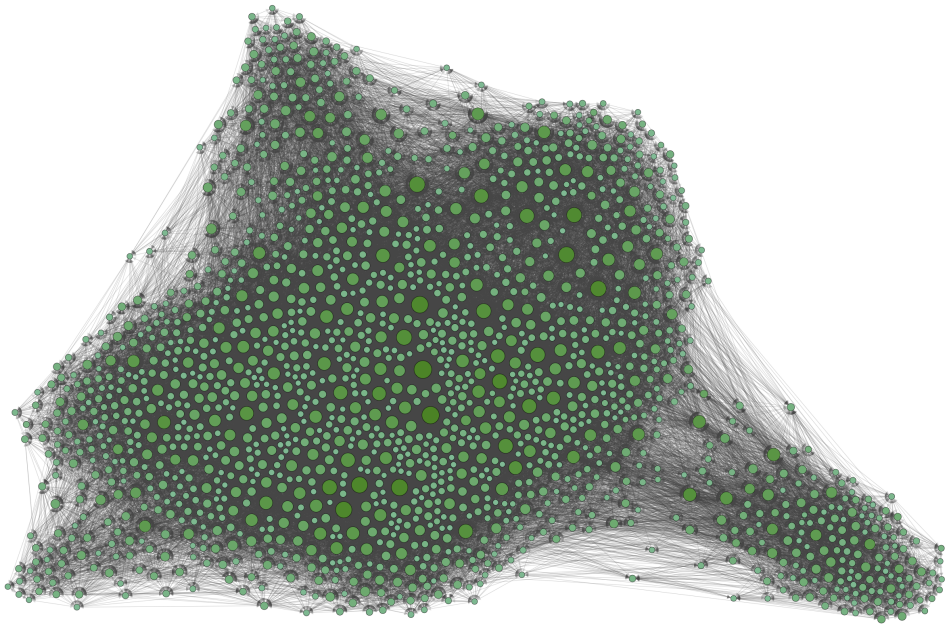


Fig. 1.4 A network based on a database of emails generated by Enron employees. The data was acquired by the US Federal Energy Regulatory Commission during its investigation after the company's collapse in 2001. At the conclusion of the investigation, the emails were deemed to be in the public domain and made publicly available for historical research and academic purposes. Only a small portion of the central core of the network is shown. The direction of the links is shown by arrows.

to talk to servers. With these three components, the Web was born. Berners-Lee even implemented the first Web server and browser software to download pages and media from servers by clicking on links. We can actually see two networks at play here: the static "link graph" made of Web pages and links, and the dynamic traffic network emerging from people navigating the Web. To paraphrase the classic philosophical riddle, if there is a link between two pages but nobody clicks on it, is it really part of the Web? The answer of course depends on which network we are thinking about when we say "Web." In later chapters we will spend more time exploring both of these information networks.

The Web is too large to visualize even a small portion of it in a meaningful way. Let us focus on Wikipedia, which is a network of pages (articles) on a single website. The Wikipedia is a collaborative encyclopedia edited by thousands of volunteers around the world, and it is one of the most popular destinations on the Web. There are versions of Wikipedia in many languages, so let us focus on the English one. Still, the English Wikipedia is a huge network with millions of articles (and growing!). So let us focus on just a small subset of articles about math, shown in Figure 1.5. Again, node size and color represent the number of incoming and outgoing links,

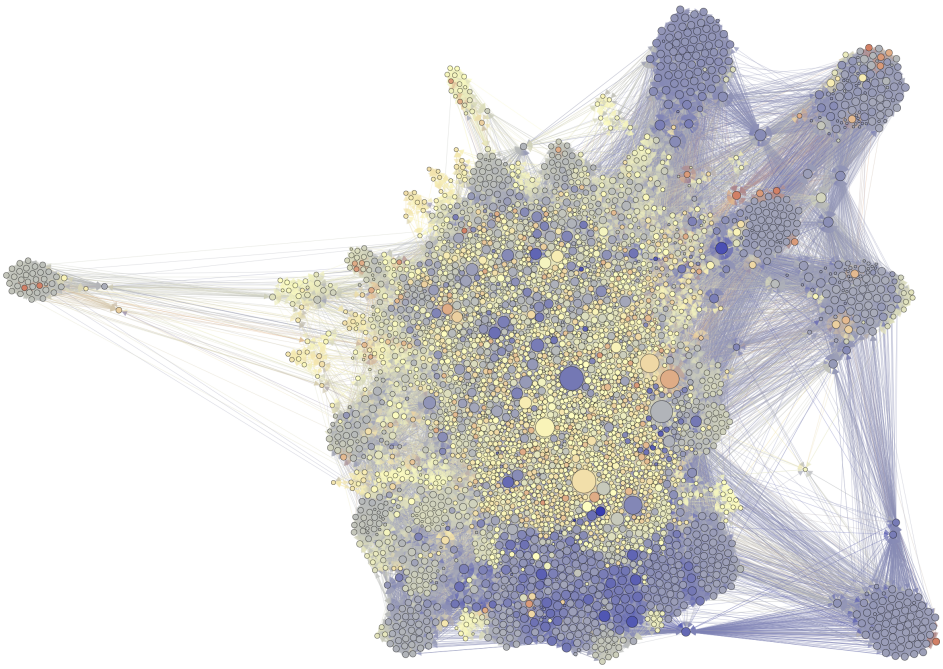


Fig. 1.5 A portion of the Wikipedia information network. Nodes are articles about math. We only consider links among Wikipedia articles, and disregard links to external pages.

respectively. We note that in this network, the articles with many outgoing links are not necessarily the same as those with many incoming links. For example, we see some large orange nodes, with many incoming but few outgoing links; and some small blue nodes, with few incoming and many outgoing links. Another feature of this network is the presence of a large "core" and several smaller groups. Here the groups are tightly connected clusters of articles on specific topics, or branches of math. We also observe several "bridge" nodes that connect multiple clusters. All of these features are found in many real-world networks.

1.1.4 The Internet

We often think of the Internet as a network of computers and other connected devices, but in reality it is a "network of networks." In fact the word originates from *internetworking,* or connecting different computer networks through special nodes called *routers*. We can therefore observe the Internet at many levels: at the lowest level we have hubs and switches, which connect individual computers in the same local or wide-area network, respectively. These networks are connected by routers, so we can zoom out and think of the network of routers. If we zoom out further we find groups of networks managed by an Internet Service Provider (ISP). This organization decides its internal network topology (how routers are

