1 What Is Agent-Based Modeling?

Go to the ant, O sluggard; consider her ways, and be wise.

—Proverbs 6:6

Ants

The ant opens her eyes and looks around. There are many of her siblings nearby, but there is no food. The ant is hungry, so she heads out from the ant colony and starts to wander around. She sniffs a little to the left and a little to the right, and still she cannot smell any food. So she keeps wandering. She passes by several of her sisters, but they do not interest her right now; she has food on her mind. She keeps wandering. Sniff! Sniff! Mmmm … good! She gets a whiff of some of that delightful pheromone stuff. She heads in the direction of the strongest pheromone scent; in the past there has been food at the end of that trail. Sure enough, as the ant proceeds along the trail, she arrives at some delicious food. She grabs some food and heads back to the colony, making sure to drop some pheromone along the way. On the way back, she runs into many of her sisters, each sniffing her way along pheromone trails, repeating the same process they will carry out all day.

What we have just fancifully described is an ant foraging for food. The opening paragraph of this section is in itself a model of ant behavior. By a *model*, we mean an abstracted description of a process, object, or event. Models can take many distinct forms. However, certain forms of models are easier to manipulate than other forms. The textual description in our first paragraph cannot easily be manipulated to answer questions about the ant colony's behavior—for example, what would we see if all the ants in the colony followed the behavior we described? It is difficult to extrapolate from the textual description of one ant to a description of an ant colony. The textual model is not sufficiently generative to answer such questions. It is a fixed description—always "behaving" the same, thus not bestowing any insight into the range of variation in behaviors. And it is not combinatorial—we cannot use the description to understand the interaction of the ants with each other or with the environment.

One way to make the preceding model more generalizable and gain insight into the behavior of an ant colony would be to implement the model in a computational form. A *computational model* is a model that takes certain input values, manipulates those inputs in an algorithmic way, and generates outputs. In computational form, it would be easy to run the "Ants" model with large numbers of ants and to observe the model's outputs given many possible different inputs. We use the term *model implementation* to refer to this process of transforming a textual model[1] into a working computational simulation (written in some form of computer "code"). Besides textual models, there are other forms of *conceptual models* that describe processes, objects, or events but are not computational; conceptual models can also be diagrammatic or pictorial.

This description lends itself particularly well to being implemented, since it results from a particular standpoint, that of an individual ant, or ant *agent*. By the word *agent*, we mean an autonomous individual element of a computer simulation. These individual elements have properties, states, and behaviors.

The ant is an agent. It has properties such as its appearance and its rate of movement. It has characteristic behaviors such as moving, sniffing, picking up food, and dropping pheromone. It has states such as whether or not it is carrying food (a binary state) and whether it can sense how much pheromone is in the environment around it (a multivalued state).

Agent-based modeling (ABM) is a computational modeling paradigm that enables us to describe how any agent will behave.[2] The methodology of ABM encodes the behavior of individual agents in simple rules so that we can observe the results of these agents' interactions. This technique can be used to model and describe a wide variety of processes, phenomena, and situations, but it is most useful when describing these phenomena as *complex systems*. Our aim in this textbook is to facilitate your creating, modifying, and analyzing the outputs of agent-based models. We begin by describing the genesis of the ant foraging conceptual model and how it was implemented as an agent-based model.

Creating the Ant Foraging Model

Many biologists and entomologists have observed ants in the wild (Hölldobler & Wilson, 1998; Wilson, 1974) and have described how ants seemed to form trails to and from food sources and their nest. In the next few paragraphs we will describe some hypotheses about how the ants accomplish this behavior and what mechanisms are at work that enable ants to find food in this way.[3] Perhaps the trails form as follows: After an ant finds food, it goes

1. Or, more generally, a conceptual model in any form.

2. We will use the acronym ABM in several ways in this textbook. Sometimes we will use it to refer to the practice of agent-based modeling. Other times, we will use an ABM to refer to an agent-based model, or ABMs to refer to agent-based models. We rely on context to disambiguate our use of the term.

3. Our account of ant behavior is inspired by the actual history of the scientific study of ants. It has been simplified here for the sake of exposition. For a more detailed account, see Theraulaz and Bonabeau (1999).

Box 1.1
Complexity and Complex Systems

The study of complex systems has become an important scientific frontier. By the term *complex system* we mean *a system composed of many distributed interacting parts*. The field of complex systems or complexity science arose in the mid-1980s and was born out of a variety of disparate fields from economics to physics to ecology. Complex systems science provides a set of tools and frameworks for viewing phenomena. Any phenomenon can be viewed as a complex system, and choosing when to do so depends on assessing when it is most useful to use the lens and/or methodologies of complex systems. One important methodology of complexity science is agent-based modeling. The history of agent-based modeling and complex systems is explored in greater depth in subsequent chapters, and its roots in computer science are described in the appendix.

back to the nest, drops the food off, and communicates to the queen ant, and this queen ant tells the other ants where the food is and sends them off to collect it. Suppose a researcher notices that the food-finding ant never communicates with any kind of "boss" ant before leaving the nest again, so the researcher might reason that food gathering was not working through a centralized leader or controller but rather through distributed control (Bonabeau et al., 1999; Deneubourg et al., 1990; Dorigo & Stützle, 2004).

Another hypothesis that could be advanced is that perhaps the ant did not communicate with a central leader, but rather simply communicated with other ants, and those other ants were able to "diffuse" the information about the food source throughout the nest. There is reason to believe such a hypothesis might be true, since bees work in a similar way; when a bee finds a food source, it returns to the hive and conducts a complicated dance that tells the other bees where to find the food (Gould & Gould, 1988). However, though ants do have some methods of communicating information directly (Hölldobler & Wilson, 1998) most species do not communicate the exact route to a food source in this way. In fact, scientists rarely observe any difference in behavior between an ant returning to the nest with food and an ant returning without food. Most ants returning without food simply leave the nest again and recommence their search for food; the ants act almost exactly like they had before they found food. From this it is possible to reason that there must be some communication method between the ants or the colony could not efficiently gather food. In the mid-twentieth century, biologists such as Wilson (1974) found that ants with food do act slightly differently from ants without food. Ants with food drop a chemical pheromone onto the ground as they are carrying their food. This pheromone interacts with the environment by diffusing and evaporating.

