CHAPTER 3

Maps, Game Theory, and Computer-Based Modeling

As WE SEARCH for deeper understanding of the relation between numbers and models, maps are an appropriate starting point. Maps eliminate detail in a straightforward way and, like games, they are among the earliest model-artifacts. Moreover, our longer-range objective, a general setting for emergent processes, is a kind of map, so a better understanding of maps will help define that objective.

Think first of a simple map, such as a road map (see Figure 3.1). If it is fairly complete, as is true of most state road maps, then the cities, towns, and villages are represented by dots or squares of varying sizes, and the roads connecting these population centers are represented by lines of various colors representing road quality. Some lakes and rivers may be indicated, but in general the map concentrates on population centers and roads. Two kinds of relations are preserved:

1. There is a one-to-one relation between the population centers and the dots on the map. Each city, town, and village is represented by a dot.
2. The dots are arranged on the map in the same configuration as the population centers in the actual geography of the state. That is, larger cities that are close together in the state are represented by large dots that are close together on the map, a town that is close to the state boundary is represented by a

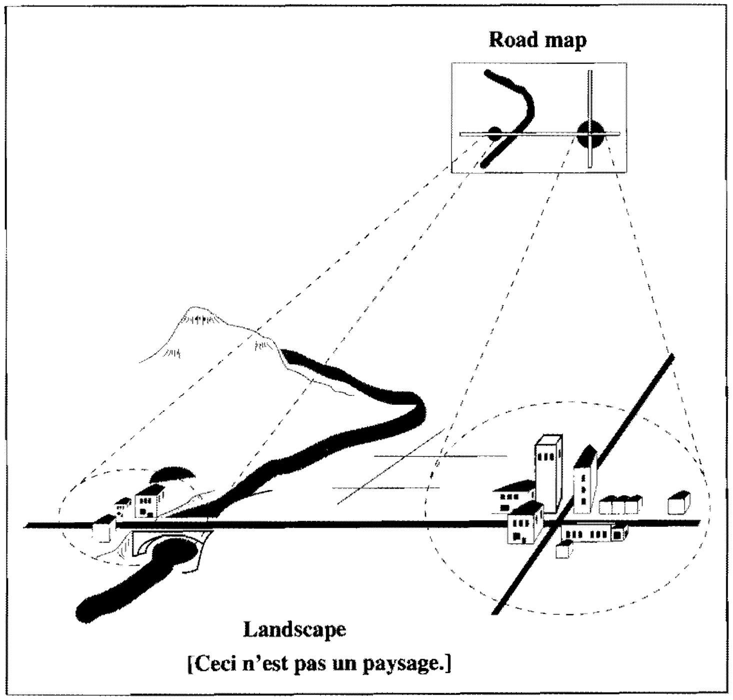


FIGURE 3.1
A road map as a model.

smaller dot close to the edge of the map, and so on. However, all distances have been scaled down, so that cities that are twenty miles apart in reality are separated by two inches on the map. The curves, straightaways, and intersections of the roads are represented on the same scale.

A moment's thought shows that *few* details are retained in this kind of map. We learn little about what we will see at the roadside in driving one of the roads, nor even much about minor zigs and zags in the road (those changes in direction too small to show up at the scale of the map), let alone any details about what the towns look like. What *is* retained is the essential information about getting from one place to another *under normal circumstances.* Road construction or a windstorm can make the route suggested by the map infeasible or impossible.

It is evident that scale plays a major role in the construction of maps. Scale also asserts itself when we extend our view beyond maps to other kinds of models. We at once encounter a whole class of models called *scale models:* scale ships, scale railroads, scale planes. We also expect scale in most statues and representational sculpture, though a monument like Mount Rushmore may be scaled to be *larger* than the original. However, if we look still farther afield, we encounter models in which scaling plays little or no role. Scaling is a special case of a deeper concept, *correspondence.*

We automatically get correspondence when we produce a scaled model, but correspondence is possible without scaling. To construct a model using correspondence, we first select the details or features to be represented, then construct the model so that some part of the model *corresponds* to each selected detail (see Figure 3.1). Think of a cake recipe. It models the steps we actually use to produce a cake. Each step in the recipe (for instance, "add a cup of sugar") corresponds to a complex activity involving a series of physical movements and measurements.

Art Samuel's checkersplayer is a case in point. The correspondence in Samuel's model is between features of the game and parts of his computer program; scale does not enter. For example, corresponding to the "pieces ahead" feature is a set of instructions that actually carries out the counting of pieces. This correspondence between features and computer subroutines will be examined more carefully in the next chapter, after some of the basic ideas are developed here.

Correspondence is best explained with the help of some notation. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a list of details to be modeled, and let $Y = \{y_1, y_2, \ldots, y_n\}$ be the corresponding aspects of the model. Then the correspondence is shown by simply lining up the two sets $\{x_1 \leftrightarrow y_1, x_2 \leftrightarrow y_2, \ldots, x_n \leftrightarrow y_n\}$. In the parlance of mathematics, we have a one-to-one *function,* $f: X \rightarrow Y$, mapping details of the object into aspects of the model. The objects on the left (the x's) are called the *arguments* of the function, and the aspects on the right (the y's) are called the *values* of the function. It is interesting that mathematicians use the term *mapping,* as a technical term, when they are being precise in defining functions. The function concept, or mapping, stands at the center of most of mathematics. Because the construction of a model depends on setting up correspondences, the function concept lets us get at the precise heart of model building. It also lets us bring important mathematical tools to bear in our attempts to model systems that exhibit emergent phenomena.

