9

Networks, Robots, and Artificial Life

9.1 Robots and the Genetic Algorithm

9.1.1 The robot as an artificial lifeform

In previous chapters we have seen that connectionist networks are adept at recognizing patterns and satisfying soft constraints. The pattern-recognition capability is useful for a variety of tasks, including visual perception, categorization, language, and even logical reasoning. The constraint-satisfaction capability can serve an equally diverse range of functions, such as controlling motor behavior, making decisions, and solving such classic problems as finding optimal routes for a traveling salesperson. A single network can combine both capabilities. For example, sensory information presented on an input layer can be interpreted on hidden layers as indicating the location of an object in a room. This information can then be used to generate appropriate motor commands on an output layer. A network like this knows how to locate and move to an object in a room – a simple but essential sensorimotor achievement. If yoked to a mechanical body and provided with a learning procedure, this sensorimotor network yields a very interesting device: a robot that can use experience to improve its own functioning. We have already encountered some elements of such a device in section 8.3.1, where the robot controllers designed by Beer (1995) were our first encounter with a newly emerging research area known as *artificial life* or *A-Life*. In the current chapter we will sample other exemplars of this line of research and consider benefits, limitations, and implications.

For connectionist modelers, embodying networks in robots can be envisioned as bringing some appealing benefits:

- If learning can be made to rely on consequences produced in the environment by the robot's actions, these embodied networks will learn much more naturally than the usual stand-alone networks provided with predetermined input–output pairings by a teacher.
- Placing networks in robots can be viewed as distributing the tasks of cognition beyond the internal cognitive systems (the networks) by coupling them to an environment. Sharing the cognitive burden in this way ought to reduce the load on the networks themselves (Clark, 1997a).

- Confronting the practical problems involved in making a robot perceive and act in an environment reminds us that these sensorimotor abilities are foundational to other cognitive performance. In real organisms, perception and action are major foci of early development and become effective, though still primitive, relatively quickly. In both phylogeny and ontogeny, systems seem to redeploy already-existing systems rather than building completely new ones, so it seems plausible that basic perceptual and motor systems provide computational frameworks which can be re-utilized in the evolution and development of higher cognitive capacities. (This essentially Piagetian point is modified, but not necessarily abandoned, by more recent investigators who would add certain conceptual, mnemonic, and other abilities to the inventory of foundational systems.)

This attractive picture has not yet been realized in its entirety. First, as always, advantages must be weighed against disadvantages. Building robots and training networks in them is expensive, in terms of both hardware and training time. Moreover, the fledgling attempts of a network to control the movements of a robot may produce serious damage to the physical robot. Some researchers sidestep these disadvantages, at the cost of weakening the advantages as well, by creating computer models in which simulated robots receive input and feedback from a simulated environment. Beer (1995) went even further by using the simulated robot body itself as the only environment in which the controller network functioned. (Recall that he used the simulated body's leg angle as the only source of sensory input to the network.) A second variation on the above picture pursued by many robot researchers, including Beer, is using simulated evolution as a method of developing networks in addition to (or in place of) learning.

One obvious advantage of the simulated evolution strategy is that it overcomes an unrealistic feature of most connectionist simulations: the networks start with random weights and must learn everything from scratch. Evolution can produce networks whose weights are fairly well adapted to their tasks prior to any experience. A second advantage is that the network architecture itself (not just the weights) can be allowed to evolve. Simulated evolution may even produce useful network configurations that would not be discovered by human designers (Harvey, Husbands, and Cliff, 1993).

9.1.2 The genetic algorithm for simulated evolution

Studies of simulated evolution generally rely on some version of the *genetic algorithm*, which was developed by John Holland (1975/1992) to explore the nature of adaptive systems (also see the textbook by Goldberg, 1989). Holland sought to simulate three processes that are critical to biological evolution: an inheritance mechanism that can produce offspring that resemble their parents, a procedure for introducing variability into the reproductive process, and differential reproduction. In the standard picture of biological evolution, the inheritance mechanism involves chromosomes (composed of genes), variability is achieved when genes recombine (an advantage of sexual reproduction) or mutate, and differential reproduction is caused by natural selection. (Alternatives to this standard picture have been proposed; for example, Gould and Lewontin, 1979, claim that differential reproduction sometimes is due to developmental constraints rather than external selection forces operating on the organism.)

In the genetic algorithm, strings of symbols play the role of chromosomes, operations such as recombination and mutation of these symbols are employed to introduce variation when the strings reproduce, and the fitness function governs selective reproduction by determining which strings are successful enough to be allowed to reproduce. The genetic algorithm applies recursively to produce a succession of generations. In each generation the most successful strings are selected to be parents, a new generation of strings is created by copying them (recombining or mutating the copies to introduce new variability), the offspring in turn undergo appraisal of their fitness, and those selected become parents of yet another generation. For example, in simulated evolution of an immune system (Forrest, Javornik, Smith, and Perelson, 1993), the evolving strings encode antibodies, and the fitness function evaluates how well each such string matches a specific antigen (represented by a string that does not evolve). In the case of connectionist networks (e.g., Belew, McInerney, and Schraudolph, 1991), a simple choice is to evolve strings of connection weights, but more interesting simulations are discussed below.

The new research area of artificial life is not limited to explorations of real and simulated robots and the evolution of networks to control them. Its general goal is to understand biological systems and processes. Its method is simulation, usually by means of computer programs. It can be carried out at a variety of levels (from individual cells or neural circuits to organisms to populations) and timescales (from that of metabolic processes to ontogenesis to phylogenesis). Robots are artificial organisms that operate at the timescale of individual actions or action sequences; networks are artificial nervous systems within these organisms and operate at the timescale of propagation of activation across connections or layers of connections. Artificial life researchers have investigated these plus much more. Before presenting a few specific studies of network controllers for robots, we will take a brief look at other research strategies in artificial life and how they have been applied in exploring very simple abstract organisms.

9.2 Cellular Automata and the Synthetic Strategy

Artificial life is related to biology somewhat as artificial intelligence (AI) is related to psychology. Psychology focuses on cognitive processes and behavior exhibited by actual organisms, whereas AI separates cognitive processes from their realization in living organisms. AI researchers have done this by constructing computer systems that function intelligently. Likewise, biology focuses on carbon-based life on earth, whereas artificial life separates the processes of life from their carbon-based realization. Like AI, artificial life relies on computers, but this time to simulate living systems and their evolution. Since behavior and cognitive processes are among the activities of living systems, the boundary between artificial life and AI is not rigid.