Maybe this pheromone is the key to ant communication? An ant that finds food communicates where the food is by depositing a pheromone into the environment, and the

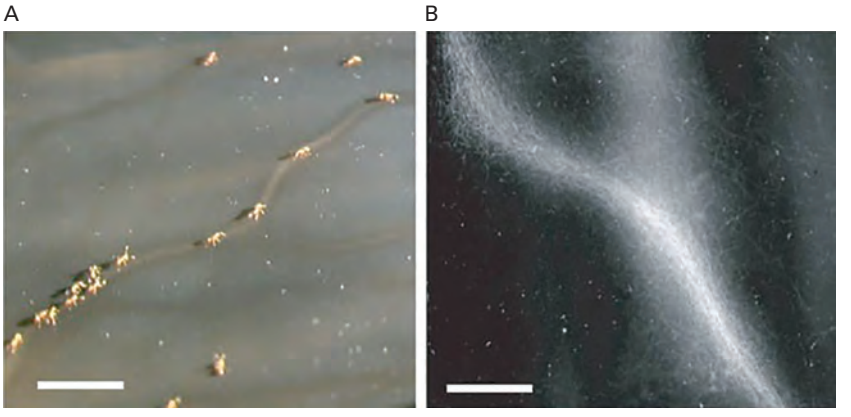

Figure 1.1
Ant trails. Pheromone trail networks of pharaoh ants on a smoked glass surface. (A) Part of a network showing bifurcations to smaller trails (scale bar: 1 cm). (B) Close-up of a single bifurcation (scale bar: 0.5 cm).[4]

other ants follow the gradient of this pheromone trail back to the original food source.[5] This would explain not only the trails that the ants form but also additional phenomena that can be observed. For instance, if ants communicated directly about food sources (as in the first two hypotheses), then any ant would have to communicate with the ant that had discovered the food source, or communicate with an ant that had communicated with that ant. However, the pheromone hypothesis does not require such a complex network of communication. Essentially, the new ant simply picks up the pheromone trail in midstream and is able to follow it correctly to the food source. No direct communication between ants is required. It also explains why the ants do not go directly to the food source (i.e., they do not make a *bee line* as bees do, where they fly directly toward the food source); instead, ants tend to follow roughly along the trail. This is because the pheromone trail can get weaker and stronger in different places as it is placed on different surfaces and in different exposures. (See figure 1.1.)

All of these descriptions require that the ants have knowledge of how to return to their nest directly regardless of how much they have wandered away from it. A combination of both experimental evidence and computational models have confirmed that ants can determine the most direct route back to their nest guided by the sun, outlines of the sky, or magnetic north (Wittlinger et al., 2006; Hartmann & Wehner, 1995; Lent, Graham & Collett, 2010).

4. Jackson et al., 2004. Adapted by permission from Macmillan Publishers Ltd: NATURE.

5. The process of communication where information is transmitted by altering the local environment is called *stigmergy* (Grasse, 1959).

Now that we have a basic hypothesis (in the form of a textual model) that describes the behavior of the ant, how do we implement the model so that we can test out this hypothesis and see if our computational model adequately accounts for what we observe in nature?[6] The first step is to create a more algorithmic description of the preceding textual model. This is just another model itself, but one that is more easily translated into an implemented model. Here is one set of rules that an individual ant could follow in order to operate in accordance with the model described earlier. We describe the rules from the point of view of an individual ant.

1. If I am not carrying food, I check if there is food where I am; if there is, I pick it up; if there isn't food right here, I try to sense a pheromone trail nearby; if I find one, then I face "uphill" along the pheromone gradient toward its strongest source.
2. If I am carrying food, I turn back toward the nest and drop pheromone on the ground below me.
3. I turn randomly a small amount and move forward a step.

These rules are easily implemented in a computer language. There are many computer languages in use for many purposes, but only a few are especially tailored to work with agent-based modeling. One of these is NetLogo (Wilensky, 1999a). NetLogo is both a modeling language and an integrated environment designed to make agent-based models easy to build. In fact, NetLogo is so easy to use that, rather than describe algorithms and models in pseudo-code,[7] throughout this textbook we will use NetLogo instead. Describing the preceding rules in the NetLogo language is straightforward. Here, for example, is one translation of rules 1–3 into NetLogo code:

```
If not carrying-food? [ look-for-food ]     ;; if not carrying food, look for it
if carrying-food? [ move-towards-nest ]     ;; if carrying food turn back towards the nest
wander                                      ;; turn a small random amount and move forward
```

This code snippet is not the complete implemented model but it does describe the core components that go into the Ants model. To complete the model we need to describe each of the subcomponents (such as "move-towards-nest," and "look-for-food," and "wander" and "look-for-food" will need to describe the ant's sniffing for pheromone). Each of which is a small amount of code. The result will be a fully implemented computational model. (See figure 1.2.)

Let us quickly try to "read" the code snippet and understand what it is doing. The "if" primitive takes as input a predicate (i.e., a statement that is either true or false) and takes

6. The process of comparing data from a computational model to real-world data is called *validation*. It will be discussed in depth in chapter 7.

7. Pseudo-code is an intermediate form between text and computer code that is often used to describe computational algorithms.

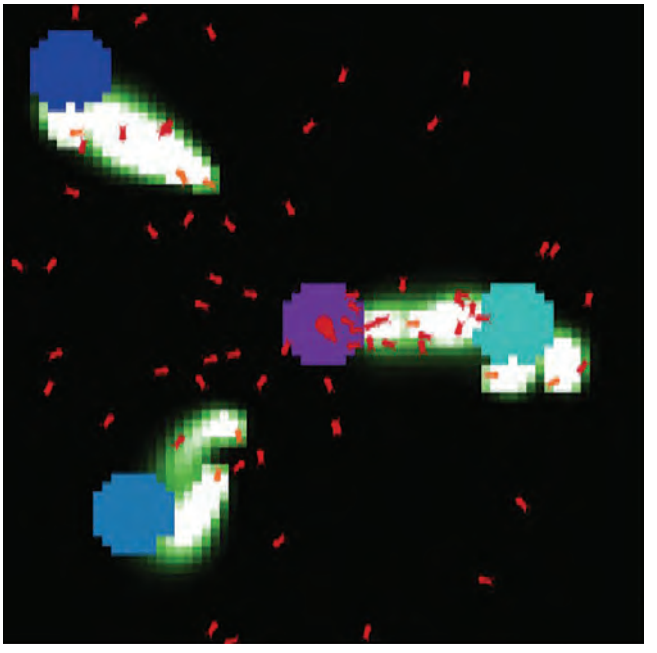

Figure 1.2
NetLogo "Ants" model of foraging behavior.[8]

one action if the predicate is true and another if it is false. The first "if" in the preceding code asks if the ant is carrying food. If she is not carrying food, then she takes the first action, that is, to look for food. She does this by checking to see if there is food where she is standing, and if there is not she looks around to see if there is a pheromone trail nearby that she can follow to find food (this is all described in the "look-for-food" procedure). If she is carrying food, then she takes the second action, which is to turn back toward the nest. To head back to the nest, she first determines if she is at the nest, if she is, then she drops her food, otherwise she turns so as to follow the trail back to the nest, dropping some pheromone along the way, since she just came from a food source. Regardless of whether she is looking for food or returning to her nest, the ant wanders a little bit along the direction she is heading. This is to simulate that ants, as with almost all animals, do not usually follow a straight line, but instead take small steps in other directions along the way.