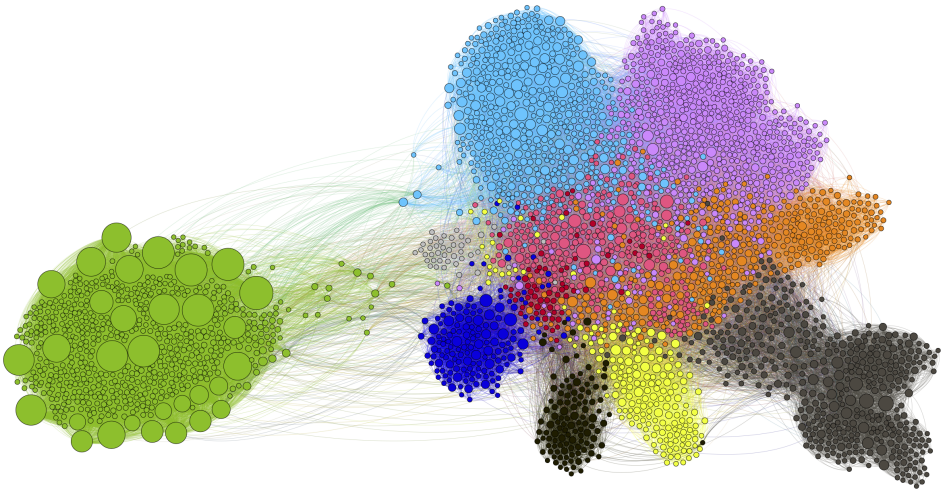


Fig. 1.6 A portion of the Internet router network. The map is a snapshot generated by the Center for Applied Internet Data Analysis (CAIDA.org) using tools that send out small packets of data (probes) between Internet hosts. Colors are assigned according to a community detection algorithm (discussed in Chapter 7) that identifies dense clusters, likely corresponding to autonomous systems.

connected) autonomously, and therefore is also called "autonomous system" (AS). Special "border" routers connect one AS to another, forming what we call the AS network.

Figure 1.6 shows a small portion of the Internet router network. Although the Internet has evolved without central control or coordination, ISPs follow local rules on how to connect their routers. They try to provide the best service at the lowest cost. Certain regularities emerge as a result. For instance, the portion of the Internet that carries the most traffic is often referred to as "backbone." The large telecommunication companies that manage the Internet backbone have a significant interest in preventing disruption, so they engineer their networks with a lot of redundancy. We thus observe a dense "core," with large routers connected to each other. As we move toward the "periphery" of the Internet — our home routers — the network is more sparsely connected. Such a hierarchical core-periphery structure is common in many different types of networks. Another peculiarity we observe in the picture of the router network is the presence of very large nodes in one of the peripheral clusters, indicating routers with many connections. This may actually be a measurement error resulting from a bias in the probe methodology used to map these networks. In fact, a router can only have a limited number of connections due to hardware constraints. Let it serve as a reminder that if we use a flawed method to collect data about a network, its analysis may lead to wrong conclusions.

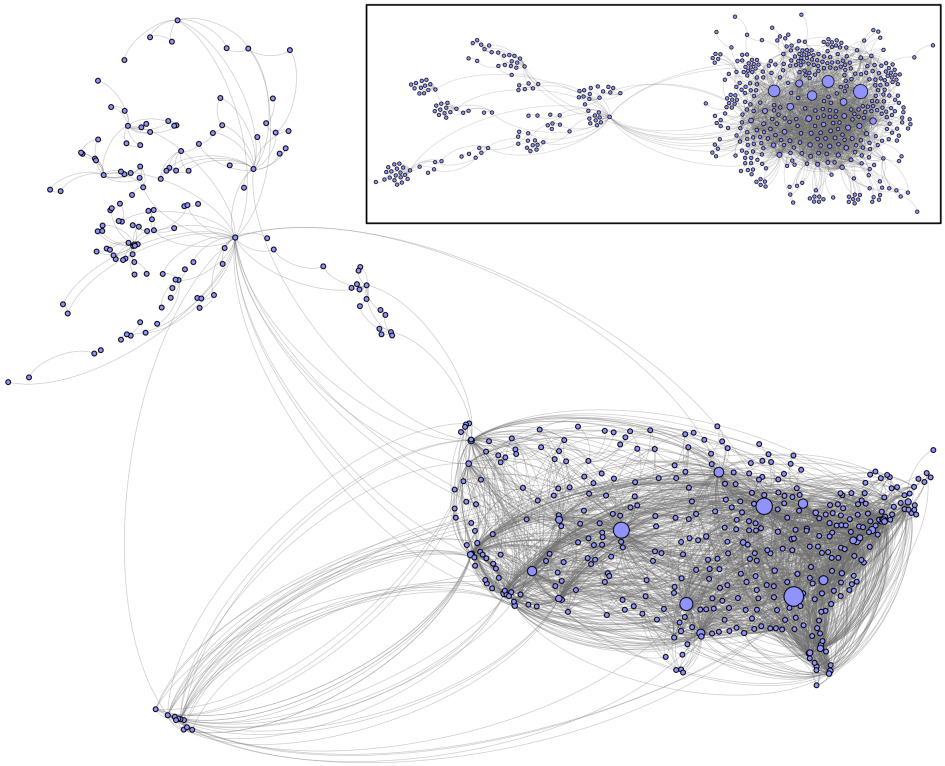


Fig. 1.7 The US air transportation network (flight data from `OpenFlights.org`). Nodes are positioned according to the geographic coordinates of the corresponding airports, so that we can make out the shape of the continental United States, Alaska, and Hawaii. The airport hubs with most connections (e.g., Atlanta, Chicago, Denver) are clearly recognizable. The inset maps the same network, but with a different "force-directed" layout, discussed in Section 1.3.

1.1.5 Transportation Networks

Another important class of networks concern various types of transportation. Nodes are locations: cities, road intersections, airports, ports, train or subway stations. These networks are very different from one another, however. Road networks, for example, evolve in a local fashion to minimize the distance traveled between nearby cities. This leads to the emergence of grid-like structures, in which most nodes have a comparable number of connections — say, four-way intersections. Figure 1.7 shows an air transportation network, which does not have a grid structure. The reason is that airlines try to minimize the number of hops between source and destination without adding costly direct flights between low-traffic airports. The simple solution is to add flights connecting airports to existing hubs. As a result, air flight networks

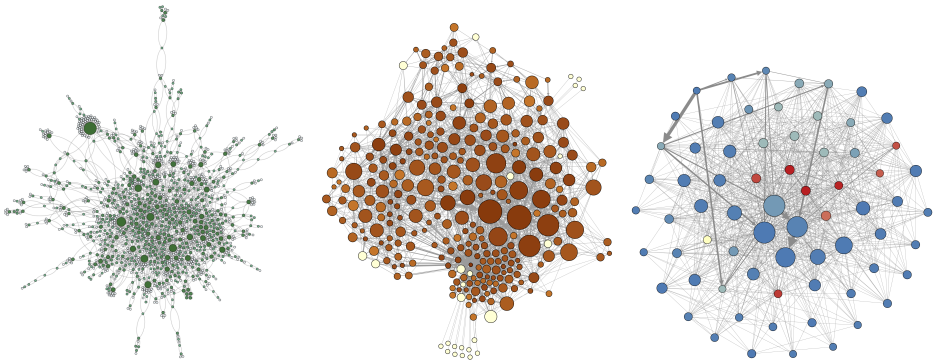


Fig. 1.8 Three biological networks. Left: protein interaction network of yeast. Center: neural network of the roundworm c. elegans. Large and red nodes represent neurons with more outgoing and incoming synapses, respectively. Right: Food web of species in the Florida Everglades. A directed link goes from a prey to a predator species. The weight (width) of a link represents the energy flux between two species. So, if species A preys on species B, there is a directed link going from B to A. Node size and color represent incoming and outgoing links, respectively, so that large blue nodes are the species at the top of food chain, while small red nodes are are the species at the bottom.

display "hub and spoke" structure: a few hubs have huge numbers of links, while the majority of nodes have very few connections.