We need not go deeply into the mathematics to see some bonuses from using functions to discuss models. First of all, we can bring numbers into play, with an increase in clarity and precision. It is one thing to discuss "economic health" in rhetorical terms, such as "nervousness in the production sector"; it is quite another to discuss it in terms of the familiar newspaper chart of changes in "gross domestic product" over time (see Figure 3.2). Such a chart matches dollars, as a measure of productivity, against a sequence of dates. This correspondence of numbers to numbers—a function—has enough precision to allow us to determine trends and make forecasts.

We set up a correspondence between the world around us and numbers any time we read an instrument. The numbers on a tire gauge, for instance, correspond to the tire's inflation. Even a calendar is such an instrument, transforming the passage of time into numbers, as did the newspaper chart. It is this transformation that gives instruments and gauges a central role in experimental science. The instruments make it possible to build numerical models of the phenomena being investigated. Because computers are, above all, number manipulators, such transformations are pivotal in constructing computer-based models.

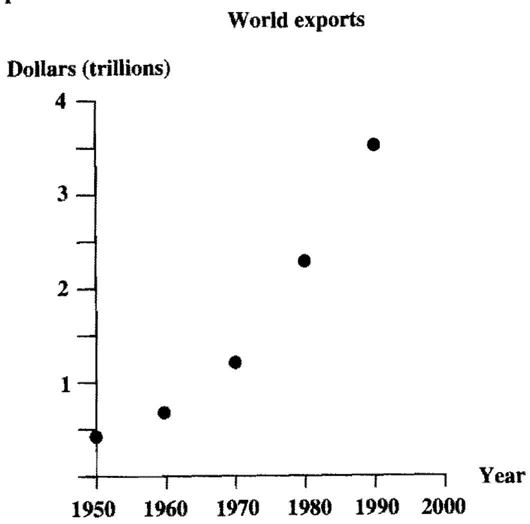
The relation between functions and correspondences also suggests a way of eliminating detail when we construct a model. Features in the checkersplayer are a case in point: *many* boards can share the *same* feature value. For example, there are many boards where the opponent has one more piece than the checkersplayer.

Presentation as a correspondence:

World exports

x (year)	$f(x)$ (trillion $)
1950	0.4
1960	0.7
1970	1.2
1980	2.3
1990	3.5

Graphic presentation:



Presentation as an equation:

$$f(x) = 0.4\times10^{(t/4)} = \text{world exports at } t$$
$$\text{where } t = (x - 1950)/10$$
$$\text{and } x = \text{year}$$

FIGURE 3.2
Functions and correspondence.

This is a *many-to-one* correspondence. The function that defines the correspondence assigns the same number to many different objects. To use the precise terminology introduced earlier in this section, many *arguments* of the function have the same *value.* In model building, a many-to-one function lets us map objects that differ in detail into a single aspect of the model.

Game Theory

It turns out that there is a close relation between maps and games. This is not so surprising given the maplike character of the boards on which many board games are played, but the relation goes deeper than one might initially suspect. This deeper relation, which will be of great help in formulating a general setting, comes clear in the context of game theory.

Though board games are very old, and games of chance have for centuries played a role in the development of mathematical probability theory, it was not until the first half of the twentieth century that a genuine theory of games came into being (von Neumann and Morgenstern, 1947). Game theory, since its inception, has strongly influenced statistics, information theory, and particularly economics, including recent cross-fertilizations yielding evolutionary game theory (see Maynard-Smith, 1978; Axelrod and Hamilton, 1982). The details of game theory fall to one side of our current exploration, but several concepts from the theory serve us well. For our purposes I concentrate on games that are *not* games of chance, such as checkers, chess, and Go, though much of what I have to say is applicable also to games involving probabilities (chance).

States

The first concept is the *state of the game.* For a board game, this state is simply the arrangement of the pieces on the board at any point in the play. From that point on, the play of the game depends only on that arrangement, not on how it was attained. (There are rare exceptions, such as castling in chess or doubling in backgammon, but these can be handled with the help of an auxiliary "piece," as when a doubling cube is used in backgammon.) In short, the state of the game at any point in the play is a sufficient summary of past history for determination of all future possibilities. In this the state of the game is closely related to the state of a physical system. For instance, we record the state of a container of gas under pressure (a tire or a scuba tank) in terms of its pressure, its temperature, and its volume. If we puncture that container, what happens next is determined by that state. When the state of a system is correctly defined, its future dynamics depends only on its current state.

The *state space* of a board game is simply a collection of all arrangements of the pieces on the board that are allowed under the rules of the game (see Figure 3.3). The qualification "allowed under the rules" is important (see Figure 3.4). In chess, the pieces can be arranged on the board in many ways, but only a small fraction of the arrangements are attainable under the game's rules. For example, the rules of the game require that the piece called a bishop always move to a square of the same color as its starting square (it only moves diagonally on the checkerboard). Moreover, the bishops on a given side start on different colors, so we know immediately that any configuration with these two bishops on the same color is illegal. More carefully: a board game starts with an initial arrangement of pieces specified by the rules; a *move* occurs when the pieces are rearranged under the rules—often the movement of a single piece. Successive moves determine the play of the game. The set of all arrangements (states) that can be attained under the rules is the game's state space (see Figure 3.5).