9.2.1 Langton's vision: The synthetic strategy

Christopher Langton is perhaps the person most responsible for having brought a body of research together under the label "artificial life" (partly by organizing a five-day Artificial Life Workshop at Los Alamos in 1987). He emphasizes the idea that artificial life, like AI, adopts a synthetic approach to understanding the evolution and operation of living systems: researchers build simulated systems out of already-identified components and see what emerges from their operation. In contrast, biologists (and psychologists) primarily take the analytic approach of decomposition and localization in their investigations of naturally occurring systems: starting with a real organism, they figure out what component processes are involved in its functioning and where in the system each process is carried out. Langton writes:

> *Artificial Life is simply the synthetic approach to biology: rather than take living things apart, Artificial Life attempts to put things together*. . . . Thus, for example, Artificial Life involves attempts to (1) synthesize the process of evolution (2) in computers, and (3) will be interested in whatever emerges from the process, even if the results have no analogues in the natural world. (Langton, 1996, p. 40)

Langton's third point follows from what it means to adopt a synthetic strategy. Elementary processes, characteristics, rules, or constraints are first identified by following an analytic strategy in particular species or bodily systems. Once identified, however, they can be put together strategically. For example, an artificial life researcher may build abstract organisms – hypothetical beings that are intended to simulate life at a certain level (the organism) and degree of complexity (usually low) but are not necessarily intended to represent any particular species. The designer can experiment with these abstract organisms by subjecting them to simulated evolution, placing them in a variety of simulated environments, changing certain rules or processes, varying values of parameters, and so forth.

As useful as the synthetic strategy has been in both AI and artificial life, not all investigators would agree with Langton that it is defining of their field. Some view their artificial systems first and foremost as models of some actual system. In AI, for example, the competing pulls between analysis and synthesis can be seen in the fact that some computer programs are constructed to play chess like a human and others are constructed to play chess well. Currently, the programs that play chess well enough to sometimes defeat grand masters do so by following search trees much more deeply than is possible for their human opponents. The computer and human are fairly well matched in skill, but differ in their means. At what point is the difference so great that the program no longer qualifies as an exemplar of a synthetic investigation into intelligence and instead should be viewed simply as a feat of engineering? And how can good use be made of both the (relatively analytic) program that seeks to closely simulate human processes and the (relatively synthetic) program that is only loosely inspired by them?

We can see how the same tension between analysis and synthesis appears in artificial life research by considering Reynolds (1987). To simulate flocking behavior, he constructed a simple model environment and a number of simple, identical artificial organisms (*boids*). In a given simulation run, the boids were placed at different random starting locations in the environment. All moved at the same time but each boid individually applied the same simple rules: match your neighbors' velocities; move towards their apparent center of mass; and maintain a minimum distance from neighbors and obstacles. Viewing the boids' movements in the aggregate, they exhibited flocking behavior – an emergent behavior in which, for example, the group would divide into subgroups to flow around both sides of an obstacle and then regroup. Note that boids are so sketchily drawn that they can stand in for fish as well as birds. Reynolds's work is probably best viewed as a rather abstract investigation into achieving global behavior from simultaneously acting local rules (synthetic strategy), but it could arguably be viewed instead as an initial step towards obtaining a realistic simulation of behaviors observed in several actual species (analytic strategy).

Despite this tension, the synthetic and analytic strategies share the same ultimate goal: to understand the processes of life. This goal imposes its own constraint that the abstract beings must have some grounding in important characteristics of real beings, a grounding that is provided by biologists who have observed the behavior of particular species. The results of synthetic research, in turn, will sometimes suggest new avenues for analytic research. For example, a study like that of Reynolds (relatively synthetic) could suggest particular variables to measure in real birds (purely analytic), and the results might contribute to a more detailed, realistic computer model (relatively analytic). The same interplay of research strategies can be observed in investigations of such activities as perception, food-finding, mating, predation, and communication, all of which have been studied by artificial life researchers as well as biologists in the field. (For an overview of such studies as well as many other kinds of research and issues in artificial life, see the volume edited by Langton, 1995.)

9.2.2 Emergent structures from simple beings: Cellular automata

Perhaps the most abstract studies in artificial life are those involving *cellular automata* – formal systems that were conceived by the Polish mathematician Stanislas Ulam. A cellular automaton (CA) consists of a lattice (a network of cells in which only neighbors are connected) for which each cell is a finite automaton – a simple formal machine that has a finite number of discrete states and changes state on each time-step in accord with a *rule table* (sometimes called a *state transition table*). A CA is defined in part by the size of its neighborhoods. For example, in a one-dimensional CA (a row of cells) the neighborhood of each cell might be the cell itself plus two cells on each side. For each possible configuration of states in a neighborhood there is a rule stipulating the updated state of the target cell on the next time-step. (This should sound familiar: the CA is the same kind of device as a coupled map lattice, used in van Leeuwen et al.'s model of shifting perceptions in section 8.4.2, except that each unit in a CML takes continuous states via the logistic equation rather than the discrete states of a finite automaton.)

The operation of a CA can be illustrated using a one-dimensional array of ten cells, each of which can take just two states: *off* or *on*. We can stipulate that a neighborhood includes only the cell itself and one cell to each side, and that the leftmost and rightmost cells count as neighbors to each other. Then there will be just eight possible kinds of neighborhoods (eight different configurations of states for a cell and its neighbors). For each of them we enter a rule in the table to show which state its target cell should enter on the next time-step. Using the numerals *0* for *off* and *1* for *on*, here is one rule table:

cell and neighbors at *t*	111	110	101	100	011	010	001	000
cell at *t* + 1	0	1	1	0	1	0	0	1

The behavior of any CA is determined solely by the initial pattern of states across its cells and its rule table. For our example, suppose that at time-step 0 the states

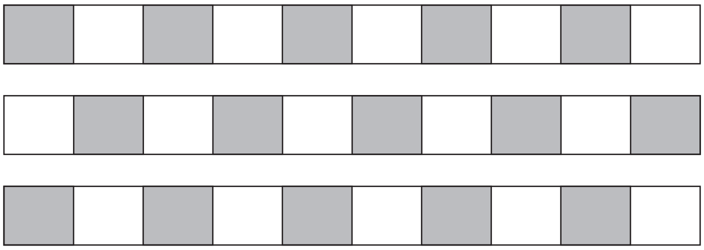


Figure 9.1 A simple outcome of using the rule table in the text. A one-dimensional cellular automaton with ten cells is shown at time-step 1 (top) and at two successive time-steps. Empty cells are on; shaded cells are off.