1.1.6 Biological Networks

Within the cells inside our bodies, special molecules called proteins interact in a variety of ways. For example, when a protein folds, its change in structure can regulate the function of another protein or the activity of an enzyme. Enzymes (themselves proteins) catalyze biochemical reactions and are vital to metabolism, which maintains life by harvesting energy for building and supporting the proteins that make up our tissues and organs. Proteins also regulate cell signaling and immune responses. All of these interactions can be seen as networks: protein interaction networks, metabolic networks, gene regulatory networks, and so on. These biological networks exist within a cell. At a higher level, within a body, neural cells form connections (synapses) giving rise to the neural networks that form our brains. And at an even higher level, entire species interact. An animal of one species may see another species as food, creating an ecological network, or food web among species. When we think of this network, ecological balance depends on the availability of species that sustain each other. Removing a node in such a food web — when a species goes extinct, for example — affects the survival of other parts of the ecosystem network. Figure 1.8 illustrates three types of biological networks: a protein interaction network, a neural network, and a food web. They are all essential elements of life on our planet.

Box 1.1 Definition of a network

A network G has two parts, a set of N elements, called *nodes* or *vertices*, and a set of L pairs of nodes, called *links* or *edges*. The link (i, j) joins the nodes i and j. A network can be *undirected* or *directed*. A directed network is also called a *digraph*. In directed networks, links are called *directed links* and the order of the nodes in a link reflects the direction: the link (i, j) goes from the source node i to the target node j. In undirected networks, all links are bi-directional and the order of the two nodes in a link does not matter. A network can be *unweighted* or *weighted*. In a weighted network, links have associated *weights*: the *weighted link* (i, j, w) between nodes i and j has weight w. A network can be both directed and weighted, in which case it has directed, weighted links.

1.2 Basic Network Elements

In very general terms a network, or graph, is a set of elements, which we call *nodes*, along with a set of connections between pairs of nodes, which we call *links*. The links represent the presence of a relation among the elements represented by the nodes. As we have seen earlier, links can corresponds to social, physical, communication, geographic, conceptual, chemical, or biological interactions. We say that two nodes are *adjacent* or *connected* if there is a link between them. It is also common to call connected nodes *neighbors*.

Networks provide a general theoretical framework allowing for a convenient conceptual representation of interrelations in a wide array of systems; we have seen several examples in Section 1.1. The study of networks has a long tradition in mathematics, computer science, sociology, and communication research. Recently, networks have also been studied intensely in physics and biology. Different fields concerned with networks often introduced their own nomenclature. For example, in some fields a network is called a *graph*, a node is referred to as a *vertex* and a links is an *edge*. (We will occasionally use these terms.) The rigorous language for the description of networks is found in graph theory, a field of mathematics that can be traced back to the pioneering work of Leonhard Euler in the 18th century. Here we do not want to provide a rigorous introduction to graph theory. We are mostly interested in building a vocabulary and introducing a set of basic notions that will allow us to take our first steps into the word of networks. However, sometimes a formal notation is helpful. In these cases we will include the formal notation in a shaded area or in a box. For example, a more rigorous definition of a network is provided in Box 1.1. In the following chapters we will introduce additional concepts and definitions as needed to analyze real-world systems.

Each network is characterized by the total number of nodes N and the total

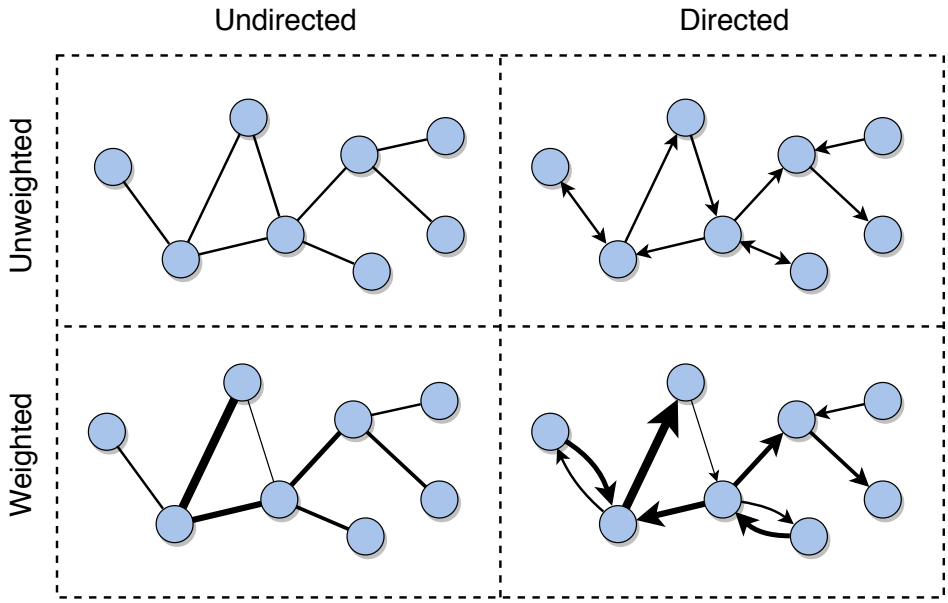


Fig. 1.9 Graphical representations of undirected, directed, and weighted networks. The circles represent the nodes. Pairs of adjacent nodes are connected by a line (link) or arrow (directed link). Arrows indicate the direction of the links. The thickness of a link represents its weight in weighted networks.

number of links L. We call N the *size* of the network because it identifies the number of distinct elements composing the system. The numbers of nodes and links do not suffice in defining a network; we have to specify the way in which the nodes are connected by the links. There are also different types of links, which define different classes of networks. In some networks, such as Facebook (Figure 1.1), the links do not have a direction and we represent then as line segments. We call these networks *undirected*. In other cases, such as Wikipedia (Figure 1.5), links are directed and we represent them as arrows. Networks with directed links are called *directed networks*. In some networks, such as food webs (Figure 1.8), links have associated weights. These are called *weighted networks*. A network can be both directed and weighted. The email network is an example of weighted directed network, in which link weights and directions represent communication traffic (number of messages) between nodes. Figure 1.9 provides illustrations of directed, undirected, and weighted networks.

There are many other classes of networks. A network might have more than one type of nodes. For example, the movie-star network (Figure 1.2) has two types of nodes representing movies and people. In this network, a link connects an actor or actress to a movie, but there are no links among people or among movies. This is an instance of a so-called *bipartite network*. In a bipartite network, there are two groups of nodes such that links only connect nodes from different groups and not nodes from

the same group. Other examples of bipartite networks include those that capture the relationships between songs and artists, between classes and students, and between products and customers. A network might also have multiple types of links. To use the movie-star example again, we could imagine adding edges between actors and/or actresses who are married to each other. Different kinds of links are also possible among nodes of the same type. In the example of the Wikipedia (Figure 1.5), in addition to the hyperlinks, we might have weighted links representing clicks from Wikipedia users, and/or undirected links between articles that share editors.

