Cellular Automata Modeling of Chemical Systems

A textbook and laboratory manual

Lemont B. Kier, PhD

Professor of Medicinal Chemistry
Senior Fellow, CSBC
Virginia Commonwealth University

Paul G. Seybold, PhD

Professor of Chemistry
External Fellow, CSBC
Wright State University

Cho-Kun Cheng, PhD

Associate Professor of Computer Science
Fellow, CSBC
Virginia Commonwealth University

A publication of the Center for the Study of Biological Complexity
Virginia Commonwealth University
Richmond Virginia

Springer

Chapter 2

CELLULAR AUTOMATA

To discover and analyze the mathematical basis for the generation of complexity, one must identify simple mathematical systems that capture the essence of the process. Cellular automata are a candidate class of such systems. . . . Cellular automata promise to provide mathematical models for a wide variety of complex phenomena, from turbulence in fluids to patterns in biological growth.

–Stephen Wolfram [1]

In the first chapter several traditional types of physical models were discussed. These models rely on the physical concepts of energies and forces to guide the actions of molecules or other species, and are customarily expressed mathematically in terms of coupled sets of ordinary or partial differential equations. Most traditional models are deterministic in nature—that is, the results of simulations based on these models are completely determined by the force fields employed and the initial conditions of the simulations. In this chapter a very different approach is introduced, one in which the behaviors of the species under investigation are governed not by forces and energies but by rules. The rules, as we shall see, can be either deterministic or probabilistic, the latter leading to important new insights and possibilities. This new approach relies on the use of cellular automata.

Cellular automata (CA) were first proposed by the mathematical physicist John von Neumann and the mathematician Stanislaw Ulam more than a half century ago [2–4] and similar ideas were suggested at about the same time, in the 1940s, by the German engineer Konrad Zuse [5–7]. Von Neumann's interest was in the construction of "self-reproducing automata." [8] His original idea was to construct a series of mechanical devices or "automata" that would gather and assemble the parts necessary to reproduce themselves. A suggestion by Ulam led him to consider more abstract systems consisting of grids with moving ingredients, operating under sets of rules. The first such system proposed by von

Neumann consisted of a two-dimensional grid of square cells, each having a set of possible states, along with a set of rules. The system he developed eventually employed as many as 29 different possible states for the cells, and was, in the least, clumsy to work with. With the development of modern digital computers, however, it became increasingly clear to a small number of scientists that these very abstract ideas could in fact be usefully applied to the examination of real physical and biological systems, with interesting and informative results [9,10].

A number of research groups have subsequently developed different realizations of the CA paradigm for the study and simulation of a broad range of physical, biological, chemical, and even sociological, phenomena. These models have contributed important new insights regarding the deeper, often hidden, factors underlying a host of complex phenomena. These diverse CA studies have been especially important in treating the often-surprising behaviors of systems where large numbers of complicated interactions between the system ingredients serve to hide the general patterns involved and, in addition, render the conventional, differential-equation-based methods difficult to implement or ineffective, i.e., complex systems. In this book we shall describe the details of particular CA models developed by our own research groups for the study and simulation of complex physical, chemical, and biochemical systems.

The present chapter will focus on the practical, "nuts and bolts" aspects of this particular CA approach to modeling. In later chapters we will describe a variety of applications of these CA models to chemical systems, emphasizing applications involving solution phenomena, phase transitions, and chemical kinetics. In order to prepare readers for the use of CA models in teaching and research, we have attempted to present a user-friendly description. This description is accompanied by examples and "hands-on" calculations, available on the compact disk that comes with this book. The reader is encouraged to use this means to assimilate the basic aspects of the CA approach described in this chapter. More details on the operation of the CA programs, when needed, can be found in Appendix I of this book.

2.1. What are cellular automata?

But just what are cellular automata? Mathematician Stephen Wolfram has defined cellular automata as follows [1]:

Cellular automata are simple mathematical idealizations of natural systems. They consist of a lattice of discrete identical sites, each site taking on a finite set of, say, integer values. The values of the sites evolve in discrete time steps according to deterministic rules that specify the value of each site in terms of the values of neighboring sites. Cellular automata may thus be considered as

discrete idealizations of the partial differential equations often used to describe natural systems.

Wolfram has elaborated on this description elsewhere [11, 12]. As we shall see, the restriction to deterministic rules is unnecessary, and we shall in fact make extensive use of probabilistic rules in our studies of real physical and chemical systems.

Cellular automata, then, are models, in the same sense that the Monte Carlo and molecular dynamics approaches are models, which can be employed for the purpose of simulating real systems. We shall use the term cellular automaton (singular) to refer to a model consisting of the following components:*

- A grid composed of cells.
- A set of ingredients.
- A set of local rules governing the behaviors of the ingredients.
- Specified initial conditions.

Once the above components of the model are defined, a simulation can be carried out. In the simulation the system evolves *via* a series of discrete time-steps, or iterations, in which the model rules are applied to all the ingredients of the system, and the configuration of the system is accordingly updated.