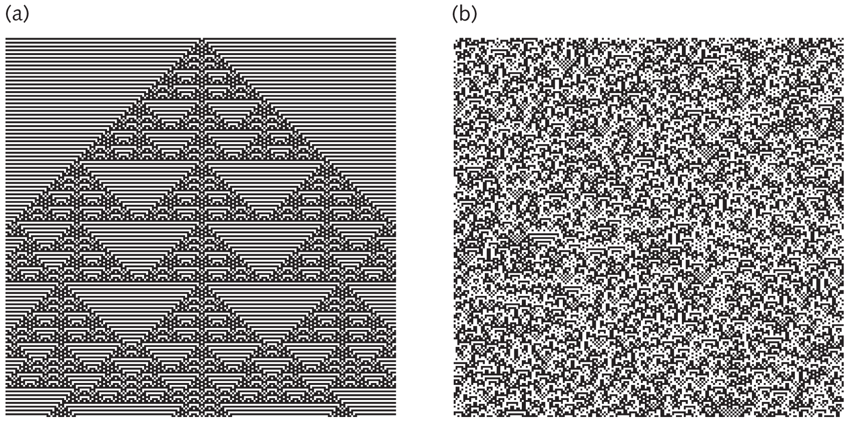


Figure 9.2 More complex outcomes obtained using the same rule table. Each panel shows a one-dimensional cellular automaton with 200 cells at 200 time-steps; each row displays the state of each cell on one time-step. In panel (a) the initial pattern had just one cell *on*, whereas in panel (b) the initial pattern had half of the cells *on* (randomly selected). Figures 9.1 and 9.2 were generated using the cellular automata simulator at http://alife.santafe.edu/alife/topics/ca/caweb/.

happen to form an alternating pattern in which every other cell is *on*, as shown in figure 9.1. Just two of the eight rules will be relevant for this simple case. Each *on* cell (shaded) is flanked by neighbors that are *off* (empty), so at time-step 1 it will turn *off* (*010* → *0*); and each *off* cell is flanked by neighbors that are *on*, so at time-step 1 it will turn *on* (*101* → *1*). The first three time-steps are displayed; clearly this array will keep switching between the *on–off–on–off–* . . . and the *off–on–off–on–* . . . patterns indefinitely.

A great variety of patterns across time can be obtained – many of which are more complex than this repeated switching between two alternating patterns – even without changing to a new rule table. For example, trying two different initial patterns with a larger CA (one row of 200 cells) yields two quite different patterns through time as shown in figure 9.2. (Starting with time-step 0 at the top, each line represents the pattern at the next time-step; the displays were made square by ending at time-step 200.) An initial pattern with just one cell *on* generates the interesting display on the left; one with half the cells *on* generates the more chaotic display on

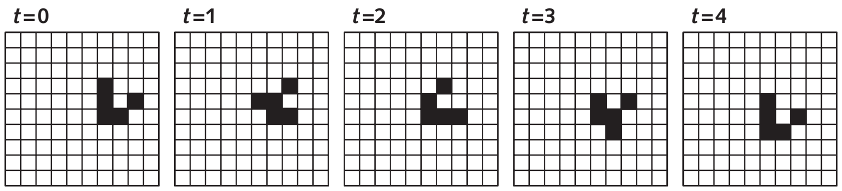


Figure 9.3 A glider in the Game of Life (see text for the rules used to generate it). On every fourth time-step the original shape is restored, but has moved one square left and one square down.

the right. These results were obtained using the CA simulator at http://alife.santafe. edu/alife/topics/ca/caweb/. You can use it to create other CAs (differing in size and rule tables) and explore how different initial patterns change through time.

Cellular automata need not be limited to a single dimension. One of the best-known exemplars is the Game of Life, developed by John Conway (see Gardner, 1970) and used in many screensaver programs. In the Game of Life a computer screen is divided into a large grid of squares. Initially, some squares are shaded (alive) and the rest are empty (dead). Each square has eight neighbors (including those on the diagonals). As time proceeds different squares come alive or die depending on two simple rules:

- If a square is dead on one time-step but has exactly three immediate neighbors that are alive, it comes alive on the next time-step; otherwise, it stays dead.
- If a square is alive on one time-step and has exactly two or three immediate neighbors that are alive, it remains alive on the next time-step; otherwise, it dies.

(Stating these rules in English efficiently summarizes the formal rule table for the 512 configurations that are possible for this size of neighborhood.) The Game of Life attracts attention due to the variety of shapes that can develop. For example, gliders are patterns which move across the screen. Figure 9.3 exhibits a glider which, after every fourth time-step, has moved one square down and one square left; in the intervening steps it transmogrifies into a variety of other forms. Since these shapes and movements are not prespecified in setting up the CA, they are generally construed as emergent structures (as were the movements of flocks of boids in the Reynolds study).

9.2.3 Wolfram's four classes of cellular automata

Different rule tables can yield very different activity, leading Stephen Wolfram (1984) to develop a general classification of cellular automata. Using CAs slightly more complex than those above (by increasing neighborhood size to two rather than one cell per side), exemplars of all four Wolfram classes can be found.

- *Class I automata* enter the same state (e.g., all dead or all alive) from almost any starting configuration, usually in just a few time-steps. If the second line of the rule table in 9.2.2 contained only *0*s, then no matter how many squares were initially alive, they would all become dead on time-step 1 and remain dead. In DST terms, the system settles on a *point attractor* (limit point).

- *Class II automata* form at least one nonhomogeneous pattern (e.g., some squares are alive and others are dead). Typically the system, once beyond any transient patterns, exhibits periodic behavior. That is, it repeatedly cycles through the same sequence of patterns (if the cycle length is zero it will settle to a single static pattern). In DST terms, the system has a *periodic attractor* (limit cycle). Figure 9.1 provides a simple example.
- *Class III automata* are disordered rather than orderly. They exhibit quasi-random sequences of patterns which (were it not for their finiteness) correspond to what is known as *chaos* in DST. The display on the right side of figure 9.2 appears chaotic or near-chaotic.
- *Class IV automata* are the most interesting. They exhibit complex behaviors (e.g., expanding, splitting, recombining) that may be interpreted as realizations of self-organization or computation. Some dynamicists call this *complexity* in contrast to *chaos*. The Game of Life exemplifies this class (see figure 9.3), and van Leeuwen et al.'s coupled map lattice (section 8.4.2), though not a CA, shows comparable behavior when parameter values are chosen so as to produce intermittency.

9.2.4 Langton and λ at the edge of chaos

Christopher Langton (1990) proposed that different values of a parameter, λ, would tend to correspond to different Wolfram classes. Although he explored two-dimensional CAs with 8 states, in our simpler examples λ is simply the proportion of rules in the rule table that have a *1* in the second row; it indicates the potential for cells to be *on* at the next time-step. Langton identified key ranges of values by conducting a Monte Carlo exploration (that is, he generated and ran a large number of CAs varying in λ and initial patterns). There was a great deal of variability in the results, but he sought to capture "average behavior" by calculating several statistics across the CAs tested at each λ. With very small λ, Class I automata tend to occur; when raised towards 0.2, Class II automata emerge. With λ in a range of approximately 0.2 to 0.4, the complex Class IV automata predominate, but as it is raised to values surrounding 0.5 order breaks down and chaotic Class III automata become predominant. Langton referred to the range in which λ tends to produce Class IV automata as *critical values* that are at *the edge of chaos* and proposed that these CAs could be used to perform interesting computations. Since the distributions in fact overlap considerably, a value of λ in the critical range can only suggest that a particular CA is likely to exhibit Class IV behavior; independent evidence would be needed to actually classify it.