An important category of networks are *networks of networks*. As the name suggests, these are composed of multiple networks with extra links between nodes of different graphs. Consider the electrical power grid, which connects power generating stations and demand centers through high voltage transmission lines. The stations are controlled by computers that monitor and manage the production and transmission of electricity. These computers are connected through the Internet. This is therefore a system of two networks: the power grid and the Internet. One network can affect the other to optimize delivery; the grid can be reconfigured to reroute power when needed. However, this kind of network of networks can also introduce unpredictable vulnerabilities. A software problem or attack can take down one or more nodes in the power grid, and without electricity the Internet in an area could also go down, leading to failures of other nodes and, in an extreme case, a catastrophic domino effect called a *cascading failure* affecting a large portion of the grid. For these reasons, networks of networks are the subject of intense study.

To keep things simple, in this book we focus on networks with a single type of node and a single type of link. In an undirected network, we will assume that there can be at most one link connecting a pair of nodes. (If the network is directed, there can be two links, one in each direction, as shown in Figure 1.9.) In addition, we will not consider *self-loops*, or links connecting a node to itself; we will assume that each link connects two distinct nodes.

1.2.1 Handling Networks in Code

To manage, analyze, and visualize networks with more than a handful of nodes and links, we need to use software tools or write our own code. There are many network analysis and visualization tools, as well as libraries to handle networks in many programming languages. In the appendix we present some of these software tools. Throughout the book we will occasionally mention a couple of these tools. For instance, the visualizations in Section 1.1 are generated with an application called *Gephi*. However, we believe that to get a hands-on understanding of networks it is necessary to "get our hands dirty" and write some code. We assume that students using this book have some familiarity with Python, a popular programming language among both novice and expert coders. To make our life easier, we will use *NetworkX* (`networkx.github.io`), a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of networks. NetworkX

provides data structures, algorithms, measures, and generators for networks, as well as rudimentary visualization facilities.

Once we import NetworkX, we can easily create an undirected network (a "Graph") and add a few nodes and links. Nodes are referred by integer IDs and links are called edges:

```
import networkx as nx # always import NetworkX first!
G = nx.Graph()
G.add_node(1)
G.add_node(2)
G.add_edge(1,2)
```

We can add several nodes or links at once:

```
G.add_nodes_from([3,4,5,...])
G.add_edges_from([(3,4),(3,5),...])
```

Here is how we get lists of nodes, links, and neighbors of a given node:

```
G.nodes()
G.edges()
G.neighbors(3)
```

And here is how you loop over nodes or links:

```
for n in G.nodes:
    print(n, G.neighbors(n))
for u,v in G.edges:
    print(u, v)
```

Similarly, we can create a directed network ("DiGraph"):

```
D = nx.DiGraph()
D.add_edge(1,2)
D.add_edge(2,1)
D.add_edges_from([(2,3),(3,4),...])
```

Note that the link from node 1 to node 2 is distinct from the link from 2 to 1 because this network is directed. Also note that when we add a link, the nodes are added automatically if they don't already exist. This is convenient. There are functions for getting the size and number of links:

```
D.number_of_nodes()
D.number_of_edges()
```

In a directed network, when we ask for the neighbors of a node, we get both the nodes linking to and from it. But there are also functions to only get the edges linking to and from, respectively called predecessors and successors:

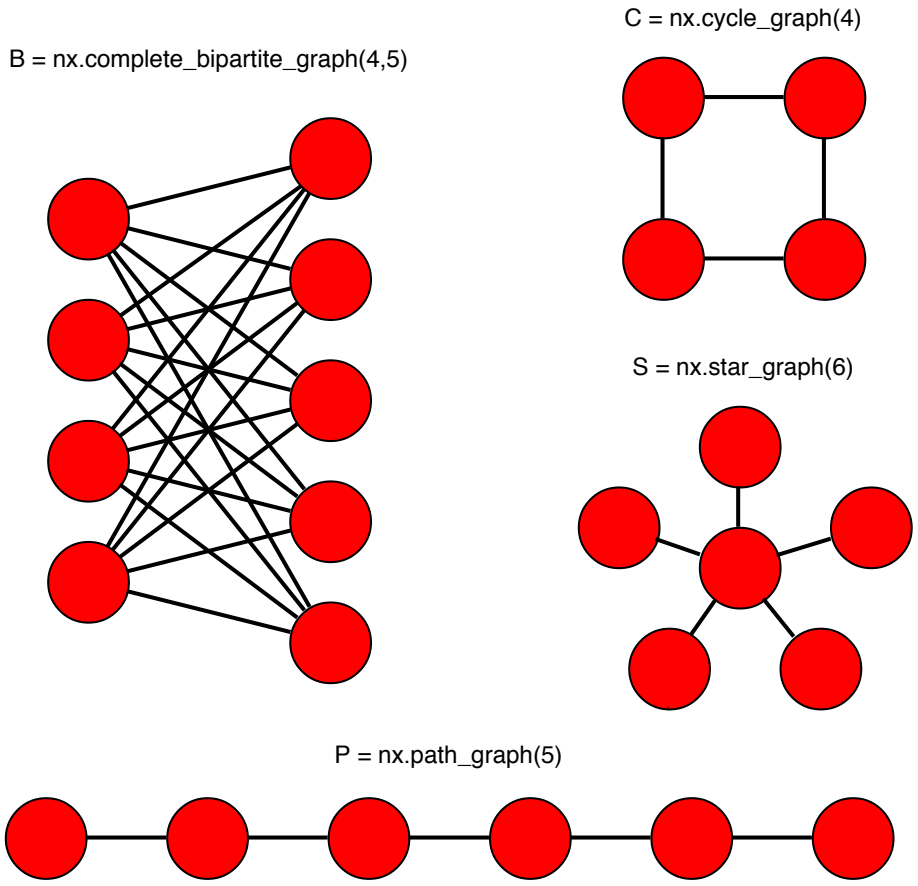


Fig. 1.10 A few simple networks generated by NetworkX functions: bipartite (B), cycle (C), star (S) and path (P).

```
D.neighbors(2)
D.predecessors(2)
D.successors(2)
```

Finally, there are functions to generate networks of many types. Typically these functions need arguments that specify the number of nodes or links. Here is code to generate a few networks, shown in Figure 1.10:

```
B = nx.complete_bipartite_graph(4,5)
C = nx.cycle_graph(4)
P = nx.path_graph(5)
S = nx.star_graph(6)
```

We strongly recommend that you read the NetworkX tutorial[2] and bookmark its documentation.[3] And remember, Google and Stack Overflow are your friends when you are stuck!

1.2.2 Density and Sparsity

The maximum number of links in a network is bounded by the possible number of distinct connections among the nodes of the system. The maximum number of links is therefore given by the number of pairs of nodes. A network with the maximum number of links, in which all possible pairs of nodes are connected by links, is called a *complete network*.