A striking feature of the cellular automata (CA) models is that they treat not only the ingredients, or agents, of the model as discrete entities, as do the traditional models of physics and chemistry, but now time (iterations) and space (the cells) are also regarded as discrete, in stark contrast to the continuous forms for these parameters assumed in the traditional, equation-based models. In practice, as we shall see, this distinction makes little or no difference, for the traditional continuous results appear, quite naturally, as limiting cases of the discrete CA analyses. Nonetheless, this quantization of time and space does raise some interesting theoretical and philosophical questions, which we shall, however, ignore at this time.†

A new, important feature is sometimes observed in studies of the evolutions of these computational systems: the development of unanticipated patterns of

---

* Historically, there has been some looseness of terminology in this field. A few authors have used the term "cellular automaton" to refer to a cell. We shall not use the term in this sense.

† Some recent theories of modern physics, such as string theory and quantum loop gravity theory, raise the interesting possibility that at some ultimate level—perhaps at very short times approaching the so-called *Planck time*, about $|||10^{-43}$ s, and at distances approaching the *Planck distance*, about 10^{-35} m—the discrete natures of time and space might reveal themselves. Time, for example, might proceed in jumps at very short times, in the same way that quantum theory shows that energy comes in jumps called quanta, when events at the submicroscopic level are examined. Physicists refer to such hypothetical time units as "chronons" in analogy to the "photons" of light energy. At the present time, of course, such extremely short times and distances lie well beyond experimental detection.

ordered dynamical behavior. These patterns have come to be called emergent properties. As Stuart Kauffman has expressed it [13]

Studies of large, randomly assembled cellular automata... have now demonstrated that such systems can spontaneously crystallize enormously ordered dynamical behavior. This crystallization hints that hitherto unexpected principles of order may be found [and] that the order may have significant explanatory import in [biology] and... physics.

Experience has shown that this enticing assessment is, in fact, too cautious: cellular automata carry great potential for revealing "hitherto unexpected principles" not only in biology and physics but also in chemistry and a host of other fields as well (see Appendix II).

As noted earlier, a variety of different models can be developed within the general CA framework pictured above. In this book we describe a particular realization of this concept that the present authors have found especially well suited for the examination of physicochemical and biochemical systems. We shall now examine the components of this CA model in more detail.

2.2. The grid and the cells

2.2.1. The grid

The grid in a CA model may contain a single cell, or more commonly a larger collection, with possibly as many as 100,000 or more cells. In principle, the grid itself might be one-, two-, or three-dimensional in form, although most studies have used two-dimensional grids. These two-dimensional grids will be the principal focus in this book. A moving ingredient may encounter an edge or boundary during its movements. Three general types of two-dimensional grids will be considered relating to the boundaries: (1) a box, (2) a cylinder, and (3) a torus. In the box grid, moving ingredients encounter boundaries on all four sides; in the cylinder they encounter only top and bottom boundaries; and in the torus no boundaries are present to restrict the ingredient movements. An illustration of a $7 \times 7 = 49$ cell grid of square cells occupied by two types of ingredients, A and B, is shown in Figure 2.1.

The nature of the grid type employed will normally depend on the boundary characteristics of the system of interest. For some systems, e.g., when the ingredients themselves are either stationary or confined, a box grid is perfectly suitable. In other cases, one may need only a constraining top and bottom (or right and left sides), and a cylindrical grid will be most appropriate. An example here would be the condensation of a gas to a liquid under the influence of gravity (see Chapter 9). For this, a bottom is required to restrain the liquid below and a top holds in the gas, but the ingredients remain free to move unobstructed to the

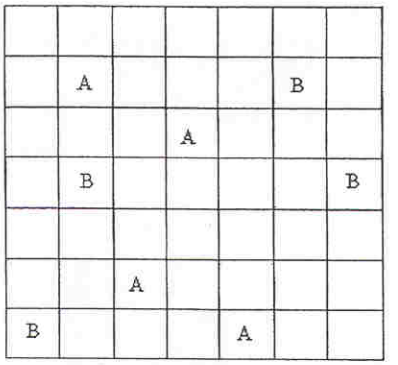

Figure 2.1. A two-dimensional cellular automata grid. Shown are two sets of occupied cells of different states, A and B. The unoccupied cells are blank

right or left, such that those moving off the edge to the right will appear at the left, and those moving off the grid to the left will appear on the right. In all these cases the ingredients can only move to unoccupied cells. The torus effectively simulates a small segment of a larger, unrestricted system by allowing cells also to move off the top edge and appear at the bottom, or move off the bottom edge and appear at the top. These features are shown in Figures 2.2 and 2.3.

In most cases, and in all cases found in this book, the cells of the boundaries are themselves inert, and have no interactions with the grid ingredients other than to constrain their movements in certain directions. However, more generally the boundary cells can be constructed to have active properties just like any other ingredient, following rules (see below) that permit joining, breaking,

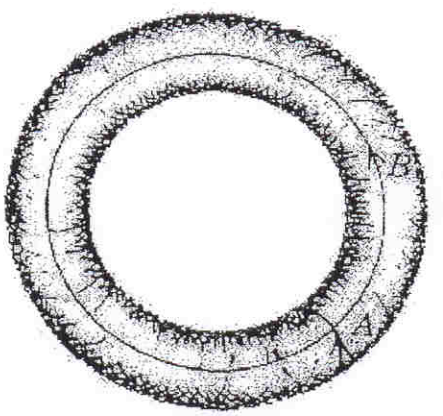

Figure 2.2. A torus grid eliminates boundaries

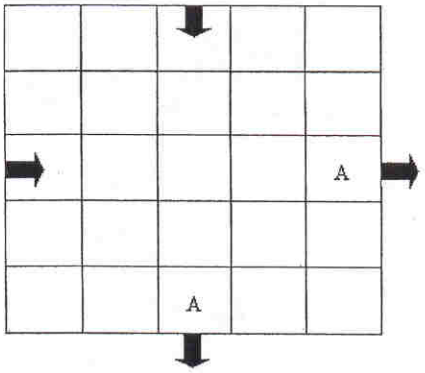

Figure 2.3. Movement of cell ingredients at the boundaries of a grid on the surface of a torus. One ingredient A might move off the grid to the right and reappear at the left edge of the grid. Another ingredient A might move off the bottom of the grid and reappear at the top of the grid

and reacting with the grid ingredients. Thus in principle, the boundaries can be either inert or active in some way.