Tree of Moves

The most important concept from the theory of games, for our purposes, is the *tree of moves* (see Figure 3.5). The *root* of the tree is the game's initial state, the first branches lead to the states that can be attained from the root, the branches on those branches lead to

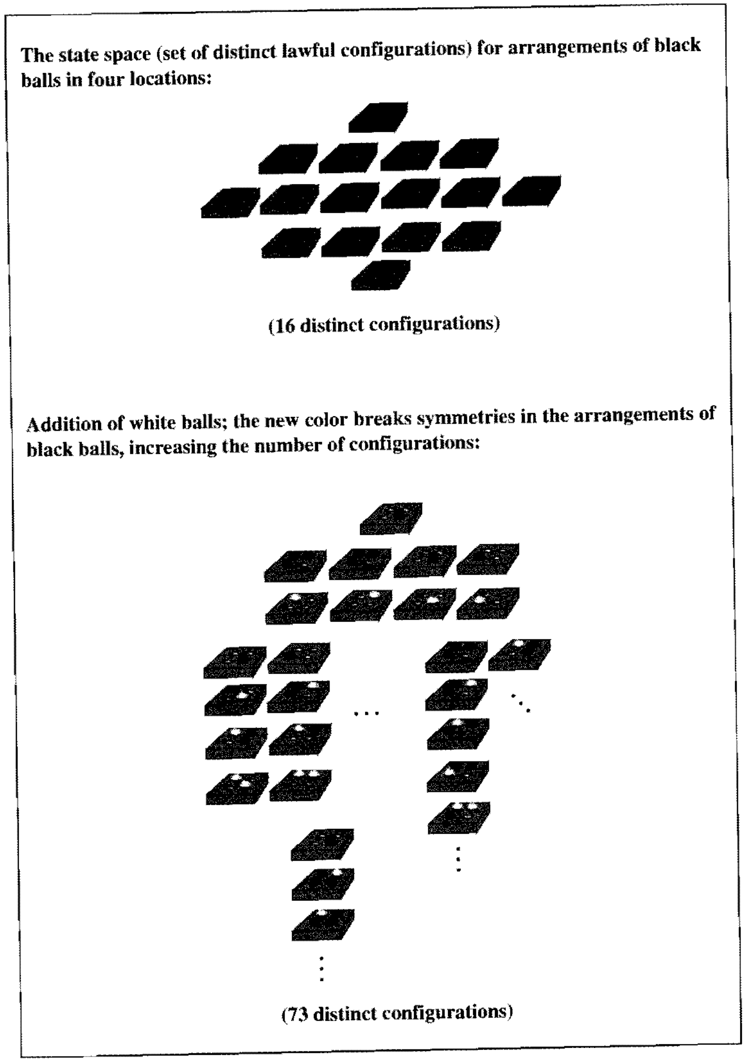


FIGURE 3.3
Some simple state spaces.

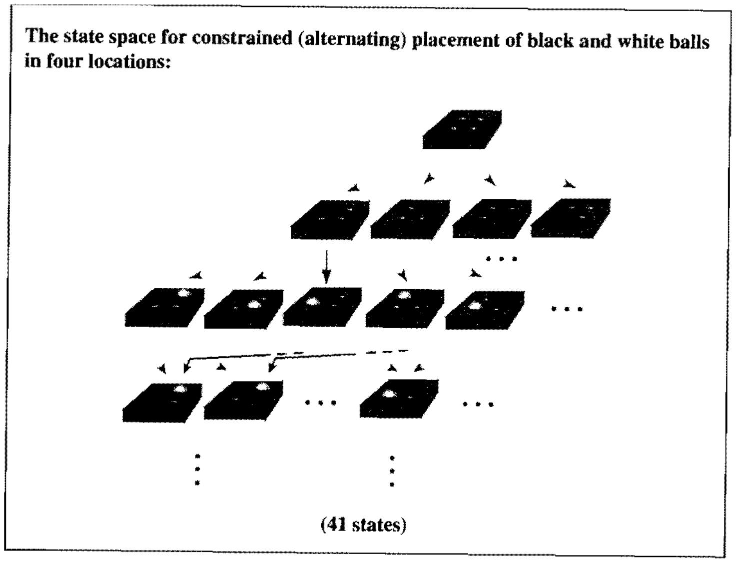


FIGURE 3.4
Legal configurations.

the states that can be attained in two moves, and so on to the *leaves* of the tree, which are the ending states. The leaves determine the outcome of the game. It is the succession of choices allowed on the way to a leaf that makes the game interesting.

For real board games, in contrast to some games invented for theoretical purposes, the result is somewhat more convoluted than a tree. In games, unlike trees, different branches may end up at the same state, so there may be fewer states than there are branches. We can move a castle and then a bishop and finish in exactly the same configuration as if we had moved the bishop first and the castle second. In particular, many branches may wind up at the same leaves (end points); in chess, many different lines of play can end with the king checkmated in the corner by a queen and a castle. This additional complication doesn't much affect the present discussion, but I will talk about "ways of playing the game," (instead of about leaves) as a way of indicating this peculiarity.

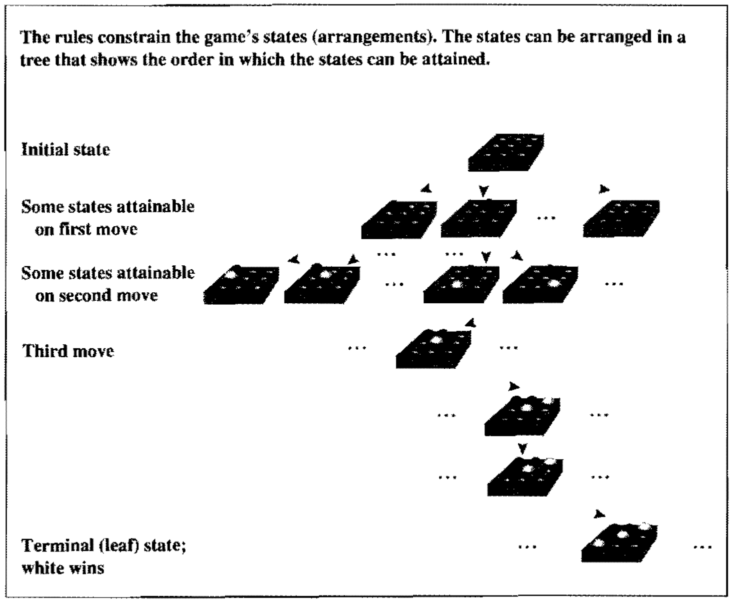


FIGURE 3.5
Part of a game tree for tic-tac-toe.