By the end of chapter 3, you will be able to make substantive modifications to this model, and by chapter 4, you will be able to construct such a model on your own.

8. http://ccl.northwestern.edu/netlogo/models/Ants (*Wilensky, 1997*).

Box 1.2
Exploring the Ant Foraging Model

To explore this model in more depth, open the Ants model (Wilensky, 1997) (found in the Biology section of the NetLogo models library). Once you open this model, you will see a set of controls that you can manipulate to change the parameters of the model. Try varying some of the parameters (e.g., POPULATION, DIFFUSION-RATE, EVAPORATION-RATE), and explore the accompanying material that discusses this model and the real-world experiments it is based upon. As you manipulate the model, consider the following questions:

1. How does the evaporation rate affect the ability of the ants to form trails to the food? What happens if there is no evaporation?
2. How does the rate of diffusion affect the kind of trails the ants form?
3. How does the number of ants affect the colony's ability to consume the food?

Results and Observations from the Ant Model

When running the Ants model, the results are initially surprising for someone who has only seen the rules for the individual ants. The "aggregate" or "macro" behavior of the model shows apparently systematic food gathering behavior. It is as if the ant colony has a clear plan for how to gather the food. Yet, we have seen that the Ants model rules do not contain any systematic foraging plan. If we look closely at the model running, we observe that initially the ants wander around at random. Then some ants will wander into a nearby food source. Once they find that food source, they will start to bring food back to the nest, laying a pheromone trail beneath them. If just one lone ant finds the food source, the pheromone trail will not be strong enough for other ants to follow it; but as more and more ants find the food source, the trail will become stronger and stronger. Eventually, the actions of many ants will create a strong pheromone trail from the nest to the food source, and so any ant can easily find the trail to the food source.

The ants as a group appear to exploit food sources in an optimal manner. That is, they first gather food from the nearest food source, then the second nearest, and so on. This appears to be a conscious plan of the ant colony; but as we know from the ant rules, this is not the case. In fact, as you observe the model closely, you will note that sometimes ants operate almost at cross-purposes with other ants, creating additional pheromone trails to farther food sources and distracting some of the ants that are currently harvesting the closer food source. The ants do not have a centralized controller; instead, the nearest food sources are the most likely to be found first by the ants randomly wandering from the nest. The nearest food sources also require the least amount of pheromone since the pheromone only has to cover the shortest path from the nest to the food. Once a sufficient number of ants have found a particular food source, the pheromone trail to it stabilizes and thus

attracts more ants to it.[9] When the food source has been completely consumed, the ants no longer lay pheromone; hence, the trail dissipates, and the ants are released to search for other food sources.

This optimal exploitation of food sources could be placed within a larger context. In many ways, the colony of ants seems to balance exploration and exploitation (Dubins & Savage, 1976. In any situation in which an entity is operating in an unknown environment, the entity must spend some time exploring the environment to understand how its actions affect its rewards, and some time exploiting the environment, that is taking actions that it knows have produced the best rewards in the past. By allocating a large number of ants to exploit the current nearest food source while other ants continue to explore, the ant colony as a whole successfully balances exploration and exploitation.

However, these "trails" that the ants build to the food source, the "optimal" behavior that they exhibit, and the "balance" between exploration and exploitation are not coded into any one ant. There is nothing that tells the ant to build a trail; there is nothing that tells the ant to go to the nearest food source first; there is nothing that tells some ants to explore while others exploit. The "trails," "optimal," and "balance" behavior of the ants is not coded into any of the ants but is instead an *emergent* phenomenon of the model (Holland, 1998; Anderson, 1972; Wilensky & Resnick, 1999). Having run and used the model, it may seem obvious that these low-level rules can create these rich and optimal global patterns. But the history of science is full of wrong turns in which scientists believed that a complex phenomenon needed a complex organizational structure and a leader (Resnick, 1994; Wilensky & Reisman, 2006; Wilensky & Resnick, 1999). In contrast, through ABM we understand that this complexity can self-organize without a leader.

What Good Is an Ant Model?

The Ant model is our first example of an agent-based model. Now that we have gone through the process of understanding how a model of ant foraging works, what knowledge have we gained from that process and how can we profitably use the model? It may seem at first that the only thing the model does is provide a visualization of one particular textual model. We describe eight main uses for agent-based models: (1) description, (2) explanation, (3) experimentation, (4) providing sources of analogy, (5) communication/education, (6) providing focal objects or centerpieces for scientific dialogue, (7) as thought experiments, and (8) prediction.

A model is *descriptive* of a real-world system. Granted, it is a simplification of the real world and does not contain all of the details and inconsistencies that are present in the real

9. The increase in pheromone attracts an increasing number of ants to the vicinity of the pheromone, which in turn increases the amount of pheromone. This process exhibits positive feedback, which will be discussed further in chapter 7.

Box 1.3
Emergence and Emergent Phenomena

Emergence is a property classically exhibited by many agent-based models, and it occurs when an attribute that can be described at a system level is not specifically encoded at the individual level. This can lead to surprises when behaviors encoded at an individual level result in unexpected behavior of the system at a macro level. Indeed, when most people encounter an emergent phenomenon, they tend not to see the emergence, and they usually explain the phenomenon as the result of deterministic, centralized control. Complex systems are characterized by emergent phenomena—patterns that appear to be quite complex can often be generated by simple rules. When exploring agent-based models, you may be surprised at the simplicity of the agent rules; a key to building agent-based models is to harness emergence by finding the simple rules that can generate the phenomenon.

world. But all models are coarse-grained descriptions of reality; and, in fact, models that are not coarse-grained descriptions are useless as descriptions because they are indistinguishable from the real world and therefore do not assist in our understanding of complex systems[10] (Korzybski, 1990). If your model includes all aspects of the real phenomenon, it is more efficient to simply observe reality, since it saves you the time of building the model. The function of a model is to help us to understand and examine phenomena that exist in the real world in more tractable and efficient ways than by simply observing reality. Even if you have never observed a real ant colony, the Ants model helps—it can help you know what to look for and generate hypotheses that you can confirm or disconfirm by observation.