2.2.2. The cells

The cells themselves can take a variety of shapes; they can be triangles, squares, hexagons, or other shapes on the two-dimensional grid, with square cells being most common. Each cell in the grid can normally exist in a number of distinct "states" which define the occupancy of the cell. The cell can be empty or contain a specific ingredient, where the ingredient, if present, might represent a particle, a type of molecule or isomer, a particular molecular electronic state, an organism, an automobile, or some other entity pertinent to the study in question.

The choice of the cell shape is based on the objective of the study. In the case of studies of water-related phenomena, for example, square cells are especially advantageous, since water molecules, H_2Os, are quadravalent with respect to their participation in intermolecular hydrogen bonding. An individual water molecule can employ two hydrogen atoms and two lone pairs of electrons to form hydrogen bonds with its neighbors. This leads to the tetrahedral configuration found in ice, a structure that is retained to some extent in the liquid state. The four faces of a square cell thus correspond nicely to the four hydrogen-bonding opportunities of a water molecule.

The interactions of an ingredient with other ingredients take place at the cell edges. Originally, cellular automata models routinely assumed that all of the edges of a given ingredient should obey the same rules. More recently, the idea of a variegated cell, in which each edge can have its own independent rules

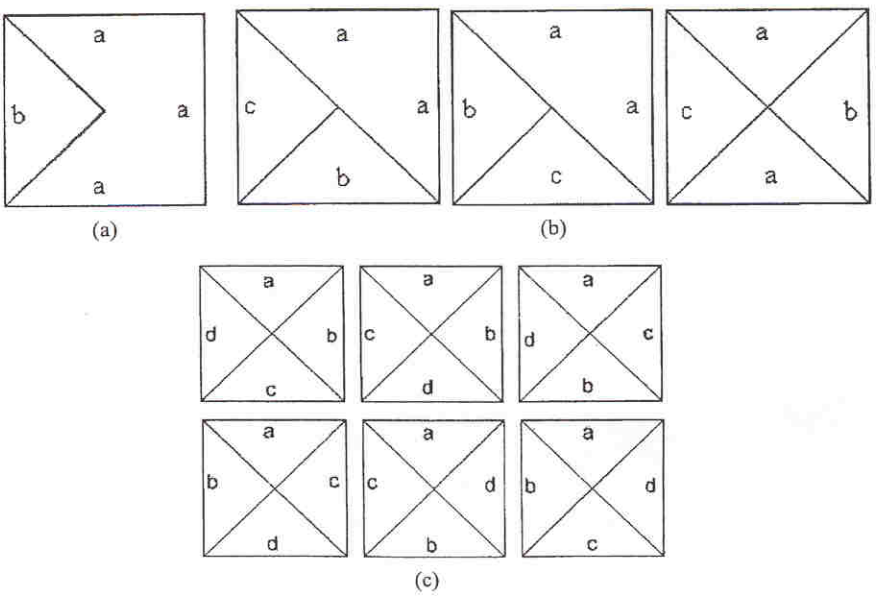

Figure 2.4. Examples of variegated cells: (a) Two different types of edges with different rules, (b) Three different types of edges with different rules, and (c) Four different types of edges with different rules

for interaction with other ingredients, has been introduced and shown to have considerable value in modeling [14]. Examples of some types of variegated cells are shown in Figure 2.4.

2.2.3. Cell neighborhoods

As we shall soon see, the movements and other actions of an ingredient on the grid are governed by rules, and these rules depend only on the nature of the cells in close proximity to the ingredient. This proximate environment of a cell is called its neighborhood. The most common neighborhood used in two-dimensional cellular automata studies is called the von Neumann neighborhood, after the original pioneer of the CA method. This neighborhood for a cell, i, refers to the four, j, cells adjoining its four faces (Figure 2.5a). Another common neighborhood is the Moore neighborhood, pictured in Figure 2.5b, referring to the eight j cells completely surrounding cell i, including those cells on the diagonals. Another useful neighborhood is the extended von Neumann neighborhood, shown in Figure 2.5c, where the four k cells lying just beyond the four j cells of the von Neumann neighborhood are included.