All in all, games are more bushes than trees. The number of leaves (ending configurations) grows *very* rapidly, even when the branching process is simple. Indeed, it is this bushiness that provides the fascination and unpredictability of games. Consider a board game in which there are ten possible moves (branches) from each configuration (state), including the initial configuration. If the game terminates after two moves, there are $10 \times 10 = 10^2 = 100$ distinct ways of playing the game. If the game terminates after ten moves, there are $10^{10} = 10{,}000{,}000{,}000$ ways of playing the game. Termination after fifty moves—a length and number of options roughly equivalent to chess—yields 10^{50} ways of playing the game, a number which substantially exceeds the number of atoms in the whole of our planet Earth.

We begin now to see that a small number of rules can define a game so complicated that we will never exhaust its possibilities. If we had a record of all the games of chess that have been played over the centuries, it is doubtful that any two would have identical move sequences (setting aside those that either terminated early or were deliberate replays of annotated games). It is this perpetual novelty that makes chess and Go classic games that continue to challenge humans after centuries of careful study. By the same token, tic-tac-toe remains a children's game because its possibilities are quickly exhausted, once certain patterns are recognized.

Strategies

In any game that is at all complex, a game plan or strategy is vital for effective play. Roughly, a strategy is a prescription that tells us what to do as the game unfolds; it specifies a sequence of decisions. The game tree provides a way of making this rough idea precise. A sequence of decisions made during the play of a game

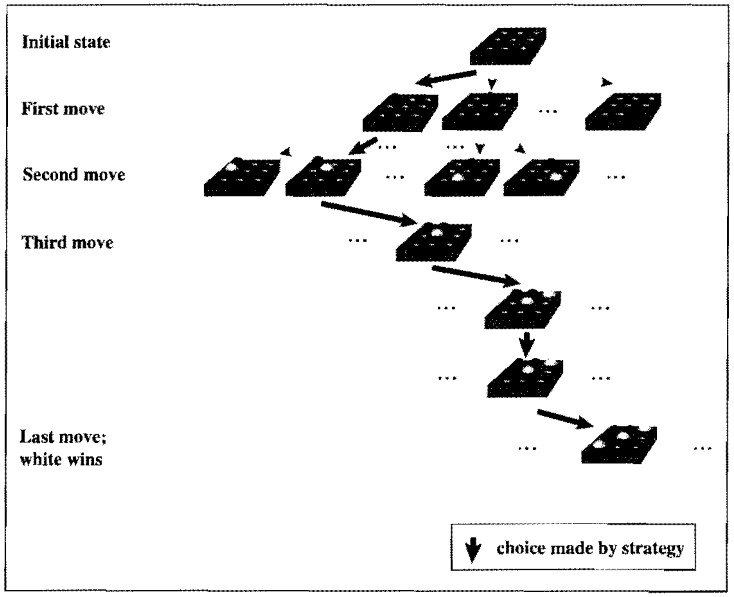


FIGURE 3.6
Opposing strategies determine a path in the game tree for tic-tac-toe.

traces a path in the game tree (see Figure 3.6). So we can define a strategy in terms of the branches it chooses in the game tree. In game theory a *complete strategy* prescribes a branch (move) for *each* state (board arrangement) that can be encountered. In other words, *a complete strategy tells us what to do in any possible situation.* Note that a strategy may be good or bad. It is simply a prescription for what to do; it could be a surefire prescription for losing.

Here is another point where functions are useful. We can use a function to define the correspondence between game states and the moves prescribed by a strategy. The function first assigns a move to the initial state (say, "move the pawn that is fourth from the left ahead one"); it then assigns moves to each of the states that can result from the opponent's response to that move, and so on to the end of the game. For every strategy a function exists that describes that strategy.

> More formally, a particular strategy g prescribes, for each state s in the set S of all game states, the decision (move) to be made at that point in the game tree. This is the same as saying that g prescribes the follow-on state s' in the game tree for each state s. For each state s, the strategy g is constrained to select only branches that lead from s. (It may be that a given player will not encounter some states in S, but then the prescription can supply a meaningless "dummy move.") In brief, $g(s) = s'$. The strategy, then, is a mapping $g{:}S \rightarrow S$, where the options for the map are constrained by the set of moves that are legal for each state s in S.

In a game with two or more players, a *multiperson game,* we can attribute a strategy to each player. Once each player fixes on a strategy, the outcome (leaf of the game tree) is determined (setting aside strategies that use random move choices, say by the roll of dice). To put it another way: the combined strategies select a path through the move tree that leads from the root node to a particular leaf (see Figure 3.6).

When the players have chosen strategies at the outset, it would seem that all the interest and surprises have been removed from the game. All that apparently remains is a kind of mechanical playing-out to reach the predetermined end. But this reasoning omits a factor: the players do not know their opponents' strategies. Each player has decided what to do in each contingency, but each player has no idea what particular contingencies will arise because of the other players' actions. So the individual player *cannot* predict the final outcome, or indeed the outcome of the first few moves, even though that outcome is predetermined. For each player the game will take unexpected twists and turns.