Models are *explanatory* in that they point out the essential mechanisms underlying a phenomenon. They can function as a proof that hypothesized mechanisms are sufficient to account for an observation. Models provide us with a proof of concept that something is possible. For example, once we built the Ants model and observed the results, we proved that an ant colony could exhibit characteristics such as "trails," "optimality," and "balance" without a centralized controller (Resnick & Wilensky, 1993; Resnick, 1994; Wilensky & Resnick, 1999). These characteristics are all emergent outcomes of low-level mechanisms. A key function of ABMs is to explicate the power of the emergence. In general, it is difficult for people to understand how such simple rules can lead to complex observed phenomena, and ABMs make this connection explicit. Even if this were not the way ants actually worked, the model illustrates that this is one mechanism that could be used. We can also compare and contrast alternative hypotheses. We could, for example, build the other food-gathering hypotheses discussed earlier as computational models and compare

10. The Argentine author Borges has a fanciful short story (1946) based on the premise of a map as big as the terrain it maps.

their results to the Ants model and to real-world observations to determine the most plausible explanation.

Models facilitate *experimentation*. Models can be run repeatedly to note variations in their dynamics and in their outputs. Some models have very little variation from run to run. Others exhibit *path dependency* (see chapters 7 and 8) and can therefore vary tremendously from run to run. Model parameters can be varied to see their effect on model behavior and outputs. For example, in the Ants model, we could change the evaporation rate of the pheromone and see what effect that has on the performance of the ants. Alternatively, we could model the return to nest behavior in more detail, by placing a sun in the environment and have the ants calculate their path back based on the sun. From there we could start to explore some of the modern research on ant behavior that says that ants know the length of their legs and use that to calculate how to return to the nest (Wittlinger et al., 2006). We could then reproduce results from real experiments using our modified ants. Thus agent-based models enable us to easily examine different mechanisms and attributes of a system and see what effect those modifications have on the overall behavior of the system. By varying these different attributes or parameters and observing their effect on the behavior of the system, we can classify model behavior into output *regimes*, or characteristic output behaviors of the model.

Models provide us with *analogies*. Since models are simplifications of reality, they enable us to find similarities with other such simplifications even if the modeled phenomena are apparently very different. In this way, we can apply reasoning gained in one domain of knowledge to other knowledge domains. For instance, in 1996, Schoonderwoerd, extending Dorigo's work on ant colonies as optimization devices, found that the problem of ants efficiently finding food sources is similar to the problem of packets on networks efficiently seeking out their destinations (Schoonderwoerd et al., 1996). Once this analogy was created it was possible to study the ant colonies in more depth and use the results gathered from those observations to create better algorithms for controlling packet behavior. The application of algorithms based on ant colonies has become a separate field within computer science known as ant colony optimization (Dorigo & Stützle, 2004).

Agent-based models can be used as a vehicle for *communication* and *education*. We can show the model to people who have never seen an ant colony before and they can explore how ant colonies behave. Models provide us with an educational tool that encapsulates knowledge that may not be readily available from real world observation. Moreover, an agent-based model allows us to expand beyond static knowledge, enabling learners to conduct experiments much as scientists do. If someone has a hypothesis as to how a particular mechanism works, they can implement that mechanism and see whether it can account for observed behavior. Wilensky and colleagues (Blikstein & Wilensky, 2009; Levy & Wilensky, 2009; Sengupta & Wilensky, 2009; Wilensky, 1999b; Wilensky & Reisman, 2006) have demonstrated the power of using ABMs for these purposes, enabling students to more deeply understand and engage with science.

The Ants model provides the scientific community with a *focal object* (in Seymour Papert's words: "an object to think with" [1980]) that enables us to discuss the behavior of ants. It forces us to specify the important mechanisms that generate ant behavior. We can discuss which mechanisms are most important for generating the behavior and how removing or adding mechanisms would change the colony-level behavior. Then we can test alternative hypotheses by specifying the mechanisms in code and running the revised model. Since we are using a computational model, we eliminate the ambiguities that usually exist in textual models where some people may read or interpret descriptions differently from others. The agent-based model of an ant colony provides us with a "glass box" (as opposed to a "black box"), through which we can examine and observe the operations of the ants and discuss and test whether or not hypothesized mechanisms are valid. More generally, the glass box of agent-based models provides us with an unambiguous representation of the problem that we are examining, and thus is useful not only within the scientific community but also in other realms such as policy analysis.

Models can sometimes present new phenomena that are not necessarily about some real-world phenomenon, but are *thought experiments* on possible computations. Some might not use the word "model" for this kind of computational artifact, but many scientists do refer to them as models. Some classic examples of these kinds of models are cellular automata, fractals, and particle swarms. We will be examining such models as well in this textbook, and indeed we start the next chapter with a classic cellular automaton.

A common conception of computer modeling is that its major purpose is *prediction*. It is true that we often use modeling to think about possible future scenarios, and to the extent that any modeling tool can be used for discussing the future, agent-based modeling can be used in this way. However, like any modeling tool, an agent-based model's predictions depend on the accuracy of its input data. This is especially true when the events we are interested in are the results of complex systems, for which small changes in inputs can often lead to very different results. It is always difficult to assess the accuracy of a model's predictions of events when those events have not yet occurred. Many times, while modelers may claim to want to use a model to predict the outcome of a system, such as an ant colony, they actually use the model to *describe* past patterns of behavior, such as what food source was first eaten, and to *explain* future patterns that might arise, such as how ants might move around on the landscape. Sometimes modelers start off motivated by generating predictions but end up finding greater power for explanation and description. Agent-based models, in particular, are distinctive from other modeling approaches in that they were designed in order to understand and explain complex phenomena that otherwise could not be explained though traditional approaches.

We have been discussing the Ants model and its results for some time now. This model is one example of an agent-based model. But there is a wide diversity of models that are called agent-based models—from biological models to models of political systems to models of particle interactions. So what are the key components of agent-based models

that make them agent-based models? Now that we have described the process of designing one particular model, let us take a step back and describe more formally what agent-based modeling is and how it can be used.

What Is Agent-Based Modeling?