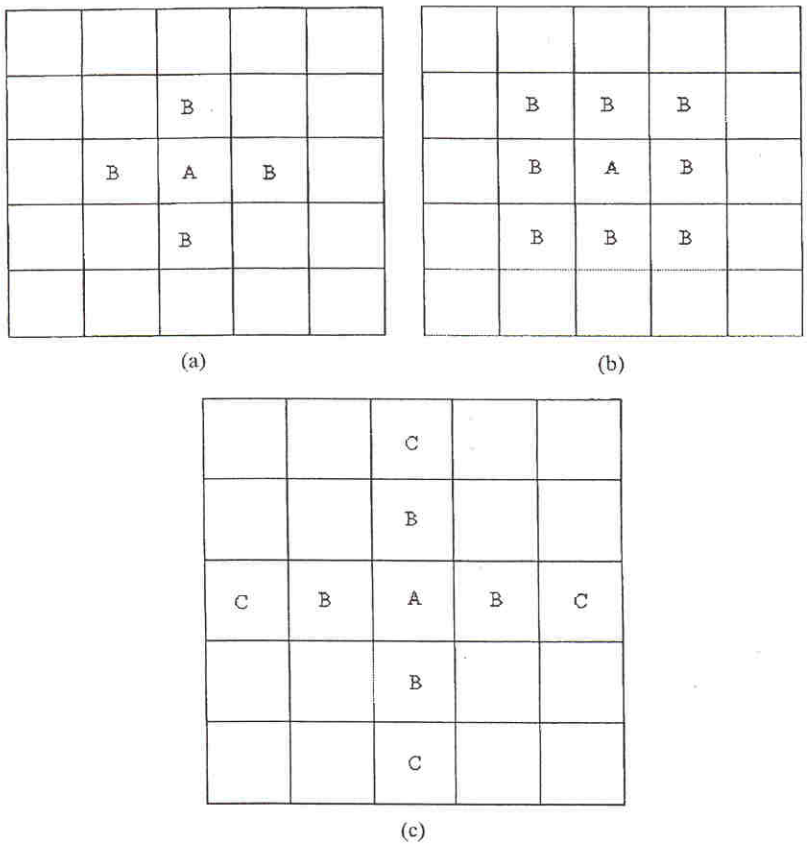

Figure 2.5. Cell neighborhoods: (a) the von Neumann neighborhood, (b) the Moore neighborhood, and (c) the extended von Neumann neighborhood of cell *i*

2.3. The rules

2.3.1. Types of rules

Several different types of rules govern the behaviors of the ingredients on the grid, and thereby the subsequent evolutions of the CA systems. Movement rules govern the movements of ingredients about the grid. These rules take several forms. The breaking probability, $P_B(AB)$, determines the degree to which two adjacent ingredients A and B tend to stay bonded, or "stick," to each other. The joining parameter, $J(AB)$, establishes the propensity of an A ingredient to move toward or away from a B ingredient, when these two ingredients are separated by an empty cell. The free-moving probability $P_m(A)$

of an ingredient A defines the ingredient's tendency to move more rapidly or more slowly on the grid. A gravitational parameter G, if present, denotes a greater-than-random tendency for an ingredient to move downward on the grid, thereby distinguishing motion in this direction from motion in the other directions on the grid. Transition rules govern the likelihood that during an iteration of the system, an ingredient will transform to some other species. The simple first-order transition probability P_T (AB) defines the probability that an ingredient of species A will change to species B during an iteration. The reaction probability P_R(AB) defines the probability that ingredients A and B will transform to species C and D, respectively, when they "encounter" each other (come into contact) during their movements about the grid. Occasionally other types of rules may be added. The key features of all these rules are that they are local, involving only an ingredient itself and possibly those ingredients in its immediate neighborhood, and that they are uniformly applied throughout the CA simulation.

2.3.2. Transition rules

Transitions occur constantly in nature; molecules change from one tautomeric form to another, radioactive nuclei decay to form other nuclei, acids dissociate, proteins alter their shapes, molecules undergo transitions between electronic states, chemicals react to form new species, and so forth. Transition rules allow the simulation of these changes.

As indicated, transition rules govern the probability that during each iteration of the simulation, an ingredient will transform to a different type of ingredient. If P_T(AB) = 1.0 the transition A → B is certain to occur; if P_T(AB) = 0.0, it will never occur. But if, for example, P_T(AB) = 0.5, then during each iteration, there will be a 50% chance that the transition A → B will occur. The first two cases can be considered deterministic, since they do not allow for different outcomes. The third case is stochastic, however, since it allows different outcomes, the ingredient might remain unchanged or it might transform to a different state.[‡] The transition probabilities may, in some cases, depend on the conditions prevailing in neighboring cells. For example, the transformation probability P_T(AB) might depend on the occupancies of neighboring cells. In reaction simulations two ingredients A and B that come in contact on the grid will have a probability P_R(AB) of "reacting," or transforming, to

[‡] The probabilities are enforced using a random-number generator in the CA program. Suppose that the random-number generator generates numbers between 1 and 1000. For a 20% probability of movement, one might assign a choice of numbers between 1 and 200 inclusively as a "move" decision, while a choice in the remaining number set, 201–1000, is a "no-move" decision. Each ingredient is then assigned a random number, and correspondingly behaves according to that value.

other species, say, C and D, during such an "encounter." In this case the reaction probability $P_R(AB)$ defines the probability that the reaction A + B → C + D will occur when A and B encounter one another in the course of their motions. If $P_R(AB) = 1$, the reaction will take place on every encounter, but if $P_R(AB) = 0.1$, for example, only 10% of such encounters will lead to reaction.