When a game is played repeatedly, the unknown aspects of the other players' strategies may become clearer. Consider a two-person game where the opponent has fixed on a particular strategy. Observing the opponent in repeated plays of the game can tell us what the opponent does at different branch points (choices) in the game tree. We can use this information to build a model of the opponent's strategy. The resulting model will lack many details, because there are just too many possible strategies to uncover a complete description through "trial and error." Nevertheless, if the model is correct in some respects, we can do better with it than without it.

These observations apply to "games" much more general than board games. Consider a game in which one of the opponents is "nature," as when we try to execute a plan (strategy) for enhancing an ecosystem (nature). The outcome may be difficult to predict, even if nature obeys a fixed set of rules (laws). Still, through observation of the effects of choices over time, we can begin to build a model of the ecosystem and its responses. Much scientific endeavor takes this form.

Two major themes emerge at this point:

1. In realistic situations, a strategy *cannot* be defined by listing all the game states with the moves prescribed for each state. Too many states exist in even a modest game to make such a list feasible. This is so even when we take into account the storage capacity and speed of the largest computers; the number of states we calculated earlier for move trees is so large that no foreseeable computer could store them. The recent chess-playing programs provide a direct example. Even with the tremendous storage capacity and speed devoted to them, they do not attempt an explicit strategy. Rather, the programs are highly selective in the way they search the tree, changing the search as the game unfolds. They test only a minuscule fragment of the whole.

> To say the same thing in mathematical terms, we cannot define the strategy function *explicitly* by listing all (state, move) pairs, $(s, g(s))$, for all states s in S.

Instead of an explicit definition, we define strategies in much the same way we define games, via a set of rules. The rules in the case of strategies are usually rules of thumb. For example, in chess these rules embody principles like "Build a strong pawn formation," "Control the center," "Look for 'fork' attacks," and so on. Such rules pick out game features that occur frequently and are relevant to decisions at various points in the game. In so doing, they group states into clusters, where the states in a given cluster have a feature that suggests similar decisions or moves. In this way we obtain an effective reduction of the enormous size of the game tree and make possible an overall prescription that controls play throughout the game. The extended discussion of Art Samuel's checkersplayer in the next chapter will show how this is done.

From this perspective, we use repeated plays of the game to discover and combine building blocks (rules of thumb, features) in order to construct a feasible strategy. The task is much less daunting than trying to define a strategy explicitly as a list of states and prescribed moves. Even if the strategy has some components that are not easily described in terms of building blocks, it is a valuable starting point for modeling the strategy. This point of view also suggests we assume that the opponent's strategy is constructed from a limited set of building blocks.

If nature is the "opponent"—if we are taking the role of scientists—we do much the same. We attempt to model the rules of the universe, though with less reason to believe that the opponent is restricted to feasible strategies. (Einstein's *cri de coeur,* "Quantum mechanics is very impressive, but I am convinced God does not play dice," is clearly an expression of faith, not an observation.) In science, as in games, part of the justification for making this assumption about building blocks is that it works. Newton's laws, Maxwell's equations, Mendeleev's periodic table, and Mendel's genes all tell us a great deal about the way the world operates.

Note that surprises are in store, even if we should be able to uncover a set of fixed laws that determines all of nature's possibilities. After several centuries we still uncover new possibilities inherent in Newton's equations, and this set of laws most assuredly does *not* encompass all of nature's possibilities.

2. Our simplifying assumption to this point has been that opponents employ fixed strategies, but this simplification sidesteps most of what happens when games are played repeatedly. Opponents learn. A more realistic view is that *all* players are *simultaneously* trying to build models of what the other players are doing. Under this extension, the situation becomes much more complicated. An observer who has an omniscient overview of the game encounters surprises akin to those encountered by individual players. Even if that observer knows the initial strategies and the details of the individual learning procedures, it is next to impossible to predict the course of the game. Emergence and perpetual novelty are ever present in games where the opponents are adapting to each other.

Emergence—A First Look

This view, of opponents adapting to each other's strategies, encourages a more careful look at emergence in rule-governed systems. A computer, once supplied with the rules of a game, and the rules that determine the players' strategies and changes in strategy (setting aside chance moves), can, move by move, determine the course of the game. So the overall system is fully defined. Despite this, an outside observer will be hard put to determine what happens next, even after extended observation. The strategies coevolve in the computer, each strategy adjusting to its experience with its opponents. This coevolution exhibits the creativity we expect of any evolutionary process; the computer is continually getting into parts of the move tree not previously observed. Species of play rise to dominance and disappear, players mimic each other, and so on. What, if anything, shows the regularity and predictability we expect of emergent patterns?

Though prediction is difficult in these circumstances, it is not a hopeless task. Everything depends on the level of detail we require of the prediction. Meteorology provides a useful simile. Weather patterns are never the same in detail, and even the larger features, such as fronts, cyclones, jet streams, and the like, show a remarkable diversity. Moreover, weather models do not yield exact predictions of such obvious events as the amount of rainfall to expect locally on the morrow. Nevertheless, modern weather prediction is very helpful. It does forecast the likelihood of rain and severe storms, it does give the likely temperature ranges, and the five-day forecasts of average temperature and rainfall are much more accurate than could be attained by simply using past statistics for that time of year.