The maximum number of links in an undirected network with N nodes is the number of distinct pairs of nodes:

$$L_{max} = \binom{N}{2} = N(N-1)/2. \tag{1.1}$$

Intuitively each node can connect to $N-1$ other nodes, and there are N of them. However, that would count each pair twice, so we divide by two. In a directed network, each pair of nodes should be counted twice, one for each direction, so $L_{max} = N(N-1)$. Counting the possible pairs of objects among a set of N objects is something that we will encounter again later in the book. Mathematicians have a name for the formula $\binom{N}{2}$: "N choose two."

The fraction of possible links that actually exist, which is the same as the fraction of pairs of nodes that are actually connected, is called the *density* of the network. A complete network has maximal density. However, the actual number of links is typically much smaller than the maximum, as most pairs of nodes are not directly connected to each other. Therefore the density is often much, much smaller than one — by orders of magnitude in most real-world, large networks. This is an important feature that helps in dealing with network structure. We call it *sparsity*. Intuitively the less edges are in a network, the sparser it is.

The density of a network with N nodes and L links is

$$d = L/L_{max}. \tag{1.2}$$

In an undirected network this is given by

$$d = L/L_{max} = \frac{2L}{N(N-1)} \tag{1.3}$$

and in a directed network the density is

$$d = L/L_{max} = \frac{L}{N(N-1)}. \tag{1.4}$$

[2] `networkx.github.io/documentation/stable/tutorial.html`
[3] `networkx.github.io/documentation/stable/`

Table 1.1 Basic statistics of network examples. Network types can be (D)irected and/or (W)eighted. When there is no label the network is undirected and unweighted. For directed networks, we provide the average in-degree (which coincides with the average out-degree).

Network	Type	Nodes (N)	Links (L)	Density (d)	Average degree ($\langle k \rangle$)
Facebook Northwestern Univ.		10,567	488,337	0.009	92.4
IMDB movies and stars		683,515	985,410	0.000004	2.9
Twitter US politics	DW	18,470	48,365	0.0001	2.6
Enron Email	DW	36,692	367,662	0.0003	10.0
Wikipedia math	D	15,220	194,103	0.0008	12.8
Internet routers	W	190,914	607,610	0.00003	6.4
US air transportation		546	2,781	0.02	10.2
World air transportation		3,179	18,617	0.004	11.7
Yeast protein interactions		1,870	2,277	0.001	2.4
C. elegans brain	DW	297	2,345	0.03	7.9
Everglades ecological food web	DW	69	916	0.2	13.3

In a complete network, $d = 1$ by definition. In a sparse network, $L \ll L_{max}$ and therefore $d \ll 1$. When a network grows very large, we can observe how the number of links increases as a function of the number of nodes. We say that the network is sparse if the number of links grows proportionally to the number of nodes ($L \sim N$), or even slower. If instead the number of links grows faster, e.g., quadratically with network size ($L \sim N^2$), then we can call the network dense.

To illustrate the importance of network sparsity, let us consider the example of Facebook. At the time when this book is being written, Facebook has around 2 billion users ($N \approx 2 \times 10^9$). If this was a complete network, there would be $L \approx 10^{18}$ links — that is a number with 18 zeroes, and there is no way to store so much data! But fortunately social networks are very sparse and Facebook is no exception. Each user has on average a thousand friends or less, so that the density is approximately $d \approx 10^{-6}$. That is still a lot of data, but Facebook can manage it.

Table 1.1 presents basic statistics about size and density of the network examples illustrated in Section 1.1. Although these networks are very different from each other, they are all sparse.

NetworkX makes it easy to measure the density of directed and undirected networks:

```
nx.density(G)
nx.density(D)
CG = nx.complete_graph(8471) # a large complete network
print(nx.density(CG))        # no need for a calculator!
```

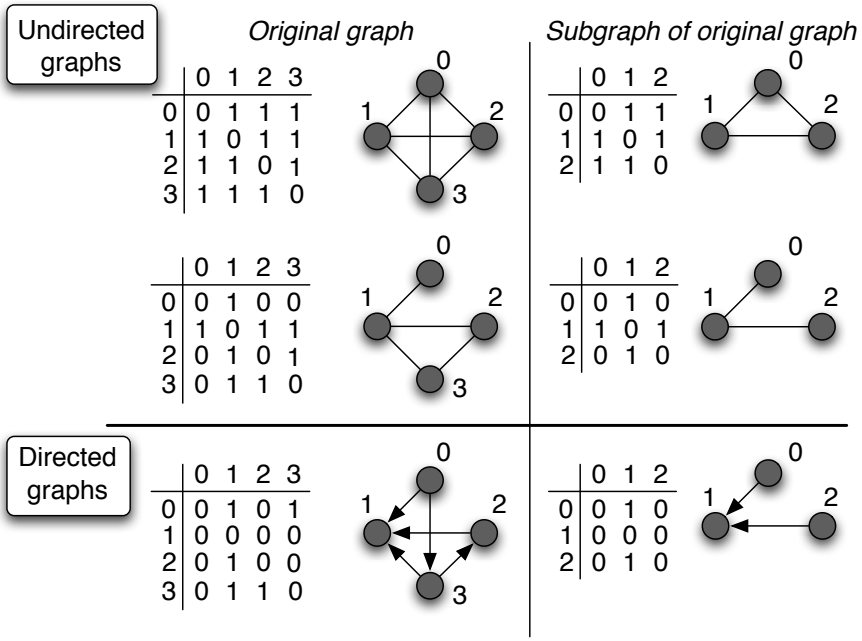


Fig. 1.11 Networks and subnetwork examples. We also show the adjacency matrix representation of each network (see Section 1.2.8).

1.2.3 Subnetworks

In many cases, we are interested in a subset of a network, which is itself a network and is called a *subnetwork* (or *subgraph*). A subnetwork is obtained by selecting a subset of the nodes and *all* of the links among these nodes.

Figure 1.11 provides some illustrations of subnetworks of undirected and directed networks. The abundance of certain types of subnetworks and their properties are important in the characterization of real networks. As an example, a *clique* is a complete subnetwork: a subset of nodes all linked to each other. Any subnetwork of a complete network is a clique.

Using NetworkX we can generate a subnetwork of a given network by specifying a subset of nodes:

```
K5 = nx.complete_graph(5)
clique = nx.subgraph(K5, (0,1,2))
```

1.2.4 Connected networks

We saw that the number of links is bound by the number of nodes. This is an upper bound; there is no lower bound, as a network might have no links at all,

uninteresting as that may be. As we shall see in Chapter 6, the higher the density, the greater the chances that the network is *connected*, i.e., that you can reach any node from any other node by traversing links. The fewer the links and lower the density, the higher the chances that the network is disconnected, so that there are multiple nodes or group of nodes that are not reachable from each other. Network connectivity is discussed in greater detail in Chapter 2.

NetworkX has algorithms to determine whether a network is connected. For example, the networks in Figure 1.10 are all connected:

```
K4 = nx.complete_graph(4)
nx.is_connected(K4)          # true
C = nx.cycle_graph(4)
nx.is_connected(C)           # true
P = nx.path_graph(5)
nx.is_connected(P)           # true
S = nx.star_graph(6)
nx.is_connected(S)           # true
```

1.2.5 Degree

The *degree* of a node is its number of links, or of neighbors. We use k_i to denote the degree of node i.

$$k_i = \sum_j A_{ij} \tag{1.5}$$

where the sum runs over all nodes and $A_{ij} = 1$ if i and j are adjacent, $A_{ij} = 0$ otherwise.