The profitable uses of the Ants model that we described earlier are not particular to that one model. These are generally applicable affordances of the methodology used to develop the model, which is agent-based modeling. The core idea of *Agent-Based Modeling* is that many (if not most) phenomena in the world can be effectively modeled with agents, an environment, and a description of agent-agent and agent-environment interactions. An *agent* is an autonomous individual or object with particular properties, actions, and possibly goals. The environment is the landscape on which agents interact and can be geometric, network-based, or drawn from real data. The interactions that occur between these agents or with the environment can be quite complex. Agents can interact with other agents or with the environment, and not only can the agent's interaction behaviors change in time, but so can the strategies used to decide what action to employ at a particular time. These interactions are constituted by the exchange of information. As a result of these interactions, agents can update their internal state or take additional actions. The goal of this textbook is to explore in detail all of the different aspects and uses of agents and their interactions.

Agent-Based Models vs. Other Modeling Forms

What makes agent-based models distinct from other models? The most common form of scientific models is the equation form. Parunak, Wilensky, and colleagues (Parunak et al., 1998; Wilensky, 1999b; Wilensky & Reisman, 2006) discuss the many differences between ABM and equation-based modeling (EBM). One distinction is that because ABM models individuals it can model a heterogeneous population, whereas equational models typically must make assumptions of homogeneity. In many models, most notably in social science models, heterogeneity plays a key role. Furthermore, when you model individuals, the interactions and results are typically discrete and not continuous. Continuous models do not always map well onto real-world situations. For instance, equation-based models of population dynamics treat populations as if they are continuous quantities when in fact they are populations of discrete individuals. When simulating population dynamics it is very important to know if you have a sustainable population. After all, a wolf population cannot continue if there are fewer than two wolves left; in reality, a millionth of a wolf cannot exist and certainly cannot reproduce, but it can result in increased wolf population in EBMs. The mismatch between the continuous nature of EBMs and the discrete nature of real populations causes this "nano-wolf" problem (Wilson, 1998). As a result, for EBMs

to work correctly, they must make the assumption that the population size is large and that spatial effects are unimportant (Parunak et al., 1998; Wilensky & Reisman, 2006; Wilkerson-Jerde & Wilensky, 2010, in press).

Another advantage of ABM over EBM is that it does not require knowledge of the aggregate phenomena: One does not need to know what global pattern results from the individual behavior. When modeling an outcome variable with EBM, you need to have a good understanding of the aggregate behavior and then test out your hypothesis against the aggregate output. For example, in the wolf-sheep (predator-prey) example, to build the EBM, you need to have an understanding of the relationship between (aggregate) wolf populations and sheep populations. To encode this aggregate knowledge such as in the classic Lotka-Volterra equations (Lotka, 1925; Volterra, 1926), you must have knowledge of differential equations.[11] In contrast, ABM enables you to write simple rules for simple entities, requiring knowledge only of commonsense behaviors of individual wolves and sheep and yet still observe the aggregate result by running the model. Thus, even if you have no hypothesis as to how the aggregate variables will interact, you can still build a model and generate results.

Because agent-based models describe individuals, not aggregates, the relationship between agent-based modeling and the real world is more closely matched. It is therefore much easier to explain what a model is doing to someone who does not have training in the particular modeling paradigm. This is beneficial because it means that no special training is required to understand an agent-based model. It can be understood by all of the stakeholders in a modeling process. Moreover, with some ABM languages like NetLogo, the syntax is so readable that stakeholders without knowledge of how to build a model can often read the model code and understand what is going on. This helps improve the verifiability of the model.[12] This "glass box" approach to modeling (Tisue & Wilensky, 2004) enables all interested parties to talk about the model all the way down to its most basic components.

Finally, the results generated by ABMs are more detailed than those generated by EBMs. ABMs can provide both individual and aggregate level detail at the same time. Since ABMs operate by modeling each individual and their decisions, it is possible to examine the history and life of any one individual in the model, or aggregate individuals and observe the overall results. This "bottom-up" approach of ABMs is often in contrast with the "top-down" approach of many EBMs, which tell you only how the aggregate

11. Differential equations are often represented with another modeling approach, systems dynamics modeling, which provides discrete approximations to the equations. We discuss system dynamics modeling in more depth in chapter 8.

12. A model is considered *verified* if the implemented model matches the conceptual model. Of course, since conceptual models and implemented models are distinct entities, it is impossible to say a model is completely verified, but it is possible to say that model is verified to a certain extent.

system is behaving and do not tell you anything about individuals. Many EBMs assume that one aspect of the model directly influences, or causes, another aspect of the model, while ABMs allow indirect causation via emergence to have a larger effect on the model outcomes.

Randomness vs. Determinism

One important feature of agent-based modeling, and of computational modeling in general, is that it is easy to incorporate randomness into your models.[13] Many equation-based models and other modeling forms require that each decision in the model be made deterministically. In agent-based models this is not the case; instead, the decisions can be made based on a probability. For instance, in the Ants model, as the ants move around the landscape, their decisions are not completely determined; instead, at each time step they change their heading a small amount based on a random number. As a result, each ant follows a unique, irregular path. In reality, ants might be affected by small changes in elevation, the presence or absence of twigs and stones, and even the light of the sun. To build a complete model of all of these factors might be very tedious and would probably be very specific to a particular environment. Moreover, since our real goal in building this model is to understand how ants gather food and not how they move about the landscape, there is no guarantee that a more deterministic model will provide us with a better answer to this question. Thus, using the random number serves as an approximation that may turn out to be just as correct in answering our driving question.

The "random" Ants model is easier to describe than a deterministic one. If we are explaining the Ants model to a person who has never seen it before, we can say that at each time step the ants change their heading by a small random amount. We do not have to describe how the model takes into account all of the environmental factors that could be involved. This simplification also speeds up model development, since we do not need to spend time formalizing all of these details. If at a future time we decide that the ants do need to make more deterministic decisions about their environment, we can incorporate that knowledge at that time. Thus, though randomness in a model acts as an approximation to real world concepts, the model can later be made less approximate by the incorporation of additional knowledge.

Finally, there are often times when we simply do not know enough about how a complex system works in order to build a completely deterministic model. In many of these cases the only type of model that we can build is a model with some random elements. Agent-based modeling and other modeling forms that allow you to incorporate random features are essential to studying these kinds of systems.

13. Since computers are deterministic machines the randomness that they possess is not true randomness but rather "pseudo-randomness." This will be discussed further in chapter 5.

When Is ABM Most Beneficial?

Agent-based modeling has some benefits over other modeling techniques, but, as with any tool, there are contexts in which it is more useful than others. ABM can be used to model just about any natural phenomenon (e.g., you could describe any phenomenon by describing the interaction of its subatomic particles). However, there are some contexts for which the cost of building an ABM exceeds the benefits, and there are other times when the benefits are extraordinary given the costs. It is sometimes difficult to discern the difference between these two scenarios. However, there are a few general guidelines that can help to identify situations where ABM will be particularly valuable. These are meant as guidelines and not particular prescriptions or "rules" about when to use ABM. Most often, you will have to judge based on the particular situation.