Chaos theory is often cited as an explanation for the difficulty of predicting weather and other complex phenomena. Roughly, chaos theory shows that small changes in local conditions can cause major changes in global, long-term behavior in a wide range of "well-behaved" systems, such as the weather. In an oft-cited example, the flapping of a butterfly's wings in Argentina can (eventually) cause worldwide changes in the weather. There is a sense in which this is true: if we knew *all* the values for *all* the relevant variables worldwide, à la Laplace (see Singer, 1959), we could predict the weather indefinitely far into the future. With such a model we could determine the long-term weather pattern with and without the flapping of the butterfly's wings. We would see that the two weather patterns would eventually diverge to a point of no correlation.

This explanation ignores important factors in real weather prediction. Because meteorologists do *not* know the values of all the relevant variables, they do not work at a level of detail, or over time spans, in which chaos would be relevant. The predictions work with large masses of atmosphere over short time spans; so butterflies, or jet airplanes, produce negligible effects. Moreover, rather than trying to develop predictions based on remote initial conditions, as with the butterfly effect, meteorologists start anew each day, using the most recent data. These observations continually bring the state of the model into agreement with what has actually occurred. Under this regime chaos theory has little relevance.

The key to effective weather prediction, then, is the discovery and use of the mechanisms (building blocks) that generate weather. This approach originated with the discovery of *fronts* by the Norwegian meteorologist Vilhelm Bjerknes in the early part of the twentieth century. Curiously, Bjerknes lived in Bergen, Norway, where weather prediction is remarkably easy throughout most of the year—it rains! The model that Bjerknes originated has been progressively improved, through the use of more sophisticated mechanisms (and equations) governing fluid flow, the discovery of jet streams, and the recognition that distant large-scale phenomena, such as the Pacific High, can be used to guide long-term predictions. Computer-based models, strongly advocated at the very beginning of the computer era by von Neumann (see Korth, 1965), significantly advanced both the detail and the time span of weather prediction.

Thus, complexity and even chaotic effects need not forestall our study of emergent phenomena. The key to deeper understanding, as with weather prediction, is to determine the level of detail and the relevant mechanisms. At the right level of detail, the model's changing states play the role of the configurations in a game. Using mechanisms as building blocks, we can construct models that exhibit emergent phenomena in much the way that interacting strategies in a game produce patterns of interaction not easily anticipated from inspection of the game's rules. The mechanisms play the role of the game's rules, setting limits to what is possible while providing extensive combinatoric possibilities.

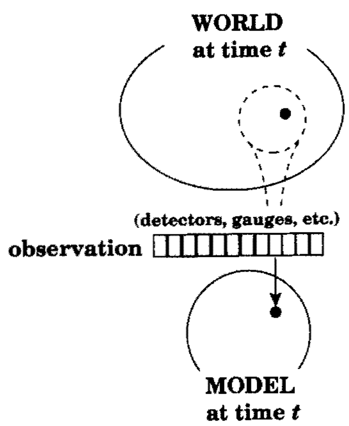
Even when we have the right level of detail and the relevant building blocks, perpetual novelty is still typical. As in games, though the definition is simple, the state space for models of complex systems is very large. And, as in games, the model rarely or never returns to states already visited. This perpetual novelty renders it difficult to make predictions, even when the mechanisms (rules) and the initial state are specified. If the basic mechanisms provide for learning or adaptation, the difficulty increases enormously. Still, by attending to selected details, we can usually extract recurring patterns, like fronts, in the complex unfolding sequence. When these recurring patterns are regularly associated with events of interest, we call them *emergent* properties. We will look much more closely at the prediction of emergence—the "when," "where," and "what"—once we have a general setting in place.

Dynamic Models

In the previous discussion, we have moved from models that have static forms, such as scale models, to models with changing configurations, usually called *dynamic models*. The object in constructing a dynamic model is to find unchanging laws that generate the changing configurations. These laws correspond roughly to the rules of a game. In a game, the rules say how the configurations (states) change as different moves are made; the players affect the course of the game by choosing moves. When we consider the weather system, we usually think of it as autonomous, proceeding without (or despite) our intervention. Still, the laws of change specify the succession of states—the weather configuration eight hours from now, twenty-four hours from now, and so on. If we had effective means of weather control, then the laws of change would specify how those controls affect the unfolding weather sequence.

To build a dynamic model we have to select a level of detail that is useful, and then we have to capture the laws of change at that level of detail. There are potential conflicts. It may be quite difficult to construct a detailed model that is "faithful" to the system being modeled. Weather prediction models provide instructive examples. Predicting that the atmospheric temperature will be less than the boiling point of water may be reassuring, but it is not much of a weather prediction. We want more detail, but then we have to deal with laws of change that involve fronts, jet streams, and the like.

There is, of course, no guarantee that we can find simple laws of change for the level of detail selected. Indeed, the art of model building turns on selecting a level of detail that admits simple laws—a point to which we'll return in later chapters. Setting the level of detail turns mostly on defining the model's *states* (see Figure 3.7). For games, we defined the state of a board game as the configuration of pieces on the board; for Bjerknes' weather model, the current state is the configuration of fronts, jet streams, and the like on the weather map. For dynamic models in general,

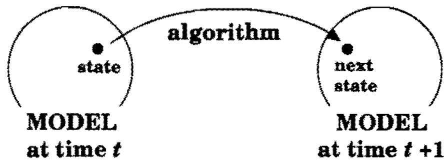


World configurations producing the same readings are lumped into a single equivalence class represented by a single state of the model.

FIGURE 3.7
Observation and state of the model.

the features and details incorporated in the model's states determine the level of detail.

Once we define the model's states, our object is to define the laws of change that work at this level. Laws of change are stated precisely with the help of a *transition function* (see Figure 3.8). The transition function assigns to each state the state that will occur



The model's transition function, often specified via partial differential equations or computer algorithms, captures the notion of causality.