A node with no neighbors has degree zero ($k = 0$) and is called a *singleton*.

The average degree of a network is denoted by $\langle k \rangle$. It is an important property and is related (directly proportional) to its density.

$$\langle k \rangle = \frac{k_1 + k_2 + \cdots + k_{N-1} + k_N}{N} = \frac{\sum_i k_i}{N}. \tag{1.6}$$

Since each link contributes to the degree of two nodes in an undirected network, the numerator of Eq. 1.6 can be written as $2L$. And from the definition of density for an undirected network (Eq. 1.3), $2L = dN(N-1)$. Therefore

$$\langle k \rangle = \frac{2L}{N} = \frac{dN(N-1)}{N} = d(N-1) \tag{1.7}$$

and conversely

$$d = \frac{\langle k \rangle}{N-1}. \tag{1.8}$$

This makes sense: the maximum degree of a node is $k_{max} = N - 1$, obtained when the node is connected to every other node. Intuitively, the density is the ratio between the average and maximum degree.

Table 1.1 shows the average degree of the network examples illustrated in Section 1.1. NetworkX has a function that returns the degree of a given node. Without arguments, it returns a dictionary with the degree of each node:

```
G.degree(2) # returns the degree of node 2
G.degree()  # returns the degree of all nodes of G
```

In Chapter 3 we will see that the degrees of a network's individual nodes are very important properties to characterize the structure of the network. So far we have defined the degree in undirected networks. Next we extend the definition to directed and weighted networks.

1.2.6 Directed Networks

In the graphical representation of a network, the directed nature of the links is depicted by means of an arrow, indicating the direction of each link. The main difference between directed and undirected networks is represented in Figure 1.9. In an undirected network, the presence of a link between two nodes connects the adjacent nodes in both directions. On the other hand, the presence of a link in a directed network does not necessarily imply the presence of a link in the opposite direction. This fact has important consequences for the connectedness of a directed network, as will be discussed in more detail in Chapter 4.

When we consider the degree of a node in a directed network, we have to think of incoming and outgoing links separately. The number of incoming links, or predecessors, is called *in-degree*. And the number of outgoing links, or successors, is called *out-degree*.

The in-degree and out-degree are defined similarly to the degree:

$$k_j^{in} = \sum_i A_{ij} \tag{1.9}$$

$$k_i^{out} = \sum_j A_{ij} \tag{1.10}$$

where $A_{ij} = 1$ if there is a directed link from i to j and $A_{ij} = 0$ otherwise. We already defined the density for a directed network (Eq. 1.4). We can define average in-degree and average out-degree similarly to Eq. 1.6.

NetworkX has functions that return the in-degree and out-degree of a given node. If the network is directed, the `degree` function returns the total degree, which is the sum of in-degree and out-degree:

```
D.in_degree(4)
D.out_degree(4)
D.degree(4)
```

1.2.7 Weighted Networks

In the graphical representation of a network, the weighted nature of the links is depicted by means of lines of different width, indicating the weight of each link. A weight of zero is equivalent to the absence of a link. The main difference between weighted and unweighted networks is represented in Figure 1.9.

A weighted network can be directed or undirected; let us first assume the simpler case of an undirected weighted network. We can measure the degree of a node in a weighted network by disregarding the weights. However, it may be important to consider the weights. We can therefore define the *weighted degree*, or *strength* of a node, as the sum of the weights of its links. Similarly, we can define *in-strength* and *out-strength* in the case of a directed weighted network.

The weighted degree, or strength, of a node i in an undirected weighted network is denoted by:

$$s_i = \sum_j w_{ij} \tag{1.11}$$

where w_{ij} is the weight of the link between nodes i and j. We assume $w_{ij} = 0$ if there is no link between i and j. We can analogously generalize in-degree and out-degree to in-strength and out-strength in a directed weighted network:

$$s_j^{in} = \sum_i w_{ij} \tag{1.12}$$

$$s_i^{out} = \sum_j w_{ij} \tag{1.13}$$

where w_{ij} is the weight of the directed link from i to j.

In NetworkX, both graphs and digraphs can have "weight" attributes attached to links. When adding multiple weighted links, each is specified as a triple where the third element is the weight:

```
W = nx.Graph()
W.add_edge(1,2,weight=6)
W.add_weighted_edges_from([(2,3,3),(2,4,5)])
```

We can get a list of links with their associated weight data, for example if we need to print the links with large weights:

```
for (i,j,w) in W.edges(data='weight'):
```

```
    if w > 3:
        print('(%d,␣%d,␣%d)' % (i,j,w)) # skip link (2,3)
```

Finally, we can get the strength of a given node using the `degree` function and specifying the weight attribute:

```
W.degree(2, weight='weight') # strength of node 2
                             # is 6 + 3 + 5 = 14
```

1.2.8 Network Representations

To store and retrieve a network in/from a computer file, we need a way to formally represent its nodes and links. There are several possible network representations. The simplest is the *adjacency matrix*, an $N \times N$ matrix in which each element represents the link between the nodes indexed by the corresponding row and column. We have already encountered the adjacency matrix in the formal definition of the degree (the A_{ij} in Eqs. 1.5, 1.9 and 1.10). In Figure 1.11 we show the graphical illustrations of different undirected and directed networks and their corresponding adjacency matrices.

For undirected networks the adjacency matrix is symmetric: we can swap rows and columns and the matrix does not change. Therefore half of the matrix contains redundant information. For directed networks, the adjacency matrix is not symmetric. For unweighted networks, the elements take only values one or zero to indicate the presence or absence of a link, respectively. For weighted networks, matrix elements can take any values corresponding to the link weights. We have already encountered the adjacency matrix elements for weighted networks (the w_{ij} in Eqs. 1.11, 1.12 and 1.13).

In NetworkX, we can get and print adjacency matrices and use the matrix representation to get and set link attributes:

```
print(nx.adjacency_matrix(G)) # graph
G.edge[3][4]
G.edge[3][4]['color']='blue'
print(nx.adjacency_matrix(D)) # digraph
D.edge[3][4]
D.edge[4][3] # not the same as the previous one
print(nx.adjacency_matrix(W)) # weighted graph
W.edge[2][3]
W.edge[2][3]['weight'] = 1
```

While the adjacency matrix representation matches the mathematical formalisms of networks, it is not efficient for storing real networks, which are typically large and sparse. The required storage space grows like the square of the network size (N^2), but if the network is sparse, most of this space is wasted storing zeros (non-existing links). With large sparse networks, a more compact network representation

is the *adjacency lists*, a list of the sets of neighbors of each node. Adjacency lists store sparse networks efficiently because the non-existing links are ignored; only the existing links (non-zero values of the adjacency matrix) are considered.