Some problems with large number of homogenous agents are often better modeled (i.e., they will provide more accurate solutions to aggregate problems faster) using an aggregate solution like mean field theory or system dynamics modeling (Opper & Saad, 2001; Forrester, 1968). For instance, if you are concerned about the temperature in a room, then tracking every individual molecule and its history is not necessary. On the other hand, if a problem has only a handful of interacting agents, then you usually do not need to bring to bear the full power of ABM and instead can write detailed equations describing the interaction—two billiard balls colliding, for example, does not require ABM. As a rule of thumb, agent-based models are most useful when there are a medium number (tens to millions) of interacting agents (Casti, 1995).

Agent-based models are more useful when the agents are not homogenous. For instance, modeling all the trades and events on a stock market floor requires a more rich and detailed examination of individual-level behavior. Different stock trading agents have different risk thresholds and hence will not make the same decision given the same environmental state. Even in the Ants model, while the ants all had the same rules of behavior, they were not homogenous in location, heading, food-carrying state, and so on. ABM is very useful when agents are heterogeneous and the heterogeneity of the agents affects the overall performance of the system. Since ABM enables each individual to be tracked and described at the individual level, it is much more powerful than techniques such as systems dynamics modeling (Forrester, 1968; Sterman, 2000; *Richmond & Peterson, 1990*). System dynamics modeling requires the creation of a separate "stock" for each group of agents with different properties, and, when the space of properties is large, this becomes difficult to build, track, and integrate. ABM, on the other hand, requires you only to specify how agents' properties are defined and not to keep track of all possible agent types, which provides a more concise description of a complex system. Thus, using ABM is especially beneficial when the agents are heterogeneous.

Having heterogeneous agents also allows the interactions between agents to be quite complex. Since we can specify an almost infinite number of different agent types, we can specify just a few simple rules to describe how those agents interact with each other to

create a very rich tapestry of interactions. Moreover, since ABM allows individual agents to keep a history of interactions, they can change their behaviors, and even their strategies, based on past events.[14] For example, in an ABM of the evolution of cooperation, it is possible that agents can learn and hence modify their behavior as a result of continual interaction with a particular group of agents. They may learn to distrust that group, or alternatively they may learn to act more favorably toward that group. Thus ABM is very useful when modeling complex interactions of adaptive agents.

In the same way that ABM is useful when the interaction between agents is complex, it is also useful when the agents' interaction with the environment is complex. The environment in an ABM is often itself composed of stationary agents, and thus modeling agent-environment interactions has all of the power of modeling any agent-to-agent interaction. For example, in an ABM of fish ecology, a fisherman can recognize a particular location as a place that he has fished before and decide not to fish there again. This agent-environment interaction enables geographic and location-dependent information to be included in the model, and thus we get richer data than a geographic-independent model. In the fish model, it may be known that the average fish population is steady over time for a large area, but that could mean that the average fish population is steady in all of the subareas, or that different subareas trade off with each other, resulting in a larger fish population in some places and a smaller one in others. Thus, the rich description of environment and geography entailed by ABM allows for the generation of more detailed information. This enables ABM to generate spatial patterns of results as opposed to spatially homogenous aggregate results.

Another way that ABM provides more detailed information than equation-based or many other modeling approaches is through its rich conception of time. In ABM, one models agents and their interactions with each other. These interactions occur temporally; that is, some interactions occur before or after others. ABM thus enables you to move beyond a static snapshot of the system and toward a dynamic understanding of the system's behavior. In this way, ABM provides a rich and detailed account of the process of a system's unfolding in time, and not just the final state of the system. For example, in a stock market model you can actually observe individuals buying and selling stocks over time instead of modeling only the change in the stock's price. By enabling a detailed conception of time, ABM vastly expands on the detail of the resultant model.

Trade-offs of ABM

Agent-based modeling provides some benefits over other methods of modeling, but, in any particular situation, choosing a modeling methodology is a case of choosing the

14. Strategies are distinct from behavior because they express how to behave in a particular set of circumstances. Thus, a change in strategy often results in a change of behavior, but a change in behavior is not necessarily the result of a change in strategy.

appropriate tool at the appropriate time, and sometimes agent-based modeling is not the right tool for the job.

For example, ABM can be computationally intensive. Simulating thousands or millions of individuals can require great computing power. Equation-based models, by contrast, are often very simple to run and essentially just require repetitive mathematical calculations. This is true only for simple equation-based models; numerically solving complicated equation-based models may take as much computational time as agent-based models. The computational expense of running an ABM is a price one pays for having the benefits of rich individual-level data. The additional computational power needed for running ABMs is the same power that allows the tracking and development of rich histories of individuals. For example, in the Ants model we could write a simple equation that describes the rate at which food is gathered based on its distance from the nest. However, through ABM, we are able to observe the behavior of the individual ants and understand how they form trails to the food source. This same trade-off is faced by any modeling or simulation environment: more detailed results and more detailed models inevitably require additional computational resources. However, if an ABM is built well, it is possible to reduce the amount of computational power required by "black-boxing" parts of the model. This can be done by strategically using equations to control computationally intensive parts of the model. When the results warrant a higher fidelity description, one can then open up the black box.

The more detail there is in a model, the more modeling decisions have to be made by the modeler. In the equation-based modeling (EBM) literature, variables whose values are determined by the modeler are referred to as "free parameters." For example, in the Ants model, the rate at which the pheromone evaporates can be modified. This creates an additional free parameter in the system. In contrast to EBM, in ABM, there are typically more free parameters for control of the additional levels of detail. Calibrating these free parameters and making sure that they are set correctly can be a time-intensive process. Some critics of agent-based models have argued that, since ABM uses so many free parameters, it can be used to generate any result desired. We disagree. In our view, EBMs and other aggregate models and simulations simply "hide" these free parameters by making implicit assumptions about the way the system works. Often, the mathematical equations hide the implicit free parameters, since it is not possible to incorporate them within the equation. In ABM, free parameters such as the evaporation rate are exposed. One can initially set up a model to use an idealized value for evaporation rates, but the rates can also be tuned to match true biological evaporation rates. ABM generally uses more free parameters than other types of modeling because these free parameters control assumptions of the model, and ABM explicitly exposes those assumptions at more levels of action.

