Exercises

1.7 Six degrees of separation.[1] [2] (Complexity, Computation) ④

One of the more popular topics in random network theory is the study of how connected they are. ‘Six degrees of separation’ is the phrase commonly used to describe the interconnected nature of human acquaintances: various somewhat uncontrolled studies have shown that any random pair of people in the world can be connected to one another by a short chain of people (typically around six), each of whom knows the next fairly well. If we represent people as nodes and acquaintanceships as neighbors, we reduce the problem to the study of the relationship network.

Many interesting problems arise from studying properties of randomly generated networks. A network is a collection of *nodes* and *edges*, with each edge connected to two nodes, but with each node potentially connected to any number of edges (Fig. 1.5). A random network is constructed probabilistically according to some definite rules; studying such a random network usually is done by studying the entire ensemble of networks, each weighted by the probability that it was constructed. Thus these problems naturally fall within the broad purview of statistical mechanics.

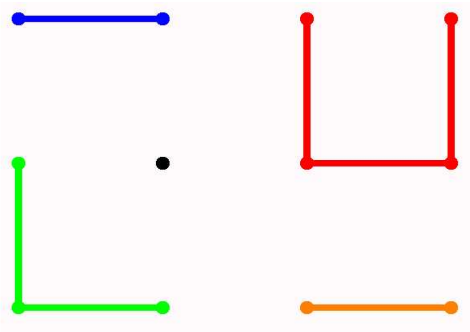

Fig. 1.5 Network. A network is a collection of nodes (circles) and edges (lines between the circles).

In this exercise, we will generate some random networks, and calculate the distribution of distances between pairs of points. We will study *small world networks* [142,95], a theoretical model that suggests how a small number of shortcuts (unusual international and intercultural friendships) can dramatically shorten the typical chain lengths. Finally, we will study how a simple, universal scaling behavior emerges for large networks with few shortcuts.

In the computer exercises section on the book web site [129], you will find some hint files and graphic routines to facilitate working this exercise. We plan to support a variety of languages and systems.

Constructing a small world network. The L nodes in a small world network are arranged around a circle. There are two kinds of edges. Each node has Z short edges connecting it to its nearest neighbors around the circle (up to a distance $Z/2$). In addition, there are $p \times L \times Z/2$ shortcuts added to the network, which connect nodes at random (see Fig. 1.6). (This is a more tractable version [95] of the original model [142], which rewired a fraction p of the $LZ/2$ edges.)

(a) *Define a network object on the computer. For this exercise, the nodes will be represented by integers. Implement a network class, with five functions:*

(1) `HasNode(node)`, *which checks to see if a node is already in the network;*

(2) `AddNode(node)`, *which adds a new node to the system (if it is not already there);*

(3) `AddEdge(node1, node2)`, *which adds a new edge to the system;*

(4) `GetNodes()`, *which returns a list of existing nodes; and*

[1] From *Statistical Mechanics: Entropy, Order Parameters, and Complexity* by James P. Sethna, copyright Oxford University Press, 2007, page 9. A pdf of the text is available at pages.physics.cornell.edu/sethna/StatMech/ (select the picture of the text). Hyperlinks from this exercise into the text will work if the latter PDF is downloaded into the same directory/folder as this PDF.

[2] This exercise and the associated software were developed in collaboration with Christopher Myers.

(5) `GetNeighbors(node)`, *which returns the neighbors of an existing node.*

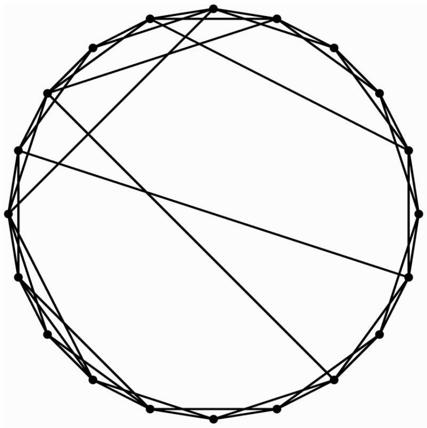

Fig. 1.6 Small world network *with $L = 20$, $Z = 4$, and $p = 0.2$.*[3]

Write a routine to construct a small world network, which (given L, Z, and p) adds the nodes and the short edges, and then randomly adds the shortcuts. Use the software provided to draw this small world graph, and check that you have implemented the periodic boundary conditions correctly (each node i should be connected to nodes $(i - Z/2) \bmod L, \ldots, (i + Z/2) \bmod L$).[4]

Measuring the minimum distances between nodes. The most studied property of small world graphs is the distribution of shortest paths between nodes. Without the long edges, the shortest path between i and j will be given by hopping in steps of length $Z/2$ along the shorter of the two arcs around the circle; there will be no paths of length longer than L/Z (half-way around the circle), and the distribution $\rho(\ell)$ of path lengths ℓ will be constant for $0 < \ell < L/Z$. When we add shortcuts, we expect that the distribution will be shifted to shorter path lengths.

(b) *Write the following three functions to find and analyze the path length distribution.*

(1) `FindPathLengthsFromNode(graph, node)`, *which returns for each* `node2` *in the graph the shortest distance from* `node` *to* `node2`*. An efficient algorithm is a* breadth-first traversal *of the graph, working outward from* `node` *in shells. There will be a* `currentShell` *of nodes whose distance will be set to ℓ unless they have already been visited, and a* `nextShell` *which will be considered after the current one is finished (looking sideways before forward, breadth first), as follows.*

- *Initialize $\ell = 0$, the distance from* `node` *to itself to zero, and* `currentShell = [node]`*.*
- *While there are nodes in the new* `currentShell`*:*
  - *start a new empty* `nextShell`*;*
  - *for each neighbor of each node in the current shell, if the distance to* `neighbor` *has not been set, add the node to* `nextShell` *and set the distance to $\ell + 1$;*
  - *add one to ℓ, and set the current shell to* `nextShell`*.*
- *Return the distances.*

This will sweep outward from `node`, measuring the shortest distance to every other node in the network. (Hint: Check your code with a network with small N and small p, comparing a few paths to calculations by hand from the graph image generated as in part (a).)