NetworkX provides facilities to loop over a network's adjacency list and retrieve links and their attributes. For example, here is one way to print the degree of each node:

```
for n,neighbors in G.adjacency():
    for number,link_attributes in neighbors.items():
        print('(%d,␣%d)' % (n,number))
```

A third, equally efficient network representation is the *edge list*, which lists each link as a pair of connected nodes. We may also need to list the nodes separately in case of singletons, which would not appear in any of the pairs. In the case of weighted networks, each link is represented as a triple, where the third element is the weight.

In this book we will use the edge list representation to store networks. NetworkX has functions to write and read network files using this representation. You can view the format of an edge list file for yourself:

```
nx.write_edgelist(G, "file.edges")
G2 = nx.read_edgelist("file.edges") # G2 same as G
nx.write_weighted_edgelist(W, "wf.edges") # store weights
with open("wf.edges") as f:
    for line in f:
        print(line)
W2 = nx.read_weighted_edgelist("wf.edges") # W2 same as W
```

1.3 Drawing Networks

We can learn a lot about a network by drawing it and inspecting its graphical representation. This requires a *network layout algorithm* to place each node on a plane. (There are also sophisticated 3-D layouts, but we do not discuss them in this book.) There are many layout algorithms that are appropriate for representing different kinds of networks; for example, we used a *geographic layout* to draw the air transportation network in Figure 1.7. The most popular class of network layout algorithms are *force-directed layout algorithms*, which are used to visualize most of the example networks in Section 1.1. The inset of Figure 1.7 uses a force-directed layout as well.

The goals of a force-directed layout algorithm are to place the nodes so that connected nodes are positioned close to each other, all the links are of similar length, and the number of link crossings is minimized. To get an idea of how force-directed layout works, imagine a force that repels any two nodes from each other,

like the force between two particles with the same electrical charge. Further imagine a spring connecting any two linked nodes, generating an attractive force when they are too far from each other. Force-directed layout algorithms simulate such a physical system so that nodes move to minimize the energy of the system: connected nodes will move toward each other and away from nodes not connected to them.

The result is not only an aesthetically pleasing drawing, but also, sometimes, a visualization of the most obvious communities in the network, as we have seen in Section 1.1. For example, in Figure 1.3, because people in a community (progressive or conservative) are densely connected to each other, they end up clustered together in the layout.

NetworkX has a function to draw a network, which uses a rudimentary network layout algorithm:

```
import matplotlib.pyplot
nx.draw(G)
```

Note that drawing requires a plot interface, such as Matplotlib's. This works reasonably well for small networks with, say, less than a hundred nodes. For larger networks, there are better visualization tools. The examples in Section 1.1 are visualized with Gephi's *ForceAtlas2* layout algorithm.

1.4 Summary

Networks are a general way to model and study complex systems with many interacting elements. We have seen several examples of networks. Nodes can represent many different types of objects from people to Web pages, from proteins to species, from Internet routers to airports. Nodes can have features associated with them beside labels: geographic location, wealth, activity, number of connections, and so on. Links also can represent many different kinds of relationships, from physical to virtual, from chemical to social, from communicative to informative. They can have a direction (like Web hyperlinks and email) or be reciprocal (like marriage). They can all be the same or have different features such as similarity, distance, traffic, volume, weight, and so on.

We have presented some basic definitions and quantities that allow us to describe a network:

1 A network is made of two set of elements: the nodes and links connecting pairs of nodes.
2 In directed networks, links have a direction. There may be a link from node 1 to node 2, and not necessarily one from 2 to 1. In undirected networks, links are reciprocal.

3 In weighted networks, links have associated weights that represent connection attributes like importance, similarity, distance, traffic, etc. In unweighted networks, all links are the same.

4 The density of a network is the fraction of node pairs that are connected. A network is complete if all pairs of nodes are connected, so that the density is one. Most real networks are sparse, meaning that they have very small density.

5 The degree of a node is the number of neighbors. In directed networks, nodes have in-degree and out-degree measuring the number of incoming and outgoing links, respectively. If the network is weighted, the strength of a node is the sum of the weights of its links. Weighted directed networks have in-strength and out-strength.

6 Adjacency lists and edge lists are efficient representations to store sparse networks.

7 NetworkX is a popular and convenient programming library to code networks in the Python language.

The definitions in this chapter form a basic vocabulary for network science. More quantities and properties will be introduced in future chapters so that we can describe, analyze, and model real networks and learn what they tell us about the underlying systems and phenomena.

1.5 Further Readings

There are several other excellent textbooks on network science to go beyond the introductory material in this book. Caldarelli and Chessa (2016) dig a bit deeper into the data science of several case studies. If you are interested in branching into physics, consider the textbook by Barabási (2016); or if you want to explore the connections to economics and sociology, we recommend the textbook by Easley and Kleinberg (2010). For more advanced physics, math, and social science topics, there are many books to choose from (Newman, 2010; Caldarelli, 2007; Barrat et al., 2008; Bollobás, 2012; Dorogovtsev and Mendes, 2013; Wasserman and Faust, 1994b; Latora et al., 2017).

The use of networks to graphically representing social relationships among individuals was introduced by Moreno and Jennings (1934), who called these social networks *sociograms*.

Much more recently, studies have shown that online social network can reveal a person's sexual orientation (Jernigan and Mistree, 2009) and facilitate highly effective phishing attacks (Jagatic et al., 2007). Conover et al. (2011b) showed that political information diffusion networks on Twitter are very polarized and segregated. As a result we can predict the political leaning of most users with high

accuracy by starting with a few node labels and propagating them through network neighbors (Conover et al., 2011a).

You can read about the vision, design, and history of the Web in a book coauthored by its inventor (Berners-Lee and Fischetti, 2000).

Spring et al. (2002) explain how probes are used to measure the topology of the Internet. Achlioptas et al. (2009) show that these approaches have sampling bias. Computer scientists analyze the structure of routers and autonomous system networks to develop models called "topology generators," which can help in the design of these networks (Rossi et al., 2013). To learn more about Internet networks we recommend the book by Pastor-Satorras and Vespignani (2007).

Data about the yeast protein interaction network is from Jeong et al. (2001). C. elegans neural network data is by White et al. (1986). To learn about the human brain network, also known as "connectome," we recommend the book by Sporns (2012). The Everglades ecological network is derived from Ulanowicz and DeAngelis (1998). To learn more about food webs we refer to Dunne et al. (2002); Melián and Bascompte (2004).

Gao et al. (2012) analyze networks of networks. Catastrophic failure in these networks is discussed by Reis et al. (2014); Radicchi (2015).

Data for several of the real-world network examples shown in this book is provided by the Network Repository (Rossi and Ahmed, 2015). The visualizations are done using Gephi (Bastian et al., 2009) with the ForceAtlas2 layout algorithm (Jacomy et al., 2014). Force-directed network layout algorithms were introduced by Eades (1984) and improved by Kamada and Kawai (1989); Fruchterman and Reingold (1991).

Exercises

1.1 Go through the Chapter 1 Tutorial on the book's Github repository.[4]

1.2 Consider any arbitrary network. Given a single link, what is the maximum number of nodes that link can connect? Given a single node, what is the maximum number of links that can connect to that node?

