Project is: Problems 15.1 + 15.2

Chapter 15

Complexity

We introduce cellular automata models, neural networks, genetic algorithms, and explore the concepts of self-organization and complexity.

15.1 Cellular Automata

Part of the fascination of physics is that it allows us in many cases to reduce natural phenomena to a few simple laws. It is perhaps even more fascinating to think about how a few simple laws can produce the enormously rich behavior that we see in nature. In this chapter we will discuss several models that illustrate some of the new ideas that are emerging from the study of "complex systems."

The first class of models we discuss are known as *cellular automata*. Cellular automata were originally introduced by von Neumann and Ulam in 1948 as an idealization of biological self-reproduction, and are examples of discrete dynamical systems that can be simulated exactly on a digital computer. A cellular automaton can be thought of as a checkerboard with colored squares (the cells). Each cell changes its color at the tick of an external clock according to a rule based on the present configuration (microstate) of the cells in its neighborhood.

More formally, cellular automata are mathematical idealizations of dynamical systems in which space and time are discrete and the quantities of interest have a finite set of discrete values that are updated according to a local rule. The important characteristics of cellular automata include the following:

1. Space is discrete, and there is a regular array of sites (cells). Each site has a finite set of values.
2. Time is discrete, and the value of each site is updated in a sequence of discrete time steps.
3. The rule for the new value of a site depends only on the values of a *local* neighborhood of sites near it.

t:	111	110	101	100	011	010	001	000
t + 1:	0	1	0	1	1	0	1	0

Figure 15.1: Example of a local rule for the time evolution of a one-dimensional cellular automaton. The variable at each site can have values 0 or 1. The top row shows the $2^3 = 8$ possible combinations of three sites. The bottom row gives the value of the central site at the next time step. This rule is termed 01011010 in binary notation (see the second row), the modulo-two rule, or rule 90. Note that 90 is the base ten (decimal) equivalent of the binary number 01011010, that is, $90 = 2^1+2^3+2^4+2^6$.

4. The variables at each site are updated *simultaneously* ("synchronously") based on the values of the variables at the previous time step.

Because the original motivation for studying cellular automata was their biological aspects, the lattice sites frequently are referred to as cells. More recently, cellular automata have been applied to a wide variety of physical systems ranging from fluids to galaxies. We will refer to sites rather then cells, except when we are explicitly discussing biological systems.

We first consider one-dimensional cellular automata with the neighborhood of a given site assumed to be the site itself and the sites immediately to the left and right of it. Each site also is assumed to have two states (a Boolean automata). An example of such a rule is illustrated in Fig. 15.1, where we see that a rule can be labeled by the binary representation of the update for each of the eight possible neighborhoods and by the base ten equivalent of the binary representation. Because any eight digit binary number specifies an one-dimensional cellular automata, there are $2^8 = 256$ possible rules.

`Program ca1` takes as input the decimal representation of the rule and produces the rule matrix (array `update`). This array is used to update each site on the lattice using periodic boundary conditions. On a single processor computer, it is necessary to use an additional array so that the state of each site can be updated using the previous values of the sites in its local neighborhood. The state of the sites as a function of time is shown on the screen with time running downwards.

```
PROGRAM ca1
! one-dimensional Boolean cellular automata
DIM update(0 to 7),site(0 to 501)
CALL setrule(update())
CALL initial(site(),L,tmax,#2)
CALL iterate(site(),L,update(),tmax,#2)
END

SUB setrule(update())
    INPUT prompt "rule number = ": rule
    OPEN #1: screen 0,0.5,0.2,0.8
    SET BACKGROUND COLOR "black"
    SET COLOR "white"
    FOR i = 7 to 0 step -1
```

```
        LET update(i) = int(rule/2^i)  ! find binary representation
        LET rule = rule - update(i)*2^i
        LET bit2 = int(i/4)
        LET bit1 = int((i - 4*bit2)/2)
        LET bit0 = i - 4*bit2 - 2*bit1
        ! show possible neighborhoods
        PRINT using "#": bit2,bit1,bit0;
        PRINT "  ";
    NEXT i
    PRINT
    FOR i = 7 to 0 step -1
        PRINT using "##": update(i);   ! print rules
        PRINT "   ";
    NEXT i
    CLOSE #1
END SUB

SUB initial(site(),L,tmax,#2)
    RANDOMIZE
    OPEN #2: screen 0.5,1,0.1,0.9
    ASK PIXELS px,py
    SET WINDOW 1,px,py,1
    SET COLOR "yellow"
    LET L = 2*int(px/8) - 8
    LET tmax = L
    LET site(L/2) = 1             ! center site
    BOX AREA 1+2*L,2*L+4,1,4      ! each site 4 x 4 pixels
END SUB

SUB iterate(site(),L,update(),tmax,#2)
    ! update lattice
    ! need to introduce additional array, sitenew, to temporarily
    ! store values of newly updated sites
    DIM sitenew(0 to 501)
    FOR t = 1 to tmax
        FOR i = 1 to L
            LET index = 4*site(i-1) + 2*site(i) + site(i+1)
            LET sitenew(i) = update(index)
            IF sitenew(i) = 1 then BOX AREA 1+i*4,i*4+4,1+t*4,t*4+4
        NEXT i
        MAT site = sitenew
        LET site(0) = site(L)     ! periodic boundary conditions
        LET site(L+1) = site(1)
    NEXT t
END SUB
```

The properties of all 256 one-dimensional cellular automata have been cataloged (see Wolfram). We explore some of the properties of one-dimensional cellular automata in Problems 15.1 and 15.2.