The interest in Class IV CAs goes beyond the fact that they can create interesting novel patterns; Langton inspired other researchers to explore their usefulness for computation and problem solving. Norman Packard (1988) focused on a rule table that had earlier been found to perform a useful (though approximate) computation. If more than half of the automaton's cells were *on* initially, usually all of its cells turned *on* eventually (requiring many time-steps, in which the configurations used to determine state updates included three neighbors on each side). If more than half were *off* initially, usually all of its cells turned *off* eventually. If about half were *on* and half *off*, its eventual configuration was less predictable. Hence, it acted as a fairly reliable detector of which state predominated in its own initial pattern – a global property captured via local computations. Packard's innovation was to use a genetic algorithm to evolve additional rule tables that could perform this task. Since the first row of the table has a fixed ordering of neighborhoods for a given number of states (he used 2) and neighbors (he used 3 on each side), CAs could be evolved using genotypes that explicitly represented only the states on the next time-step (the $2^7 = 128$ binary digits in the second row of the the table). A simpler example of a genotype can be obtained from the rule table in section 9.2.2, which has just 8 binary digits due to the smaller neighborhood size:

0 1 1 0 1 0 0 1

The fitness function was provided by the success of the many CAs that evolved (i.e., whether they correctly determined that the initial proportion of active cells was greater than or less than 0.5). Packard was especially interested in the fitness of rule tables with λ in Langton's region of complexity (centered around 0.25 or, on the other side of the chaotic region, around 0.80). He found that they indeed (on average) were best suited to perform the computation.

Packard interpreted his findings as supporting Langton's proposal that interesting computations (class IV automata) emerge in the critical region he identified for λ. However, there is more to the story. A research team at the Santa Fe Institute (Melanie Mitchell, James Crutchfield, and Peter Hraber, 1994) later evolved CAs to perform the same computation, but used a more standard implementation of the genetic algorithm. Contrary to Packard, they found that rule tables with λ values not far from 0.5 performed best and provided a theoretical argument as to why this would have to be the case. While granting that some interesting CAs such as the Game of Life do have λ values in the range Langton identified, they offered their findings as an existence proof against "a generic relationship between λ and computational ability in CA" and concluded there was "no evidence that an evolutionary process with computational capability as a fitness goal will preferentially select CAs at a special λ_c [critical λ] region." They did not, however, deny that relatively simple CAs are characteristic at the extremes of the λ range nor did they evaluate rule tables for other kinds of computation in that paper. In their more recent work (e.g., Crutchfield, Mitchell, and Das, 1998), this team has continued simulated evolution studies of CAs but have focused on applying a computational mechanics framework and a variety of refined quantitative analyses to obtaining "a high-level description of the computationally relevant parts of the system's behavior" (p. 40). This leaves Langton's intriguing proposal about λ as a possible evolutionary dead-end in understanding CAs.

We will end our brief discussion of cellular automata here; it should have given the flavor of the more abstract end of artificial life research. We must skip over a great deal of work in the mid-range of biological realism and complexity, leaving Reynolds's boids as our one example. The rest of the chapter will focus on the evolution of connectionist networks rather than CAs, beginning in section 9.3 with networks that simulate simple food-seeking organisms (which learn as well as evolve) and progressing in 9.4 to network controllers for robots (which develop phenotypes as well as evolve). Robot controllers were our entry point to the science of artificial life in sections 8.3.1 and 9.1, and we look at one additional robot project in 9.5. Finally we return to philosophical issues and implications in 9.6.

9.3 Evolution and Learning in Food-seekers

9.3.1 Overview and study 1: Evolution without learning

If you wish to use networks to control sensorimotor behavior in artificial organisms more complex than cellular automata, how do you get a network that does a good job? A talented designer may quickly arrive at a network that works well for a particular environment and task, but what if some aspect changes? Including a learning procedure has been the traditional way to make networks adaptive. Artificial life research using the genetic algorithm suggests that simulated evolution is another route to adaptivity that is worth exploring. We have already been introduced to the intersection between connectionist networks and artificial life techniques in the work of Beer (section 8.3.1). Here we see how including both kinds of adaptivity in networks simulating simple food-seeking organisms has produced a better understanding of how learning across the lives of organisms can actually have an impact on the evolutionary process. This line of research began with Hinton and Nowlan (1987) and was further pursued by Ackley and Littman (1992) and by Stefano Nolfi and his collaborators. We will sample it in this section by presenting two simulation studies on abstract organisms (Nolfi, Elman, and Parisi, 1994), and then in section 9.4 we will track Nolfi's move to related work with robot controllers (Nolfi, Miglino, and Parisi, 1994).

Nolfi, Elman, and Parisi (hereafter called NolfiEP) invented simple abstract organisms that evolved and learned to traverse a landscape with scattered food sites. Each of these food-seekers was simulated using a very simple connectionist network which encoded and linked a limited repetoire of sensations and motor behaviors. Each network's architecture was fixed but its connection weights were adjusted in the course of learning and evolution. It had four input units: two sensory units encoded the angle and distance of the nearest food site, and two proprioceptive units specified which action the organism had just performed. These two kinds of information were sent through the network's seven hidden units in order to determine which action would be performed next, and the decision was encoded on two output units. After applying a threshold, there were just four possible actions: turn right (01), turn left (10), move forward one cell (11), or stay still (00). NolfiEP's first simulation (study 1) used this architecture for all of its networks. In a second simulation (study 2; see section 9.3.2), two additional output units were added whose task was to predict the next sensory input. The expanded version of the network is shown in figure 9.4, but we will begin with study 1 and the network without the prediction units.

There is another difference between the two studies. In study 1, improvements in food-finding behavior were achieved exclusively by simulated evolution. The main goal was to show that purposive behavior could be sculpted from initially random behavior by applying a genetic algorithm across generations. In study 2, there was another source of change in addition to evolution: learning was used across the lifespan of each organism to modify three of the four sets of connection weights. Here the main goal was to explore how learning and evolution might interact.