In order to set or modify these low-level free parameters/modeling decisions, ABM does require that the modeler have some knowledge of how individual elements of the system

operate. Without knowledge of or an educated opinion about how individual agents in a complex system operate, there is no material from which to build an agent-based model. Gaining this knowledge of individual agents requires additional levels of understanding of the system—effort that may not be necessary for building an EBM or aggregate model. For example, in the Ants model, some description of an individual ant's behavior is a requirement. It would make little sense to build an agent-based model that describes only the aggregate amount of food consumption. But the work to gain this agent-level knowledge pays off—it is these same low-level assumptions that give us a richer understanding of the phenomena being observed. ABM does require us to gain an understanding of the microbehavior of a system, but without modeling the microbehaviors, ABM would not provide as elaborate a model as it does. While ABM does require us to know or guess about the individual-level mechanisms operating, it does not require knowledge of the aggregate level mechanism. It is often useful to have a description of aggregate level system behavior so that the results of an ABM can be validated. But since ABM focuses on the individual, there is no need to have a causal description at the aggregate level. For many phenomena, especially social phenomena, it is often easier to think about the individual level than the aggregate level. For example, it is easier to think of someone spreading a rumor than to think about the rate of spread of the rumor through the population. To get started building an ABM, your knowledge of the low-level elements need not be in-depth. In fact, it can take the form of an educated guess. For instance, even if we have not studied individual ants, but only ant colonies, we can make a hypothesis as to how individual ants communicate and generate a model that represents that hypothesis. If the model produces valid results, then we can say that it represents one potential way of how the ant colony might operate. This "proof of concept" or "existence proof" use of ABM is powerful.

What Is Needed to Understand ABM?

Now that we have introduced the concept of ABM, we can outline the rest of this book, which will provide you with the tools and capabilities to understand, build, and analyze your own ABMs. In chapter 2, we will construct some very simple ABMs and start to build the knowledge necessary to construct more detailed ABMs. In chapter 3, we present three already constructed ABMs and learn how to run them and to modify their code. In chapter 4, we build our first full ABM and examine how to expand upon it to create increasingly detailed ABMs. In chapter 5, we introduce and describe the major components of an ABM and also introduce network-theoretic, environmental, and user-related concepts useful for building ABMs. In chapter 6, we discuss how to present ABM results including setting up experiments and analyzing results. We cover output-related issues such as graphing, statistical analysis, network analysis and GIS analysis as well as visual displays. In chapter 7, we examine how to compare model results to the real world and use that comparison to refine and extend a model. We show how to determine if a model is verified

(the code represents the conceptual model) and validated (the model has a correspondence to the real world). We also cover issues to consider in replicating an ABM. Since validation and replication usually require statistical comparison, there is a short introduction to the necessary statistics.

In chapter 8 we tie many of these threads together to discuss how ABM is applied in real-world settings and examine advanced uses of ABM. We highlight some of the principal examples from domains such as ecology, economics, land-use planning, computer science, and political science. We discuss what ABM methodology has contributed to scientific knowledge and what it will be used for in the future. We discuss how to incorporate these richer data sources into your ABM. These sources include GIS, Social Network Analysis, and sensor data (visual and nonvisual). We address how to export data from ABMs to advanced mathematical analysis packages. We also discuss how to make ABMs more powerful, by incorporating techniques such as machine learning, system dynamics modeling, and participatory simulation. We conclude with a discussion of future research trends and challenges within ABM and upcoming areas of applications of ABM to new knowledge domains.

In the appendix, we examine the origins and history of ABM, with an emphasis on its computational roots. This is provided to set a historical context for the rest of the book enabling us to understand how a variety of fields came together to create what we now call ABM. Some readers may wish to read the appendix at this time, though it can be completely skipped for the reader focused on model building. But before we embark on our ABM journey, allow us to take one last look at the Ants model.

Conclusion

The Ants model is interesting for biologists, and we have even discussed how it can be used to reason analogically about other systems, like computer networks and path planning. But what if we wanted to transform the Ants model to be more like some other system? There are many similarities between ant colonies and human organizational systems. They both exhibit problem-solving behavior. They both are results of organized structures that have evolved over the millennia. For example, what if we tried to reconceptualize the ants in the system as humans? Then we can visualize the ant colony as the central business district of a town. With this slight shift in perspective, we can start to see how the model could resemble a human city, with individuals that go off to work every day and return in the evening.

The last description suggests a major difference between the ant and the human systems. Humans tend to leave for work around the same time in the morning and return home around the same time in the evening. So we need to modify our model slightly: Instead of having the ants (now humans) leave randomly from the nest, we have them leave at random intervals around a start time, go off to find some food, and stay near

the food until a certain amount of time passes, then return to their homes. And humans do not live in a colony; they live in different locations around a city, so we need to give each human a different home that they start with, and then allow them to walk (with some randomness) to their work. But humans do not walk randomly (well, not much of the time); they instead take preplanned routes on roads. So now let us put down a road network for them to drive on their way to work. But if they are driving to work, then they will be limited by the speed of the traffic. So now we need to implement a vehicle simulation on the roads. And so it goes. … Slowly, our model of one specific ant blossoms into a model of many ants collecting food, which then metamorphoses into a model of urban commuting patterns. A powerful aspect of ABM is that it enables us to find universal patterns that characterize apparently quite different phenomena, to generate these patterns with simple rules, and to explore the effects of simple modifications to those rules.

What models will you build? What are the simple rules that describe the agents in your model? What are your agents? Are they humans, ants, cars, computers, deer, viruses, cells, coffee trees, hurricanes, air particles, electrons, snowflakes, sand grains, students, teachers, videogames, marketing strategies, innovations, or any of a vast number of objects, events, or things? Whatever it is you want to model, ABM provides you with tools and capabilities that enable you to simulate and analyze it as a complex system. As you progress through this book, you will be introduced to the tools and develop the skills that you need to explore the world around you in an agent-based way.

At this point, it is recommended that you work through the three NetLogo tutorials found in the NetLogo user manual that is available from the help menu of the NetLogo application. It will be necessary to work though the tutorials to do many of the explorations at the end of this chapter and to follow chapter 2.

Explorations

Beginner NetLogo Explorations

1. Complete the tutorials that are available in the NetLogo User Manual.
2. Look over the models in the Sample Models section of the NetLogo models library. The models are grouped by subject area. Pick out a model you find interesting and try running it in different ways. What set of parameters gives you the most interesting behavior? Is there a way to change a parameter in a small way and get a very different behavior? Explicitly describe the rules the agents are following.
3. Describe a phenomenon that you think it would be interesting to model using ABM. What are the agents in this model? What properties do they have? What kind of actions can the agents take? What kind of environment do the agents exist in? What is the order of events that occurs at each time step of the model? What types of output will this model generate? What do you expect to observe as a result of running this model?