Problem 15.1. One-dimensional cellular automata

a. Use `Program ca1` and rule 90 shown in Fig. 15.1. This rule also is known as the "modulo-two" rule, because the value of a site at step $t+1$ is the sum modulo 2 of its two neighbors at step t. Choose the initial configuration to be a single nonzero site (seed) at the midpoint of the lattice. It is sufficient to consider the time evolution for approximately twenty steps. Is the resulting pattern of nonzero sites self-similar? If so, characterize the pattern by a fractal dimension.

b. Consider the properties of a rule for which the value of a site at step $t+1$ is the sum modulo 2 of the values of its neighbors *plus* its own value at step t. This rule is termed rule 10010110 or rule $150 = 2^1 + 2^2 + 2^4 + 2^7$. Start with a single seed site.

c. Choose a random initial configuration for which the independent probability for each site to have the value 1 is 50%; otherwise, the value of a site is 0. Consider the time evolution of rule 90, rule 150, rule $18 = 2^1 + 2^4$ (00010010), rule $73 = 2^0 + 2^3 + 2^6$ (01001001), and rule 136 (10001000). How sensitive are the patterns that are formed to changes in the initial conditions? Does the nature of the patterns depend on the use or nonuse of periodic boundary conditions?

Because the dynamical behavior of many of the 256 one-dimensional Boolean cellular automata is uninteresting, we also consider one-dimensional Boolean cellular automata with larger neighborhoods. The larger neighborhood implies that there are many more possible update rules, and it is convenient to place some reasonable restrictions on the rules. First, we assume that the rules are symmetric, for example, the neighborhood 100 produces the same value for the central site as 001. We also assume that the zero neighborhood 000 yields 0 for the central site, and that the value of the central site depends only on the sum of the values of the sites in the neighborhood, for example, 011 produces the same value for the central site as 101 (Wolfram 1984).

A simple way of coding the rules that is consistent with these requirements is as follows. Call the size of the neighborhood z if the neighborhood includes $2z+1$ sites. Each rule is labeled by $\sum_m a_m 2^m$, where $a_m = 1$ if the central cell is 1 when the sum of all values in the neighborhood equals m; else $a_m = 0$. As an example, take $z = 2$ and suppose that the central site equals one when two or four sites are unity. This rule is labeled by $2^2 + 2^4 = 20$.

Problem 15.2. More one-dimensional cellular automata

a. Modify `Program ca1` so that it incorporates the possible rules discussed in the text for a neighborhood of $2z+1$ sites. How many possible rules are there for $z = 1$? Choose $z = 1$ and a random initial configuration, and determine if the long time behavior for each rule belongs to one of the following classes:

   (a) A homogeneous state where every site is either 0 or 1. An example is rule 8.

   (b) A pattern consisting of separate stable or periodic regions. An example is rule 4.

   (c) A chaotic, aperiodic pattern. An example is rule 10.

   (d) A set of complex, localized structures that may not live forever. There are no examples for $z = 1$.

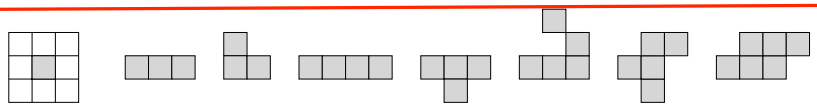

Figure 15.2: (a) The local neighborhood of a site is given by the sum of its eight neighbors. (b) Examples of initial configurations for the Game of Life, some of which lead to interesting patterns. Live cells are shaded.

b. Modify your program so that $z = 2$. Wolfram (1984) claims that rules 20 and 52 are the only examples of complex behavior (class 4). Describe how the behavior of these two rules differs from the behavior of the other rules. Determine the fraction of the rules belonging to the four classes.

c. Repeat part (b) for $z = 3$.

d. Assume that sites can have three values, 0, 1, and 2. Classify the behavior of the possible rules for the case $z = 1$.

The results of Problem 15.2 suggest that an important feature of cellular automata is their capability for "self-organization." In particular, the class of complex localized structures is distinct from regular as well as aperiodic structures. This intermediate structure is the focus of *complexity theory* whose goal is to explain complex phenomena in nature.

One-dimensional models are too limited to study the complexity of nature, and we now consider several two-dimensional models. The philosophy is the same except that the neighborhood contains more sites. `Program ca2` sets up the rule matrix and updates sites using the eight neighbor sites shown in Fig. 15.2a. There are now $2^9 = 512$ possible configurations for the eight neighbors and the center site, and 2^{512} possible rules. Clearly, we cannot go through all these rules in any systematic fashion as we did for one-dimensional cellular automata. For this reason, we will set up our rule matrix based on other considerations.

The rule matrix incorporated in `Program ca2` implements the best known two-dimensional cellular automata model: the *Game of Life*. This model, invented in 1970 by the mathematician John Conway, produces many fascinating patterns. The rules of the game are simple. For each cell determine the sum of the values of its four nearest and four next-nearest neighbors (see Fig. 15.2a). A "live" cell (value 1) remains alive only if this sum equals 2 or 3. If the sum is greater than 3, the cell will "die" (become 0) at the next time step due to overcrowding. If the sum is less than 2, the cell will die due to isolation. A dead cell will come to life only if the sum equals 3.

```
PROGRAM ca2
LIBRARY "csgraphics"
DIM update(0 to 511),cell(50,50)
CALL setrule(update(),L)
LET flag$ = ""
DO
```