An initial population of 100 organisms was created for study 1 by randomly assigning weights to the connections in 100 otherwise identical networks (four input

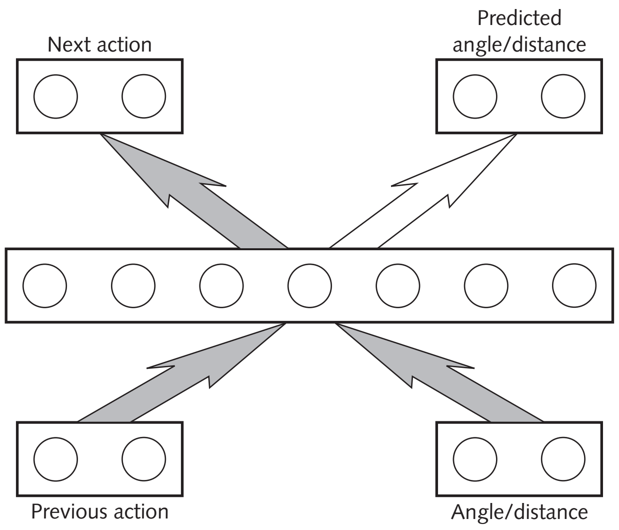


Figure 9.4 The network used by Nolfi, Elman, and Parisi (1994) to simulate abstract food-seeking organisms. Each large arrow is a complete set of connections between units. The three shaded arrows indicate which layers and connections made up the network used in study 1: based on sensory information and the organism's previous action, the next action is determined. The additional output units for predicting the next sensory inputs were added to the network in study 2.

units, seven hidden units, two output units). Each organism lived for 20 epochs, during which it navigated its own copy of a 10 cell × 10 cell environment in which 10 of the 100 cells contained food. In each epoch it performed 50 actions in each of 5 environments (differing in which cells were randomly assigned to contain food); at the end of its life the number of food squares it had encountered was summed. Organisms in this initial generation tended to perform poorly. For example, a typical trajectory in one of these environments, as indicated by the dotted line in figure 9.5, included just one food encounter. Nonetheless, the 20 organisms who happened to acquire the most food were allowed to reproduce. Reproduction was asexual (five copies were made of each organism), and variation was introduced by mutation (in each copy, five randomly chosen weights were altered by a randomly chosen amount). By the tenth generation, the organisms had evolved sufficiently to find many more food squares, with more gradual improvement thereafter. The solid line in figure 9.5 shows a typical path traversed by an organism in the fiftieth (last) generation. In contrast to the earlier path, this one looks purposive. NolfiEP emphasized the importance of achieving lifelike, goal-directed behavior by means of a lifelike, evolutionary process. While acknowledging certain simplifications in their method (e.g., asexual copying of complete networks rather than sexual reproduction with crossover of the genetic codes governing the construction of networks), they found simulated evolution to be a successful and biologically plausible tool for developing networks. They particularly appreciated the biological plausibility of this technique compared to the standard network development technique of supervised learning. Nature provides variation and selection but no explicit teachers.

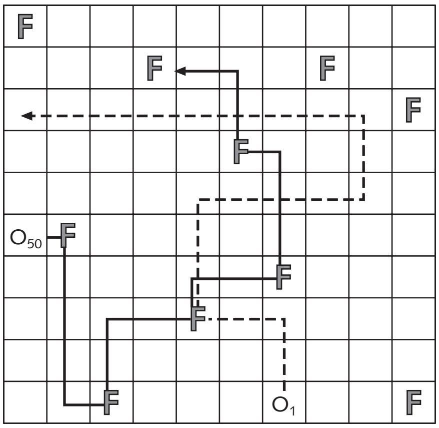


Figure 9.5 Typical trajectories through the 10 × 10 environments of the model organism in Nolfi, Elman, and Parisi's (1994) study 1. The dotted line is a trajectory for a model organism in the first generation; it encountered just one food site. The solid line is a trajectory for a model organism in the fiftieth generation, which encountered six food sites.

9.3.2 The Baldwin effect and study 2: Evolution with learning

Are any roles left, then, for learning? Nolfi and Parisi (1997) discussed three. At the very least, learning augments evolution by permitting adaptations to environmental changes that occur too quickly for an evolutionary response. Learning also enables flexibility, because behavior can be determined by more information than could be encoded in the genome. However, in both of these roles, learning is essentially an add-on that enhances individual performance but does not interact with the evolutionary process. More intriguing is the possibility of a third role for learning: to *guide* evolution. This idea was given its most direct and extreme interpretation in Lamarckian evolution – the discredited nineteenth-century claim that acquired characteristics become directly incorporated in the genome and can be inherited in the next generation. A more indirect way for learning to have an impact on evolution was first suggested by James Mark Baldwin (1896). The basic idea is that successful learners will also be successful breeders, and this source of selection will subtly push evolution in an appropriate direction; across many generations, the genome itself will move towards variations that originally relied on learning. This *Baldwin effect* has been accepted for decades as consistent with a contemporary Darwinian framework, but was often overlooked or misinterpreted. However, Hinton and Nowlan (1987) revived interest by achieving the effect in connectionist networks undergoing simulated evolution and sketching a neat computational interpretation of this heretofore obscure corner of evolutionary theory. They limited their investigation to an extreme case in which only one specific set of weights could render the organism adapted, and all others were maladaptive.

Study 2 in NolfiEP explored how learning could guide evolution by expanding on both the simulations and the computational interpretation pioneered by Hinton and Nowlan. They first added two output units to the original network architecture, as we already have seen in figure 9.4. These units were designed to *predict* the sensory outcome of making the movement encoded on the other two output units – that is, the new angle and distance of the nearest food site. The other major design decision was to make the weights of the connections leading into these new units modifiable by backpropagation. If learning has been successful, the predicted angle/distance should be the same as the actual angle/distance presented to the input units on the next time-step. This allowed for a learning scheme in which the desired or target output pattern need not be supplied by an external teacher, because it is available from the environment as soon as the organism makes its intended movement. That is, the difference between the predicted and actual angle/distance of the nearest food is used as the error signal for learning. Because backpropagation allocates error back through the network, this scheme modifies the weights for all connections except those linking the hidden units to the two original output units for the next action (which have no way of getting a desired action for comparison). Nolfi et al. applied this learning procedure during the life cycle of each organism, and organisms were selected for reproduction in the same manner as in study 1: at the end of each generation's lifespan, the 20 organisms who found the most food were allowed to reproduce. The offspring were created by copying and mutating the *original* weights of the parents, not those acquired by learning. Hence, there was no Lamarckian inheritance of acquired characteristics.

NolfiEP were investigating whether learning might play a useful role in guiding evolution, and their results indicated that it could. Learning during the lifetime of the organisms led to much better performance in later generations – by a factor of two compared with non-learning lineages – even though the descendants could not benefit directly from that learning. NolfiEP's explanation of how selective reproduction and learning interact to produce better organisms in this situation is that learning provides a means for determining which organisms would most likely benefit from random mutations on their weights. An organism that gains from learning is one with a set of initial weights which, if changed somewhat, produce even better results. That would tend to put the good learners into the group selected (based on good performance) to reproduce. By comparison, an organism that does not gain from learning is one whose weights are such that small changes will not produce any benefits. That organism may have found a local minimum in weight space (see figures 3.1 and 3.3). If so, small changes in weights – whether produced by learning or evolutionary changes – will not bring further benefits. Hence, including learning in the life histories of the organisms yields information that permits the evolutionary devices of variation and selection to operate more effectively. NolfiEP's work provides a novel explanation of the Baldwin effect by obtaining it in networks that evolve.