1.3 Consider the road map in Figure 1.12. If one were creating a network representation of traffic patterns, which of the following would be the best choice to make up the links of the network? (Hint: your answer to the next question may inform your answer to this question, and vice-versa.)

- **a.** Pedestrians traveling along the streets
- **b.** Road segments, *e.g.,* 5th Ave. between 12th and 13th streets
- **c.** Entire roads, *e.g.,* 5th Ave.
- **d.** Vehicles traveling on the roads

[4] `github.com/CambridgeUniversityPress/IntroductoryNetworkScience`

1.4 Consider the road map in Figure 1.12. In a network representation of traffic patterns, which of the following would be the best choice to make up the nodes of the network? (Hint: your answer to the previous question may inform your answer to this question, and vice-versa.)

a. City blocks, *e.g.,* the block between 5th-6th avenues and 12th-13th streets
b. Street intersections, *e.g.,* 5th Ave. and 12th St.
c. Pedestrians moving along the streets
d. Vehicles traveling on the roads

1.5 Consider the road map in Figure 1.12. The grid-like structure of this network means that most nodes have the same degree. What is the most common degree for nodes in this network?

1.6 Consider the US air transportation network shown in Figure 1.7. What are the nodes in this network? What are the links?

1.7 Compare the US air transportation network in Figure 1.7 with the Manhattan road map in Figure 1.12. The air transportation network displays a distinguishing feature that the Manhattan road network lacks. What is this key characteristic?

a. Singleton nodes with no links
b. Multiple routes between nodes
c. Nodes with more than one connected link
d. Hub nodes with many links

1.8 Consider the road map in Figure 1.12. Manhattan has a lot of one-way streets. This implies that a good network model of traffic flow would probably have directed links. Consider a subgraph of this network with grid-like connectivity and all one-way streets (*i.e.,* each node is a 4-way intersection of two one-way streets). What is the most common in-degree of nodes in this subgraph? What is the most common out-degree?

1.9 If we wanted to represent the volume of traffic flow between each pair of intersections in the Manhattan traffic network (Figure 1.12), how would we include this data in such a graph?

1.10 Consider a directed network of N nodes. Now consider the total in-degree, *i.e.,* the sum of the in-degree over all nodes in the network. Compare this to the analogous total out-degree. Which of the following must hold true for any such network?

a. Total in-degree must be less than total out-degree
b. Total in-degree must be greater than total out-degree
c. Total in-degree must be equal to total out-degree
d. None of these hold true in all instances

1.11 In a social graph from Facebook, which type of link best represents the "friend" relation? Directed or undirected?

1.12 In a social graph from Twitter, which type of link best represents the "follower" relation? Directed or undirected?

1.13 Consider a Twitter retweet network, where users are nodes and we want to show how many times a given user has retweeted another user. What link type best captures this relation?

a. Undirected, unweighted
b. Undirected, weighted
c. Directed, unweighted
d. Directed, weighted

1.14 Consider a hashtag co-occurrence graph from Twitter. In this network, hashtags are the nodes, and a link between two hashtags show how often those two hashtags appear in tweets together. What link type would best capture this relation?

a. Undirected, unweighted
b. Undirected, weighted
c. Directed, unweighted
d. Directed, weighted

1.15 Consider a network created from characters in a story or play. The nodes are people, and a link exists between two nodes if those characters ever engage in dialogue. Which type of edge would best represent this relation?

a. Undirected, unweighted
b. Undirected, weighted
c. Directed, unweighted
d. Directed, weighted

1.16 Suppose we want to make a more complex version of dialog network that captures how much each character speaks and to whom. What type of link would best represent this relation?

a. Undirected, unweighted
b. Undirected, weighted
c. Directed, unweighted
d. Directed, weighted

1.17 Imagine that your social network has a subnetwork where you and 24 of your friends (25 people total) are all friends with each other. What is such a subnetwork called? And how many links are contained in the subnetwork?

1.18 Consider an undirected, connected network with N nodes. What is the maximum number of links this network can have? If we do not require that the network be connected, does that maximum number of links change?

1.19 Consider a bipartite network of N nodes, N_1 nodes of type 1 and N_2 nodes of type 2 (so that $N_1 + N_2 = N$). What is that maximum number of links in this network?

1.20 Given a complete network A with N nodes, and a bipartite network B also with N nodes, which of the following holds true for any $N > 2$:

a. Network A has more links than network B
b. Network A has the same number of links as network B
c. Network A has fewer links than network B
d. None of these hold true for all such $N > 2$

1.21 Recall that in a complete network there exists a link between each pair of nodes. We know a complete undirected network of N nodes has $N(N-1)/2$ links. Must any undirected network of N nodes and $N(N-1)/2$ edges be complete? Explain why or why not.

1.22 Consider this adjacency matrix:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 1 & 3 & 1 & 1 & 0 \end{pmatrix} \end{array} \tag{1.14}$$

An entry in the ith row and jth column indicates the weight of the link from node i to node j. For instance, the entry in the 2nd row and 3rd column is 2, meaning the weight of the link from node B to node C is 2. What kind of network does this matrix represent?

a. Undirected, unweighted
b. Undirected, weighted
c. Directed, unweighted
d. Directed, weighted

1.23 Consider the network defined by the adjacency matrix in Eq. 1.14. How many nodes are in this network? How many links? Are there any self-loops?

1.24 Consider the network defined by the adjacency matrix in Eq. 1.14. Is this network strongly connected? Is it weakly connected?

1.25 Consider the network defined by the adjacency matrix in Eq. 1.14. Are there any nodes with outgoing links to every other node? If so, which nodes? Are there any nodes with in-links from every other node? If so, which nodes?

1.26 Consider the network defined by the adjacency matrix in Eq. 1.14. A *sink* is defined as a node with in-links but no out-links. Which nodes in the network, if any, have this property?

1.27 Consider the network defined by the adjacency matrix in Eq. 1.14. What is the in-strength of node C?

1.28 Convert the network defined by the adjacency matrix in Eq. 1.14 to an undirected, unweighted graph. (When converting a directed graph to an undirected one, nodes i and j are connected in the undirected graph if there is a directed link from i to j, or from j to i, or both.) You may want to print out the resulting matrix and/or draw a network diagram for reference. How many nodes are in this converted network? How many links?

1.29 Consider the unweighted, undirected version of the network defined by the adjacency matrix in Eq. 1.14. What is the minimum degree in this network? What is the maximum degree? What is the mean degree? What is the density?

1.30 Imagine two different undirected networks, each with the same number of nodes and links. Must both networks have the same maximum and minimum degree? Explain why or why not. Must they have the same mean degree? Explain why or why not.

1.31 Netflix keeps data on customer preferences using a big bipartite network connecting users to titles they have watched and/or rated. Netflix's movie library contains approximately 100k titles if you count streaming and DVD-by-mail. In 4Q 2013, Netflix reported having about 33 million users. Assume the average user's degree in this network is 1000. Approximately how many links are in this network? Would you consider this network sparse or dense? Explain.

Fig. 1.12 Surface-level road map of upper Manhattan from the 1920s.