The relations of world configuration and observation to the states and transition function of the model are summarized by the following diagram:

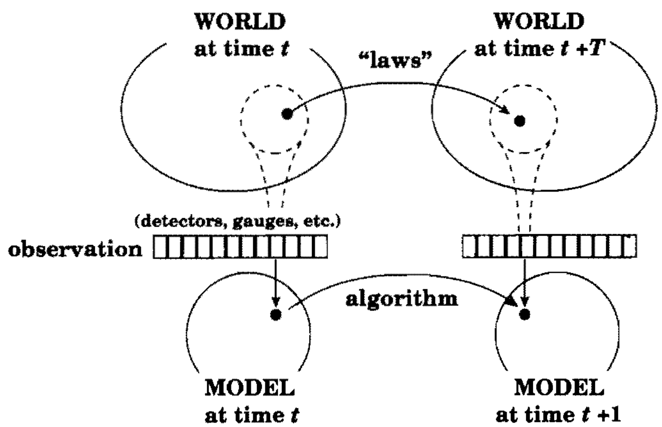


FIGURE 3.8
Transition functions.

next under the laws of change. Where the course of change of the system can be affected from "outside"—where the system can receive inputs from the outside—the transition function provides a different correspondence for each input to each state. That is, different inputs cause different next states, so the transition function provides a correspondence between each possible (state, input) pair and the state that results. The transition function is reminiscent of the function that defines a strategy, where moves are the counterparts of inputs. Newton's equations again provide an example: they define the dynamics of gravity via a transition function that relates mass and acceleration (states of a particle), and force (an input).

If the transition function (law) is "faithful," we can make predictions into the indefinite future. Knowing the current state and input, we can determine the next state. Then, knowing that state and the next input, we can determine the state after that, and so on indefinitely. Therein lies the great advantage of a faithful formal model: by simply iterating the use of the transition function, we can explore future possibilities. The transition function determines the future fully and unambiguously (if the inputs are known). The only uncertainty resides in the appropriateness of the level of detail and in the faithfulness of the transition function. That is, the uncertainty lies in the model's *interpretation*, the mapping between the world and the model.

This capacity for prediction provides the deep connection between modeling and emergence. The (usually simple) specification of a model—the transition function—can yield a limitless array of consequences and predictions. A well-conceived model can, like chess, yield organized complexities that repay decades and centuries of study. Moreover, these complexities may involve possibilities not conceived by the modeler, as when Newton's model is used to guide rockets to Mars and to determine the evolution of galaxies. As with Jack's magic seed, Newton's model opens worlds of wonder that transcend the simplicity of the starting point.

The idea of "faithfulness" takes us from a game, which may only

remotely reflect the world, to a model that correctly reflects selected aspects of the world. Interestingly, faithfulness in this sense can be given a simple, precise definition: we say a model is *perfect* if the interpretation satisfies a criterion called *commutativity of the diagram* (see Figure 3.9). Commutativity holds when the order in

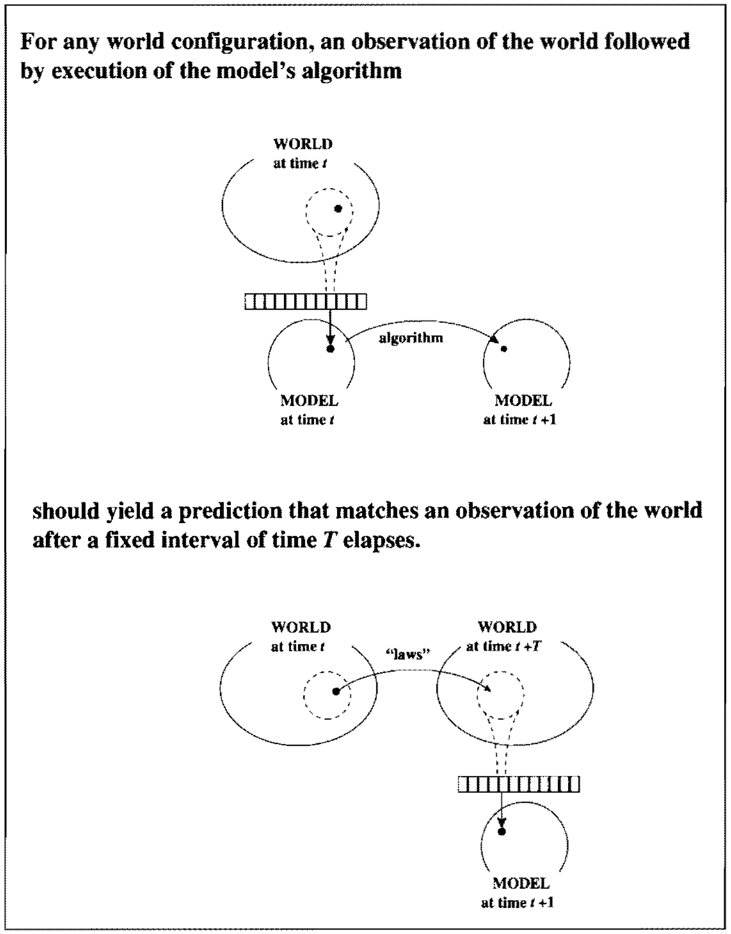


FIGURE 3.9
Commutativity of the diagram—a perfect model.

which we do things is irrelevant to the result. If we take a step right and then a step down, we arrive at the same place as if we take a step down and then a step right. Addition is commutative: 5 + 3 = 3 + 5. To use this idea for models, we set up a diagram that shows, in its upper half, one time-step of change in the world (say, possible changes in the weather in an eight-hour period). In the lower half of the diagram we show a one-time step change in the model, under its defining law of change (transition function). We can observe the world now (left side of the diagram) or later (right side). In either case that observation determines a state for the model. Commutativity of the diagram holds when, *for every state,* taking the observation (down) and then executing one time-step in the model (right) yields the same result as waiting one time-step in the world (right) and then taking an observation (down). That is, the model's law of change correctly predicts the result of a future observation. Because this definition holds for all states, we can iterate the process to get predictions indefinitely far in the future, as suggested earlier. Of course, we can only sample the states of the world, even if we are only interested in a sharply delimited region (for instance, an experiment). So in practice we can only approximate the "all states" requirement for a perfect model. Nevertheless, the concept of a perfect model supplies a valuable guide for constructing useful models.