There is another aspect of the interaction between learning and evolution that is noteworthy. Evolution imposes needs on the organism, and learning has improved the organism's ability to satisfy those needs. While labeling the task *food searching* is simply an interpretation, since the organism gains nothing from the food squares in this simplified simulation, nonetheless, the task of visiting certain squares is imposed on the organism by the selection procedure. The fact that learning to predict the environment serves to promote this end is behavioral evidence that visiting food squares has become the goal for the organisms. The activation patterns on the hidden units can be viewed as providing representations of the environment. In the learning task these representations enable the organism to better predict its future sensory input; in the evolutionary task, they permit it to better secure food. Since learning one task (predicting the future appearance of the environment) enhances performance on the other (finding and acquiring food), the representations must carry information that is relevant to both tasks.

We can understand how this might be possible by considering a situation in which the nearest food location is at an angle of 90°. This is information that should lead both to a decision to turn right and to an expectation that after one does, the food will be at approximately 0°. Both the outputs specifying actions and those predicting future angle/distance of food depend upon grouping the input patterns into similarity groups. This is a function served by the hidden units, so the same similarity groups will be available to subserve both tasks. It is in this way that learning to perform one task can facilitate an organism's performance of another task.

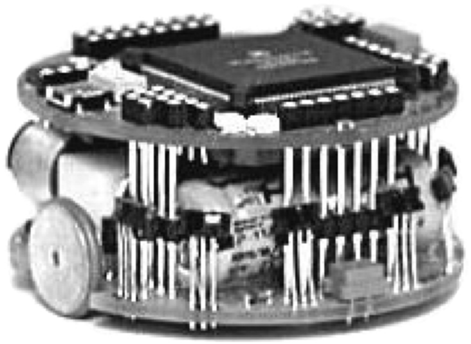
9.4 Evolution and Development in Khepera

9.4.1 Introducing Khepera

Ideally, the interaction of evolution and learning would be studied in a less abstract organism than the food-seekers just discussed. Two of the above investigators joined with another collaborator to take a step forward in complexity by developing networks to control a tiny mobile robot called Khepera (Nolfi, Miglino, and Parisi, 1994; hereafter called NolfiMP). As shown in figure 9.6, it was equipped with physical sensors and motor mechanisms and hence could navigate an actual environment (a 60 × 35 cm arena with walls and a small circular target area). For practical reasons, though, NolfiMP developed the control networks using a simulation of the robot in its environment. (In other studies they addressed the question of how such simulations could be applied to developing controllers for real robots; see below.)

Khepera has a diameter of 55 mm (about 2 inches) and is supported by two wheels and two teflon balls. Each wheel is driven by a small motor that allows it to rotate forwards or backwards. Khepera also has eight pairs of sensors. The light sensors can detect lit-up areas at a range of distances, and the infrared sensors can detect obstacles (objects or walls) in close proximity by bouncing their own light off them. As diagrammed in figure 9.6, there are six front and two rear pairs of sensors. They influence Khepera's movements by means of whatever internal control network is provided. An engineer could quickly design such a network, but then Khepera would be just another robot (one with little practical skill) rather than a simulated lifeform. The real interest is in watching the control networks emerge via lifelike processes of simulated evolution and learning, in pursuit of an ultimate goal of better understanding real evolution and learning. NolfiMP's decision to use a simulated rather than physical robot added another degree of removal from this ultimate goal, but it allowed them the freedom to make some other aspects of their study more complex than would otherwise be practicable.

NolfiMP prepared for their simulation by using the physical robot to generate a pool of sensory inputs and a pool of motor outputs. That is, first they placed Khepera in different orientations and locations in the physical arena, producing a systematic sample of states on its sensors in which the walls and target area would be seen from different angles and distances. Then they gave Khepera's two motors different combinations of commands and recorded its movements. The resulting pools of information were used in constructing a simulated world in which the task was to move towards the small target area in the arena. Simulated evolution and learning



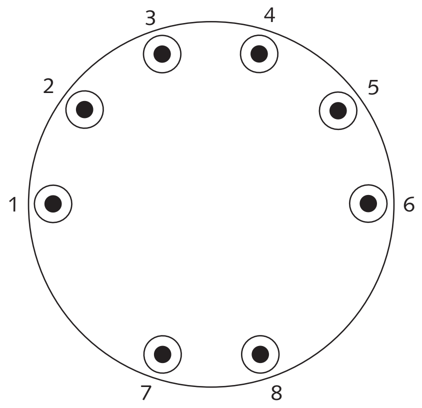


Figure 9.6 The Khepera robot and a diagram showing the locations of its sensors. The filled circles represent the infrared sensors used to detect objects, while the open circles represent light sensors.

interacted to develop networks which adaptively linked the sensory and motor patterns so as to perform this target-seeking task. (Given a different task, the same sensory inputs would get linked differently, though still systematically, to motor outputs – the robot might avoid the target area rather than seek it, for example.)

9.4.2 The development of phenotypes from genotypes

NolfiMP's primary innovation in this particular study was to develop a more biologically realistic model of how a genotype (the design specifications inherited in the genes) figures in the development of a phenotype (the actual organism that results from applying those specifications). In previous studies using networks as artificial organisms, the genotype specified a single phenotypic network. If the network then changed its architecture or weights due to learning in its environment, the genotype played no further role in guiding the resulting series of phenotypes. NolfiMP, in contrast, made the genotype active throughout the life of the organism. Because both genotype and environment influenced the developing network (a series of phenotypes), the same genotype could manifest itself differently in different environments.

In order to create this more biologically realistic genotype–phenotype relationship, NolfiMP used "genes" (structure-building instructions) to produce "neurons" (units) with "axons" (potential connections) that gradually grew into a "nervous system" (neural network). Key points in this process are illustrated in figure 9.7 and described below. The full set of genes – the genotype – ensures that each nervous system is limited to feedforward connections and has a maximum of 17 internal neurons (hidden units, which may be arranged in a maximum of 7 layers), 10 sensory neurons, and 5 motor neurons. Whether a given neuron becomes part of the mature nervous system (i.e., becomes functionally connected within a path from sensory to motor neurons) is determined by the interaction of the robot's genotype and its experiences.