(2) `FindAllPathLengths(graph)`, *which generates a list of all lengths (one per pair of nodes in the graph) by repeatedly using* `FindPathLengthsFromNode`*. Check your function by testing that the histogram of path lengths at $p = 0$ is constant for $0 < \ell < L/Z$, as advertised. Generate graphs at $L = 1000$ and $Z = 2$ for $p = 0.02$ and $p = 0.2$; display the circle graphs and plot the histogram of path lengths. Zoom in on the histogram; how much does it change with p? What value of p would you need to get 'six degrees of separation'?*

(3) `FindAveragePathLength(graph)`, *which computes the mean $\langle \ell \rangle$ over all pairs of nodes. Compute ℓ for $Z = 2$, $L = 100$, and $p = 0.1$*

[3] There are seven new shortcuts, where $pLZ/2 = 8$; one of the added edges overlapped an existing edge or connected a node to itself.

[4] Here $(i - Z/2)$ mod L is the integer $0 \le n \le L - 1$ which differs from $i - Z/2$ by a multiple of L.

a few times; your answer should be around $\ell = 10$. Notice that there are substantial statistical fluctuations in the value from sample to sample. Roughly how many long bonds are there in this system? Would you expect fluctuations in the distances?

(c) *Plot the average path length between nodes $\ell(p)$ divided by $\ell(p = 0)$ for $Z = 2$, $L = 50$, with p on a semi-log plot from $p = 0.001$ to $p = 1$.* (Hint: Your curve should be similar to that of with Watts and Strogatz [142, fig. 2], with the values of p shifted by a factor of 100; see the discussion of the continuum limit below.) *Why is the graph fixed at one for small p?*

Large N and the emergence of a continuum limit. We can understand the shift in p of part (c) as a continuum limit of the problem. In the limit where the number of nodes N becomes large and the number of shortcuts $pLZ/2$ stays fixed, this network problem has a nice limit where distance is measured in radians $\Delta\theta$ around the circle. Dividing ℓ by $\ell(p = 0) \approx L/(2Z)$ essentially does this, since $\Delta\theta = \pi Z \ell / L$.

(d) *Create and display a circle graph of your geometry from part (c) ($Z = 2$, $L = 50$) at $p = 0.1$; create and display circle graphs of Watts and Strogatz's geometry ($Z = 10$, $L = 1000$) at $p = 0.1$ and $p = 0.001$. Which of their systems looks statistically more similar to yours? Plot (perhaps using the scaling collapse routine provided) the rescaled average path length $\pi Z \ell / L$ versus the total number of shortcuts $pLZ/2$, for a range $0.001 < p < 1$, for $L = 100$ and 200, and for $Z = 2$ and 4.*

In this limit, the average bond length $\langle \Delta\theta \rangle$ should be a function only of M. Since Watts and Strogatz [142] ran at a value of ZL a factor of 100 larger than ours, our values of p are a factor of 100 larger to get the same value of $M = pLZ/2$. Newman and Watts [99] derive this continuum limit with a renormalization-group analysis (Chapter 12).

(e) *Real networks. From the book web site [129], or through your own research, find a real network[5] and find the mean distance and histogram of distances between nodes.*

In the small world network, a few long edges are crucial for efficient transfer through the system (transfer of information in a computer network, transfer of disease in a population model, ...). It is often useful to measure how crucial a given node or edge is to these shortest paths. We say a node or edge is 'between' two other nodes if it is along a shortest path between them. We measure the 'betweenness' of a node or edge as the total number of such shortest paths passing through it, with (by convention) the initial and final nodes included in the 'between' nodes; see Fig. 1.7. (If there are K multiple shortest paths of equal length between two nodes, each path adds $1/K$ to its intermediates.) The efficient algorithm to measure betweenness is a depth-first traversal quite analogous to the shortest-path-length algorithm discussed above.

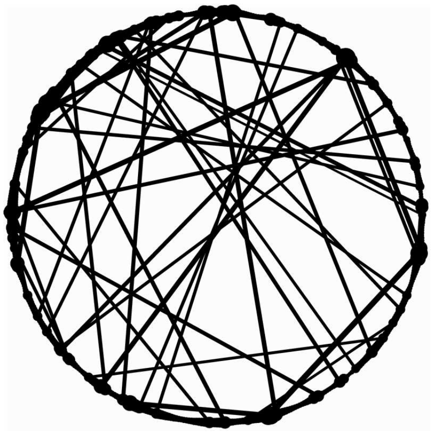

Fig. 1.7 Betweenness Small world network with $L = 500$, $K = 2$, and $p = 0.1$, with node and edge sizes scaled by the square root of their betweenness.

(f) *Betweenness (advanced). Read [96, 46], which discuss the algorithms for finding the betweenness. Implement them on the small world network, and perhaps the real world network you analyzed in part (e). Visualize your answers by using the graphics software provided on the book web site [126].*

[5] Examples include movie-actor costars, 'Six degrees of Kevin Bacon', or baseball players who played on the same team.