2.3.3. Movement rules

Much of the dynamic character of cellular automata models is developed through the movements of the ingredients about the grid. During each time-step interval, or iteration, in the CA simulation, an ingredient on the grid has the possibility of moving vertically or horizontally to an adjacent, unoccupied cell. In the absence of further restrictions, a free ingredient would therefore, over time, perform a random walk about the grid. Normally, however, there are other ingredients on the grid, and the presence of these ingredients will influence the motion of the first ingredient. During each iteration the movement of every ingredient on the grid is computed based on rules (described below) that involve the status of its neighboring cells, i.e., whether these cells are empty or occupied, and, if occupied, by what types of ingredients. Deterministic cellular automata use a fixed set of rules, the values of which are immutable and uniformly applied to the ingredients. In probabilistic, or stochastic, cellular automata, the movements of the ingredient are based on probabilistic rules, embodied as probabilities of moving or not moving during an iteration. We shall now consider the several types of movement rules individually.

2.3.4. The free-moving probability P_m

The free-moving probability $P_m(A)$ defines the probability that an ingredient A in a cell, i, will move to one of the four adjacent cells, j, in its von Neumann neighborhood, if that space is unoccupied. An example would be the ingredient A in cell, i, in Figure 2.6a, which might move to any of the unoccupied neighboring cells, j. Two ingredients might move simultaneously as in Figure 2.6b. This will be discussed later. As a matter of course this probability is usually set at $P_m = 1.0$, which means that a movement in one of the allowed directions always happens (a rule). However, in some cases P_m can be set to lower values if certain species in the CA simulation are to be regarded as moving more slowly than others.

2.3.5. The joining parameter J

The first of the two trajectory or interaction rules is the joining trajectory parameter, $J(AB)$, which defines the propensity of movement of an ingredient A toward or away from a second ingredient B, when the two are separated by

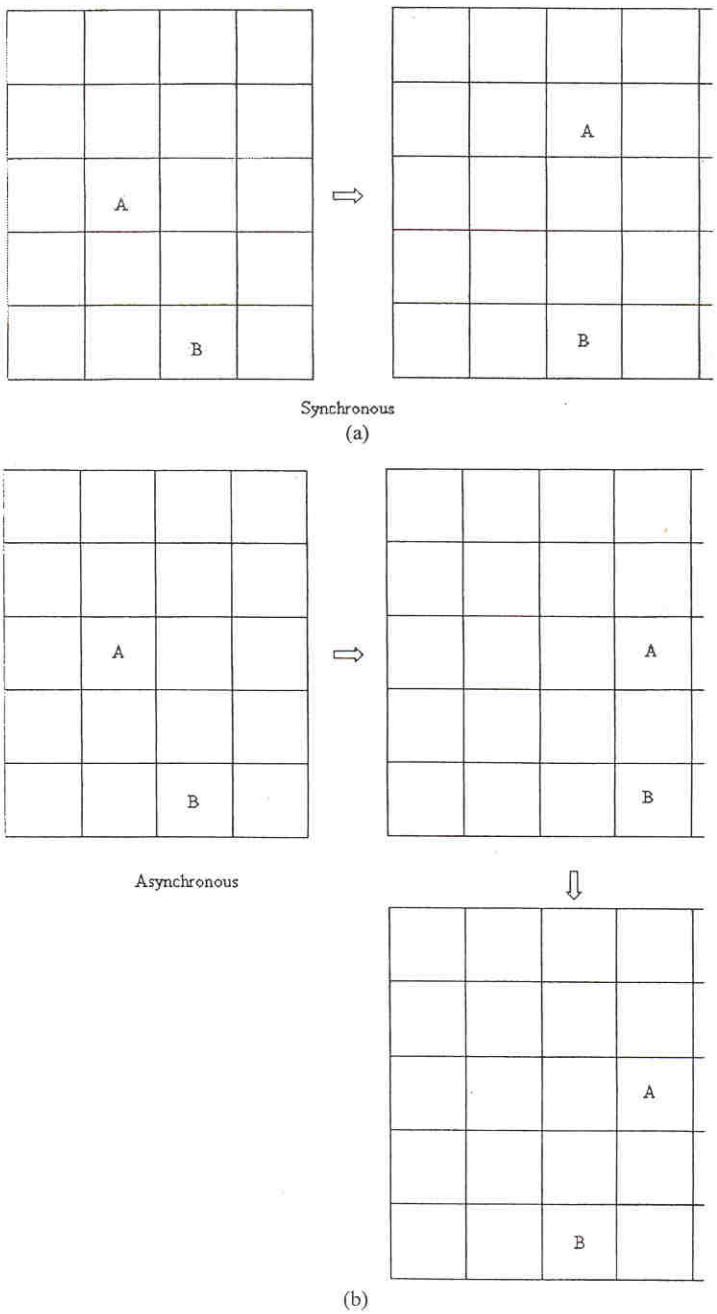

Figure 2.6. Possible movement of cell i occupant A to an unoccupied cell j

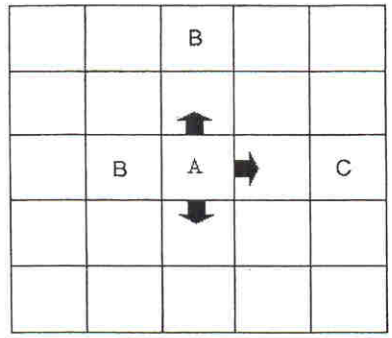

Figure 2.7. Effect of the joining parameter J on the movement of ingredient A in different directions

a vacant cell (Figure 2.7). It thus involves the extended von Neumann neighborhood of ingredient A, and has the effect of adding a short-range attraction or repulsion component to the interaction between ingredients A and B. J is a nonnegative real number. When $J = 1$, species A has the same probability of movement toward or away from B, as when the B cell is not present. When J is greater than 1, ingredient A has a greater probability of movement toward a B ingredient than when ingredient B is absent, simulating, in effect, a degree of short-range attraction. When J lies between 0 and 1, ingredient A has a lower probability of such movement, and this can be considered as a degree of mutual repulsion. When $J = 0$, ingredient A cannot move toward B at all.