The genotype contains a separate block of genes for each of the 32 possible neurons. Some of the genes specify basic information about the neuron: its location

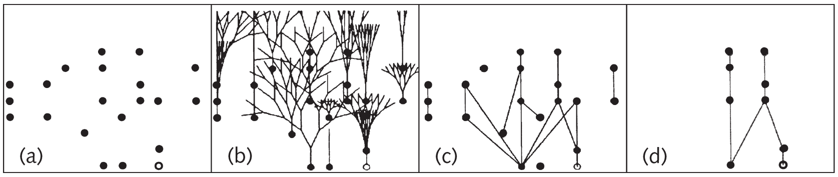


Figure 9.7 An evolved network controller for the Khepera robot at four stages of development: (a) the initial 21 neurons; (b) the growth of branching axons; (c) the network after it has been pruned to leave only the connections where the axon had made contact with another neuron; (d) the functional network. Adapted from Nolfi, Miglino, and Parisi (1994).

in a two-dimensional Euclidean space (suggestive of a vertical slice through a cortical column in a real brain), the weight on any connections to units above it, and its threshold or bias. Additionally, each sensory neuron has a gene specifying to which sensor it is to be connected and whether it detects ambient light or obstacles, and each motor neuron has a gene specifying whether it should be connected to the motor for the left or right wheel. (If more than one motor unit is connected to a given motor, the behavior of the motor is determined by averaging the activations of these units.) Finally, the most interesting genes code for the growth of axons that may connect to other neurons.

Carrying out the basic instructions produces up to nine layers of neurons; the nascent network in figure 9.7(a) has 21 neurons in eight layers. Those in the outer layers are connected to the robot's sensors (at bottom; not shown) or motors (at top; not shown), but initially none of the neurons are connected to other neurons. The genes that encode growth give each neuron the potential to send out an axon which may branch up to four times. One gene specifies the length of each branch and another specifies the angle at which it branches. Realizing this potential depends on experience. The rest of figure 9.7 shows the consequences of applying these instructions and experiential constraints:

Figure 9.7(b): Depending upon the genetic instructions, the branching can yield a sweeping arborization extending up through several layers (e.g., that of the leftmost sensory neuron) or instead can yield arborizations that are narrower and/or shorter. Not all neurons send out an axon, however; this is governed by the expression threshold gene in interaction with experience. If this gene's value is 0, an axon will sprout immediately (maturation with no need for learning). Otherwise, it specifies a threshold value for the variability of the neuron's last ten activation values, which must be exceeded for an axon to sprout. Once axonal growth has begun, a new uncertainty arises: whether any of the branches will contact another neuron. If so, a connection is established.

Figure 9.7(c): The details of axonal branching are omitted and each connection is indicated by a straight line. Some of the connections are nonfunctional, however, because they do not lie on a path extending all the way from the sensory to the motor layer.

Figure 9.7(d): The isolated connections and neurons are omitted, leaving the functional part of the neural network. In this example, it includes just two sensory neurons (one of each type), four internal neurons in three different layers, and two motor neurons. It is the outcome of the joint activity of genes and environment in a single organism within a single generation. We refer to this as "learning" to distinguish it from evolution, but the word is a crude shorthand for a complex process by which a genotype in an environment creates a developmental series of phenotypes in which maturation and learning are intertwined.

9.4.3 The evolution of genotypes

Along with NolfiMP, we next turn our attention to evolution. To cultivate a population of simulated robots, they employed a strategy very similar to that used to cultivate food-seekers in the previous simulation. They began with 100 randomly generated genotypes, each of which was used to obtain one simulated Khepera. Each such robot lived for 10 epochs of 500 actions each; in each epoch, both the robot and the target area were placed in random locations in the arena and the robot produced actions that moved it through the arena. Under the joint influence of its genotype and the inputs it received through its sensors, the robot developed a specific network architecture and assignment of connection weights. The robot's fitness was determined in each epoch by its ability to reach the target area; it was calculated by summing the value of 500–*N* across all 10 epochs, where *N* is the number of actions the robot needed to reach the target the first time in an epoch. The 20 robots who achieved the greatest fitness in a given generation reproduced, creating 5 copies of their genotype (varied by random mutations). NolfiMP also alternated between "light" and "dark" environments. In even generations the robots were placed in an environment in which a light illuminated the target area; in odd generations the light was left off, thus reducing the usefulness of the light sensors.

By examining the best 20 individuals in each generation, NolfiMP were able to show that the robots exhibited considerable increases in fitness (between three- and four-fold) over the first 50 to 100 generations if analysis was limited to the even-numbered generations (light environment). Odd-numbered generations (dark environment) showed a much shallower fitness function (only a two-fold increase). NolfiMP then set out to evaluate the distinctive effects of learning in the two kinds of environment by examining mature phenotypes of the control network. To do this they allowed two clones with the same highly evolved genotype to develop in light and dark environments and compared the two networks that resulted (see figure 9.8). While there are some clear similarities between the networks produced by the same genotype, there are also impressive differences. The network that developed in the light environment came to rely on three infrared sensors and two light sensors, while the network that developed in the unlighted environment relied only on (the same) three obstacle sensors. There were also quite different patterns of connection from sensory neuron 2 to the motor neurons.

9.4.4 Embodied networks: Controlling real robots

We have been speaking of evolution and learning in many generations of robots, but recall that this was actually a simulation study that did not use the actual environment and physical robot after obtaining samples of its sensory and motor events.

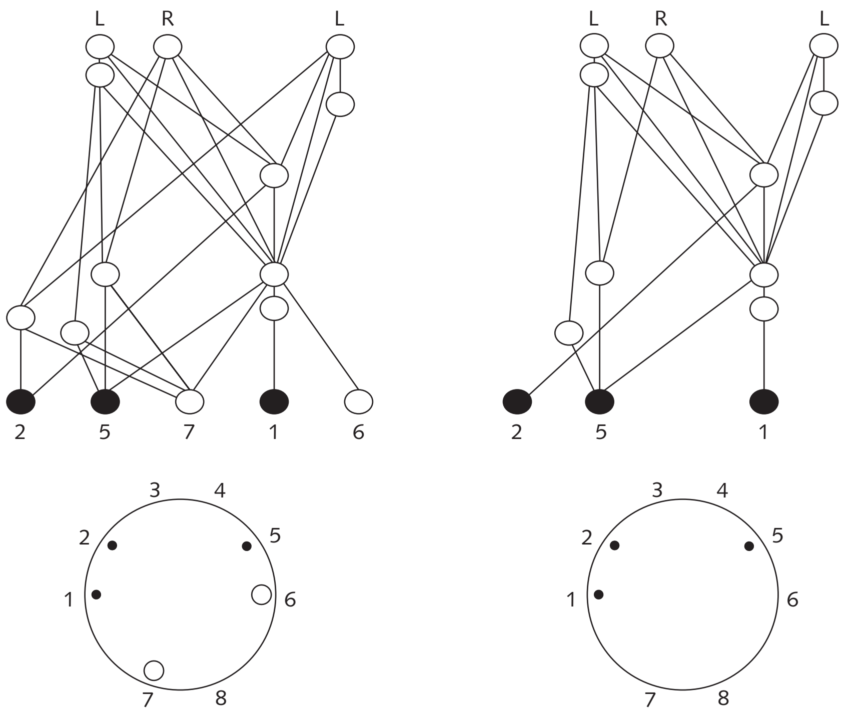


Figure 9.8 Two evolved network controllers cloned from the same genotype. The controller on the left is from a robot that developed in a light environment, whereas the controller on the right is from a robot that developed in a dark environment. Beneath each controller is a depiction of the sensors that were active in each case.