Ants and Other Model Explorations

4. Examine the code for the NetLogo Ants model, described in the chapter. The wiggle procedure right now has the ant turn a random amount to the left and then back to the right a random amount. This approximates a random walk that is centered on moving straight ahead. If you changed the procedure so the walk was biased to the left or to the right, how would that change the results? What if the limits of how much the ant turned were changed? Make these modifications and observe the results.

5. *Termites model* Run the Termites model (found in the Biology section of the NetLogo models library). In this model there are only two objects: termites and wood chips. What are the termites doing in this model? Without looking at the code or the info window, can you describe the rules governing the termites' behavior? Hint: It might help to reduce the number of termites and wood chips and to slow down the speed slider.

6. *Daisyworld model* Some ABMs are used not as models of real-world events, but rather as thought experiments. Run the Daisyworld model (found in the Biology section of the NetLogo models library). This model defines a world in which the whole surface is covered by daisies, and it examines how different factors affect the global temperature of the world. Adjust the parameters of the model and observe how the model reacts. The standard parameters result in a temperature slightly below 50. Find a set of parameters that move the temperature closer to 12. This model rarely results in a constant temperature; it usually oscillates. Describe this oscillation.

Concept Explorations

7. *Modeling at different levels* Agent-based models can be written at different levels. For example, one model may have agents that are populations of wolves and sheep, whereas another model may have agents that are individual wolves and sheep. Write a description of how packs of wolves interact with flocks of sheep at the group level. Now write a description of how individual sheep interact with individual wolves. How are your descriptions different? What phenomena are you describing? At what times would the group level description be helpful? At what times would the individual level description be helpful?

8. *Emergence and ABM* Agent-based models often exhibit emergent properties. One characteristic of an emergent phenomenon is that the system exhibits a property that is not defined at the individual level. For instance, examine the Traffic Basic model (found in the Social Sciences section of the models library). Run the model several times and observe the results. What causes the traffic jams in the model? Does there appear to be any external event that causes them? Inspect the cars. Is there any property of these cars that describes a traffic jam? If one car moves slowly, is that enough to cause traffic jams?

9. *ABM for education and understanding* ABM provides us with a new way of understanding the world around us. ABM has many uses in research. But ABM also has great potential as a tool for education. For instance, molecules in a free gas can be thought of

as agents moving around and colliding with each other. Examine the GasLab Free Gas model (found in the Chemistry and Physics section of the models library). Do you think this model is easier to understand than a traditional equation-based approach to understanding Free Gas phenomena? What affordances does the ABM approach give us that traditional approaches lack? Are there ways that ABM can be more confusing than traditional approaches? If so how?

NetLogo Explorations

10. Create at least two different ways of distributing turtles randomly across the screen. In one method, use only turtle motion commands such as *forward*, *left*, and *right*. In another method, use *set* or *setxy*. Create buttons to launch these procedures. Compare and contrast your different methods. Is one of these more efficient? Is one of them more realistic? In what situations would each of them have advantages over the other?
11. Write a procedure to get a color to spread from patch to patch. (There are many ways to do this. Pick one you like.) Create a button to launch this procedure.
12. Write a procedure that makes the turtles chase after the mouse cursor. Create a button to launch this procedure.
13. Select a new shape for the turtles from the shapes editor, and then create a "cloud" of turtles (a bunch of turtles in the same local area) using your new shape. Create some green patches. Make the turtles follow the mouse cursor around the screen but avoid the green patches. Make the green color spread from green patches to other patches nearby. Create buttons to launch these procedures.
14. Create a "cloud" of turtles, half of them one color and half of them another color. Based on a probability have one color of turtles move up and the other color turtles move down. Label the turtles with their WHO number. Create buttons to launch these procedures. Create a monitor that keeps track of how many times one of the turtles has moved.
15. (a) Create a new model with a SETUP procedure that creates turtles. (b) Create a slider that controls the number of turtles created. (c) Write a GO procedure that makes the turtles wander around the screen randomly. (d) Change the GO procedure to make the turtles afraid of each other. (e) Make the turtles die when they reach the edge of the screen. (f) Create a plot that displays the number of turtles.
16. Write two procedures. In the SETUP procedure, turn the left side of the screen red, and the right side of the screen green and create two turtles. Give one turtle a shape from the shapes editor and make a new shape for the other turtle. In the GO procedure, make the turtles move randomly about the screen. When a turtle is in an area of one color, create a circle of patches of the other color centered on the turtle.
17. Both turtles and patches can create visual images in the NetLogo view. Create a turtle and have it draw a circle (using the PEN-DOWN command). Create the outline of a circle with patches without using a turtle pen. Write a procedure that asks a turtle to draw a square given a starting location and side length. Write a similar procedure using patches.

Compare and contrast the code for these two procedures. Which set of code is more compact? Are there advantages or disadvantages to using patches or turtles to accomplish this task?

18. Open the *random walk example* (found in the Code Examples folder in the models library). Inspect the code. What do you predict the turtle's path will look like? Run the model. Does the path look like you expected it would? Modify the model code so that the turtle's path is still random but is less "jagged," i.e., is smoother and straighter.

19. Most of this textbook addresses the use of ABM to model and scientifically explain phenomena. However, you can also use ABM to create powerful visualizations. As we have mentioned, ABM has even been used to create Academy Award–winning special effects. Examine the Particle Systems Basic model (found in the Computer Science section of the models library). This model creates interesting visual images from the manipulation of simple agents. Explore this model, and understand how the agents behave and what properties they have. Describe a non-agent-based model that would create similar results as the base model with the initial parameters. Now examine the NetLogo model again. Change the initial number of particles, the step size, and the gravity. Can you describe both an agent-based model and a non-agent-based model that creates these results? Which of these two models is easier to describe? Why?

20. *Computer modeling and chaos theory* Chaos theory was developed from traditional equation-based modeling, but one of its inspirations came from computer modeling. Edward Lorenz discovered that mathematical systems could produce very different results depending on the initial conditions that the systems have. He realized this because he tried to restart a computer model of the weather system halfway through a run with a new set of parameters that lacked a small amount of precision from his previous set of parameters. The resulting model behaved very different from his original model. Agent-based models can exhibit this same "sensitivity to initial conditions." For instance, examine the Sunflower model (found in the Biology section of the models library). This is an agent-based model of how rows of sunflower seeds are added to a sunflower. Run the model with the default settings. Now change one of the parameters. Run the model again. Repeat this process. As you keep manipulating the parameters, do you eventually get to the point where you can predict the behavior of the model with new parameters? Explain your answer. Why is this model predictable or unpredictable?