2.3.6. The breaking probability P_B

The second trajectory or interaction rule is the breaking probability, P_B. This parameter, in effect, assigns a "stickiness" to the interaction between two ingredients that are in contact, i.e., adjacent to each other on the grid. The breaking rule assigns the probability $P_B(AB)$ that an ingredient A, adjacent to an ingredient B, will break apart from B, as shown in Figure 2.8a. The value for P_B necessarily lies within the range 0–1. Low values of P_B imply a strong cohesion between A and B, whereas high values indicate little cohesion. Thus if $P_B = 0$, the ingredients will not separate from each other, and if $P_B = 1$, they have no tendency to adhere to one another. If P_B lies between these values, there is an intermediate tendency to break apart. When molecule A is bordered by two ingredients, B and C, the simultaneous probability of A breaking away is given by the product $P_B(AB) \times P_B(AC)$, as shown in Figure 2.8b. If ingredient A has three adjacent ingredients (B, C, and D), the simultaneous breaking probability of ingredient A, the probability that it will move to the remaining adjacent empty cell, is $P_B(AB) \times P_B(AC) \times P_B(A, D)$, shown in Figure 2.8c. Of

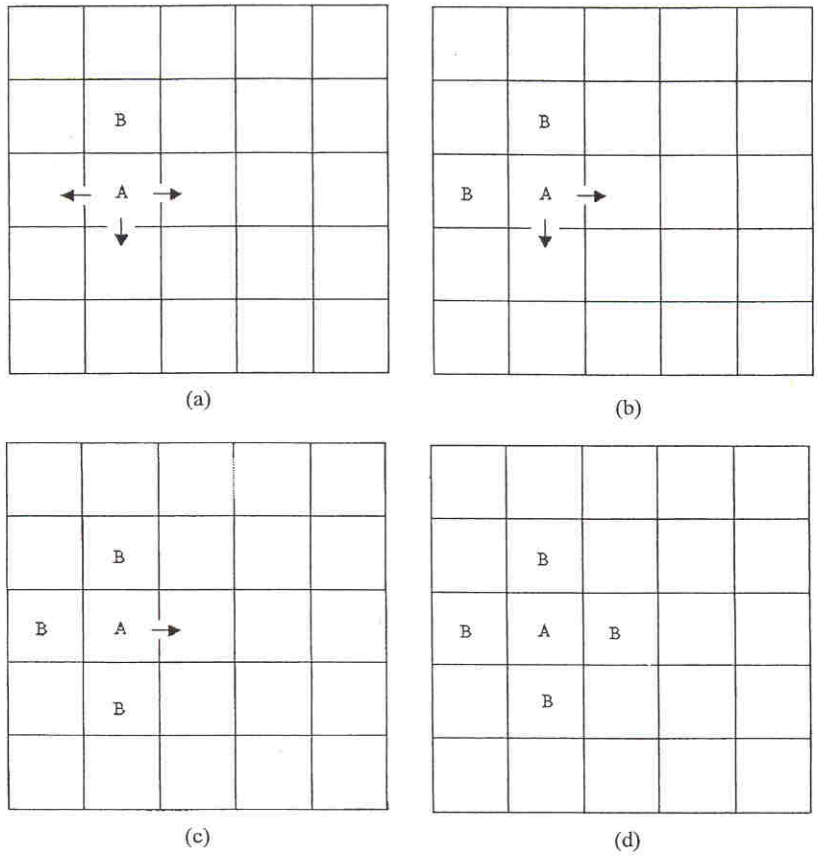

Figure 2.8. The possible directions of ingredient A breaking away from ingredient B: (a) A is bound to one B cell, (b) A is bound to two B cells, (c) A is bound to three B cells, and (d) A is bound to four B cells

course, if ingredient A is surrounded by four ingredients (Figure 2.8d), it cannot move.

2.3.7. Relative gravity rules

The simulation of a "gravity" effect can be introduced into a cellular automaton model in two different ways. Separation phenomena like the demixing of immiscible liquids can be simulated using a relative gravity rule [15]. For this, a boundary condition is first imposed at the upper and lower edges of the grid to apply vertical limits on the motions of the ingredients (a cylindrical grid). The differential effect of gravity on different ingredients A and B is simulated by