Computer-Based Models—A Closer Look

I have mentioned computer-based models several times, and they have a critical role in the construction of dynamic models. They have become ubiquitous in modern science, being used to model everything from the spread of epidemics to fusion in the sun. It will help us to understand dynamic models if we look at computer-based models a little more closely now, though still in preliminary fashion. Earlier I asked how we can use numbers, and coordinated changes in numbers, to simulate the flight of a jet over Chicago in a thunderstorm. I want now to give a partial answer to that question.

The starting point, once again, is the notion of *state.* The natural question is, "What can we possibly mean by the state of a jet airplane flying over Chicago?" The answer is closely connected to the information the pilot uses to fly the jet.

To get at this connection between information and state, let me start with a simpler system: the control panel of the family car. The car's control panel is not in principle much different from that of the jet, it is just much, much simpler. It tells us only the essentials that we need to know when driving, typically the speed of the car, the fuel level, the engine temperature, the battery charge, and the oil pressure. These readings model the state of the car, at a certain level of detail, when it is under way. We could add more readings, such as the air pressure in the tires or the amount of antifreeze in the radiator, to get a more detailed state. This more detailed state would provide the wherewithal for a more sophisticated model; however, decades of experience have shown that the gauges first mentioned are sufficient for operating the car in most situations.

Because the jet is far more complicated, the pilot's compartment is filled with a panoply of displays, gauges, dials, and warning lights that provide information about the conditions that affect the jet's flight. They tell about the plane's speed and position, the amount of fuel in its various fuel tanks, the operating condition of the engines, the position of the landing gear, and hundreds of other bits of data. Indeed, there is enough information for the pilot to fly the plane "blind," using instrument readings alone.

For both the car and the jet, the displays and gauges produce readings that either are numbers or are easily reduced to numbers. A warning light can read either on or off, which can be represented as a 1 or a 0, and even the sophisticated positional display is presented by an array of dots (called pixels), which can be represented as an array of 1's and 0's. In other words, it is easy to reduce the information on the control panels to numbers. These numbers can, as usual, be stored in registers in the computer. Together they define the state of the model, much as the arrangement of pieces defines the state of a board game.

We give the computer a representation of the state of the model by entering these numbers into the storage registers. Then we enter instructions (a program) that cause these numbers to change over time as specified by the transition function. This is the counterpart of defining the rules of the game. The numbers in the registers change in a way that mimics the state changes in the object being modeled. The *universality* of the general-purpose computer assures that any transition function defined by a finite number of rules can be so mimicked.

As in a game, we now confront the notion of choice. The driver or the pilot can choose among alternatives, such as making the car or jet go faster or slower. Phrased in terms of states this means that, once again, from any state we can construct a tree of legal alternatives. In a game, these alternatives were the legal moves allowed by the rules. In the case of the car or the jet, the laws are those imposed by nature and the technology. Executing a sequence of controlling actions is the counterpart of making a sequences of moves in a game. In both cases we choose a path through the tree of possibilities.

When both the numbers and the program have been stored in the computer, we simply start the computer executing its instructions. Think again of a video game or a flight simulator. The instructions, acting on the stored numbers defining the model's state, determine what happens instant by instant. What we see on the screen is a back-translation of the numbers to gauge readings, displays, and so on, that capture the look and feel of the original machine. Controlling actions amount to input to the program at various stages of the calculation. The input is supplied by typing, or by the video game's joystick, or by realistic controls in a full-fledged flight simulator. The result is a dynamic, computer-based model—a major vehicle for the scientific investigation of emergence.

CHAPTER 4

Checkers

ANY SERIOUS STUDY of emergence must confront learning. Despite the perpetual novelty of the world, we contrive to turn experience into models of that world. We *learn* how to behave, and we anticipate the future, using the models to guide us in activities both common and uncommon. Somehow, through learning, these models emerge from the torrent of sensations that impinge upon us at every moment. Certainly, a deeper understanding of learning will contribute to a deeper understanding of emergence. In attempting to understand the relation between learning and model building, we could scarcely find a better starting point than Art Samuel's mechanization of learning in the checkersplaying program.

It is strange that, until recently, Machine Learning has been a sideshow in Artificial Intelligence (AI)—strange, because most would say that an organism that does not learn is not intelligent. Nevertheless, for most of its history, AI has placed work on learning at the periphery of its activities. Samuel's 1959 work on checkersplaying, and the work on cyclic neural nets (Rochester et al., 1956), both completed almost a half-century ago, still lie close to the cutting edge of research in Machine Learning. Because the two efforts took place near the dawn of the computer age, they have a stark construction, unencumbered by elaborate concern with computer languages, interfaces, and the like. For this reason, they make a meaningful starting point for a close look at the computer-based models that underpin Machine Learning. We will delve into Samuel's checkersplayer in this chapter, and will devote the next chapter to cyclic neural nets. Although these two