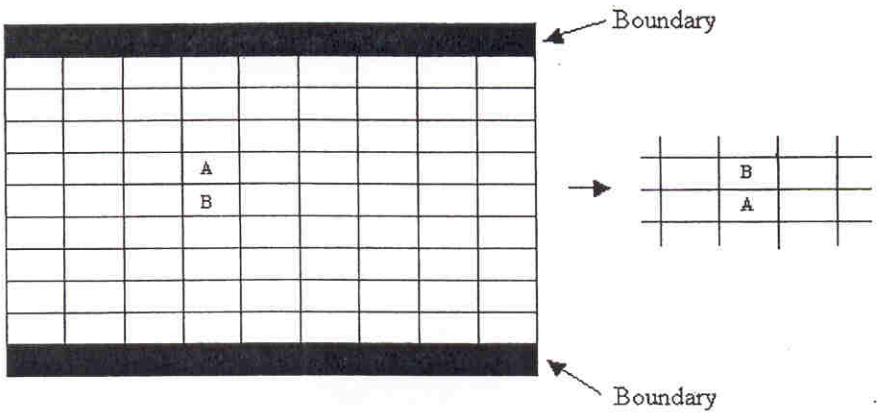

Figure 2.9. Illustration of relative gravity rules influencing cells A and B

introducing reciprocal rules governing their tendencies to exchange positions when they come together. When one ingredient moves to a position on top of the other, the rules are applied. The first rule, $G_R(AB)$, applies when A is above B and is the probability that ingredient A will exchange places with ingredient B, so that A will appear below, and B above. The complementary rule is $G_R(BA)$, which expresses the probability that molecule B, originally above A, will exchange positions with A and end up below. These rules are illustrated in Figure 2.9.

When $G_R(AB)$ is greater than $G_R(BA)$, there will be an overall tendency for the A ingredients to congregate below the B ingredients, and when $G_R(AB)$ is less than $G_R(BA)$, the A ingredients will tend toward the upper part of the grid. In the first case the As can be thought of as forming a more dense liquid than the Bs, and in the latter case, a less dense liquid. The G_R rules are probabilities that the events will occur.

2.3.8. The absolute gravity rule

In other simulations an absolute gravity rule, denoted by $G_A(A)$, is more appropriate [16]. This rule favors motion in a preferred direction. For example, one might wish to simulate the motions of different gas molecules, some heavier than others, in a gravitational field. The value $G_A(A) = 0$ is the neutral value, so that the movement probabilities are equal in all four directions. Values greater than $G_A(A) = 0$ increase the likelihood of downward movements. Thus a value of $G_A(A) = 0.2$ would impose a slight tendency on the ingredients of species A to move downward on the grid, and $G_A(A) = 0.5$ a would impose a much stronger tendency.

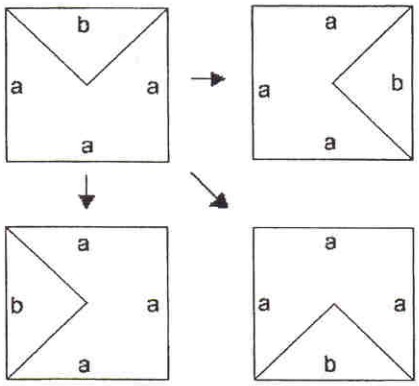

Figure 2.10. Illustration of cell rotation rules

2.3.9. Cell rotation rules

In those cases where variegated ingredients are used (for examples, see Figure 2.4), it is necessary to ensure that there exists a balanced representation of the possible rotational states of these ingredients on the grid. To accomplish this, the variegated cells are rotated randomly, by 90°, −90°, +180°, or −180°, during every iteration of the run. Only free cells rotate; when a variegated ingredient has a neighboring ingredient in its von Neumann neighborhood, it does not rotate. This rotation process is illustrated for three possible state changes in an iteration in Figure 2.10.

2.3.10. Synchronous or asynchronous application of the rules

A complete time-step (iteration) in a CA model involves the application of all the applicable model rules to all the ingredients on the grid. During an iteration the movement rules can be applied either simultaneously (synchronously), Figure 2.6a, or sequentially (asynchronously), Figure 2.6b. Alert readers will recognize at this point that synchronous application of the governing movement rules for a CA simulation, as outlined above, can lead in some instances to conflicts, e.g., the assigning of two ingredients to move to the same empty cell, just as a similar conflict arises when two cars simultaneously attempt to move into the same vacant parking space. As a result, synchronous rule application is not practical for cellular automata modeling within the framework described here. In asynchronous application of the rules, in contrast, ingredients are selected in random order for application of the movement rules, and such potential conflicts are avoided. Only asynchronous application of the rules to the different ingredients will be used in the applications covered in this book.

The further question of the order to use in applying the different movement rules for a single ingredient does not arise in the present case, since the P_m, $P_B(AB)$, and $J(AB)$ are joined in a single probabilistic equation when determining whether or not a single ingredient will move. The key factor is that any order assigned should be followed consistently and should be reasonable.

2.4. Running a simulation

Having defined both the grid type and size and the governing rules for a simulation, the latter by assigning specific values to the parameters described above, one next needs to define the remaining conditions of the simulation. These include (1) the natures and numbers of the starting ingredients, (2) the configuration of the initial state of the system, (3) how many runs of the simulation are to be carried out, and (4) the length of the runs, i.e., how many iterations they should include.

2.4.1. The initial ingredients

Before beginning, it is necessary to define the starting condition of the system. Here one first declares what types of ingredients should be present at the start of the run and how many of each type should be present. For example, one might wish to work with four species, but start with just two of them on the grid, the others being generated as the system evolves. One might then designate the initial numbers as, say, 250 A ingredients and 500 B ingredients, with zeros for the remaining species C and D. The ingredients are customarily distinguished on the computer screen by different colors. In the present example, the As might be blue, the Bs green, and the Cs and Ds, which are not initially present, red and brown, respectively.

In order for ingredients to move on the grid, there must be empty cells available to accommodate them. For studies of aqueous systems, it has been found that leaving about 31% of the cells on the grid empty provides a reasonable description of the actual condition for motion of the ingredients representing water molecules [17, 18]. Therefore in a water study using a 50×50 grid with 2500 cells, there should be 1725 cells occupied by "water" ingredients and 775 empty cells.

2.4.2. The initial conditions

The default condition for placement of the starting ingredients is to position them randomly on the grid. For some studies, however, a different distribution might be needed. For instance, when one wants to examine the dissolving of a block of ingredients into a surrounding liquid, one might wish to place an array of the ingredients in the center of the grid and then surround them with

the ingredients of the liquid. In another case one might wish to examine the diffusion of a gas into an open area or the passage of ingredients through a membrane, so that appropriate structures for these studies must be constructed as part of the initial configuration on the grid. In the simulations to be described in this and later chapters, any required special constructions or distributions will be set up automatically by the program, so that it will not be necessary for the reader to create them.

2.4.3. The CA runs

It needs to be emphasized at this juncture that when the CA rules are stochastic, i.e., probabilistic, each simulation run is, in effect, an independent "experiment." This means that in principle, the results from separate runs can possibly be quite different, as chance would have it. This, indeed, does occur in some studies, as we shall see. In general, the behavior of a single ingredient is completely unpredictable. However, for most examples, we shall examine which *collective outcome*, either that from a run with a great number of ingredients or that from a great number of runs with few ingredients, tends to display a similar pattern. In the same way that laboratory experiments when repeated tend to yield similar results, so too CA simulations yield similar patterns. These patterns are the emergent properties associated with the simulations. In the same way, laboratory workers customarily repeat their experiments several times in order to establish the statistical validity of their results, yielding, e.g., an average result and a standard deviation, which is a measure of the experimental error associated with the measurements. This same procedure for the CA runs, repetition of the runs, yields an analogous indication of the uncertainty involved in the simulations. We shall see that as a rule relative error tends to decrease as the sample size increases, or as more runs are employed, just as it does in laboratory experiments.

Accordingly, two further simulation details need to be established, the number of independent runs to be performed and the length (in iterations) of these runs. These numbers will depend very much on the nature of the simulation to be performed. In some cases a relatively short run of, say, 70 iterations, might be enough to make the point needed. But in other, more typical, situations runs of several thousand iterations might be more appropriate, and one might wish to perform several runs in order to establish the uncertainties in any numerical results that appear. In some cases it will be desirable to allow the runs to proceed long enough for some sort of steady-state or equilibrium condition to be achieved. Such a stability point can normally be recognized when the output values exhibit a relatively constant average value over a number of iterations.

It is useful to note here that the CA simulations to be described in this chapter and later chapters tend to be ergodic in the sense that the time average value for a particular property of a single system (i.e., the average taken over a

long time period after the system has reached its steady-state condition) and the ensemble average value for this property (i.e., the average obtained from a large number of runs for the system at a specific time after reaching steady state) are closely identical [19]. Normally it is much simpler to allow a single simulation to run for some time and to obtain property averages from the postequilibrium portion of the run, than to perform a large number of separate runs.

2.5. The output

The output of a CA simulation carried out on a computer comes in two different forms: a visual output that is displayed on the computer screen, and numerical data summaries compiled in output files that are generated during each run. The visual output allows the observer to follow the system as it evolves, and can be very helpful in comprehending the overall process of the system's evolution. The data summaries in the output files are more suitable for quantitative analysis of the details of this evolution.

The numerical data files list the values of specific properties, or attributes, of the system as they change with time. The overall configuration of the system, i.e., the specific arrangement of its ingredients, evolves in time as the cellular automaton rules are applied during each iteration. Accordingly, the system's attributes evolve with time. For kinetic studies the relevant attributes are normally the counts of the different species present and the numbers of the various types of transitions that take place during each iteration. In studies of liquids the numbers of ingredients engaged in different types of "bonding" arrangements are typically listed as they vary with the iterations. The positions on the grid of the different ingredients might also be of interest in some studies. The variations in the system attributes with time can often be related to important macroscopic phenomena taking place in the real systems being simulated.

2.5.1. Liquid simulation outputs

For typical simulations used in the study of aqueous and other liquid systems, several attributes are customarily recorded and used in comparative studies. These attributes used singly or in combination are useful for analyses of different phenomena (Examples of the use and significance of these attributes will be described in later examples). The commonly collected attributes for the liquid systems relate mainly to the states of bonding, i.e., the numbers of adjacent ingredients in the von Neumann neighborhood, for the ingredients. Their designations are as follows:

f_0—fraction of ingredients not bound to other ingredients,

f_1—fraction of ingredients bound to only one other ingredient,

f_2—fraction of ingredients bound to two other ingredients,
f_3—fraction of ingredients bound to three other ingredients, and
f_4—fraction of ingredients bound to four other ingredients.

In addition, the average distance that a cell travels might be another datum collected, as might be information related to the positions on the grid of the different types of ingredients.

2.5.2. Chemical kinetic outputs

In chemical kinetic studies the most relevant attributes are the counts of the various species present and the numbers of transitions of various types that occur during each iteration. For example, in a study of three types of reacting ingredients, A, B, and C, the numbers of each species will change with time, and this variation can reveal important information about the kinetics of the reactions involved. Also informative will be the numbers of transitions, say, from A $\rightarrow$ B to A $\rightarrow$ C, that take place in each iteration.