Could a network developed by such simulation techniques be inserted into the physical robot and succeed in controlling it? Nolfi, Floreano, Miglino, and Mondada (1994) explored this question by evolving controllers using (a) real robots and environment, (b) simulated robots and environment, and (c) a hybrid strategy in which controllers developed for simulated robots were inserted into real robots for the last 30 of a total of 330 generations. The robots again were Khepera, but the environment and tasks were a little different than those of Nolfi, Miglino, and Parisi and the networks were simpler and did not learn. In the hybrid study, performance temporarily declined immediately after the network was transferred into a real robot, but then quickly recovered. They noted several ways in which the simulation procedure differed from the real one, and recommended the hybrid strategy as offering an efficient way to get effective controllers.

This last study brings us a step closer to reality. Unlike many artificial organisms, Khepera is embodied. Though its controllers exist as computer code, the robot itself has a physical body that registers light and takes actions in an actual arena. Sampling these physical events lent a degree of realism to NolfiMP's computer simulation of Khepera, but only by hooking the controller network into an actual robot would completely realistic connections between particular sensory and motor patterns get made. When a real Khepera makes a move, it will vary slightly from the planned movement and what is seen next will depend upon the actual movement. By evolving networks in real robots and environments, we assure that they must cope with constraints and variation that cannot be perfectly simulated. Work of this kind provides grounding for those studies that use simulated or hybrid strategies for evolving controllers.

Nonetheless, many aspects of life are still merely simulated in the studies using actual robots. The robot is itself a silicon model of a hypothetical organic being. Its controller simulates a real nervous system, the evolution of its controller simulates a simplified version of biological evolution, and its learning procedures are intended to simulate key properties of learning in real organisms. Langton's synthetic approach to understanding life brings tradeoffs between realism and the degree to which the investigator can manipulate the components of interest and interpret the results. The various studies by Nolfi and his colleagues make different choices about those tradeoffs, and the investigations of cellular automata are even more extreme in their preference for manipulability over realism. There is something to be learned from each of them, and more to be learned from comparing them.

9.5 The Computational Neuroethology of Robots

We have not yet discussed one of the major benefits of using artificial neural networks as robot controllers: investigators can analyze the behavior of their components in much the same manner as neuroscientists use cellular recordings to analyze the activities of individual neurons in real brains. This enables researchers to discover the mechanisms that determine the behavior of the robot in its environment. Such robot studies are part of the field of *computational neuroethology*, a term coined by Dave Cliff (1991). The corresponding studies of living organisms are situated in the parent field of *neuroethology*. (A complete account of the nomenclature and range of studies comprising computational neuroethology would be much more complex; for example, Randall Beer independently coined the term in presenting his studies of artificial insects, as did Walter Heiligenberg for his computer models of real animals.)

To exhibit the potential of this approach, Cliff, Harvey, and Husbands (1997) reported on studies they conducted on networks evolved to control robots moving about in rooms. The robot, the room, and for that matter the network (as usual) were simulated. The robot specifications include a cylindrical shape and three wheels – two in the front which drive it and one in the rear to provide stability. Each of the wheels can turn at five different speeds: full forward, half forward, off, half reverse, and full reverse. The robot has six tactile sensors: bumpers in front and rear and whiskers positioned partway forward and partway back on each side. It is also equipped with two photoreceptors, whose direction of view and angle of receptivity are under evolutionary control. The architecture of the controller networks was evolved through a variation on the genetic algorithm, with an evaluation procedure based on the ability of the robot to move rapidly to a predesignated part of its environment. This placed an emphasis on evolution of the photoreceptors to guide behavior. (No learning procedure was incorporated in these simulations.)

The networks that Cliff et al. evolved in this manner are rather complex, with many backward as well as forward connections. They noted that the networks were very different (better) than what would be created by a human engineer. To simplify the networks for purposes of analysis they removed from consideration all redundant connections and units with no outputs. (The genome itself determined which connections were actually present, in contrast to Nolfi et al.'s inclusion of developmental processes for that purpose as described in section 9.4.2.) They then performed the equivalent of multi-cell recording in real animals – that is, they recorded the activation level of a number of units (in fact, all of them) while the robot performed the activity of interest. To further simplify analysis, they eliminated from consideration those units that were largely inactive during the behavior. This still left a relatively complex network, but one much reduced from the original and in which it was possible to provide a (still complex) description of the flow of activation. To give the flavor, we quote just one portion of the analysis of one robot controller: "Initially, relatively high visual input to unit 6 excites unit 2, which inhibits unit 12, so units 12 and 13 stay inactive. Meanwhile, the effects of visual input arriving at unit 11 give a low-radius turn. Eventually, the robot turns towards the (dark) wall and the visual input falls, so unit 2 no longer inhibits units 12" (Cliff, Harvey, and Husbands, 1997, p. 140). They then commented on how such analysis is similar in character to the analysis one would give of a biological nervous system:

> The task of analysing an evolved neural-network robot controller is similar to the task of analysing a neuronal network in a real animal. The techniques we have employed bear some resemblance to those used in neuroethology, and they give broadly similar results: a causal mechanistic account of the processes by which perturbations in the system's inputs give rise to alteration in the system's outputs. That is, the internal mechanisms of the agent are not treated as a black box, and so it is possible to understand how the observed behaviour is generated. (pp. 149–50)

In addition to a mechanistic analysis focusing on individual units, Cliff et al. also developed a quantitative dynamical analysis that revealed how one controller network, which turned out to be successful in many environments other than the one in which it was evolved, developed dynamical attractors which governed the network's behavior. Like Beer's analysis, this combination of mechanistic and dynamical analysis has the kinds of advantages we discussed in section 8.5.3. Because the analysis involved a much larger network, though, it is harder to extract a higher-level description of the mechanism.

9.6 When Philosophers Encounter Robots

The various research programs that we have surveyed in this chapter – most of which used the genetic algorithm to evolve cellular automata or network controllers for robots – fall broadly under the rubric of *artificial life*. Like the more established sciences, artificial life has attracted the attention of philosophers. Most see it as *raising* new philosophical questions (or new variations on traditional philosophical questions), but at least one (Daniel Dennett) sees it as offering a new method for *addressing* philosophical questions.

9.6.1 No Cartesian split in embodied agents?

For many who have cast a philosophical eye on work in artificial life, one of the most notable features is the emphasis placed on embodying cognitive agents and locating them in environments. For Wheeler (1996), this challenges one of the fundamental Cartesian splits – that between mind and world – that modern cognitivists have tended to accept. The other Cartesian split – Descartes' radical mind–body dualism – is rejected by both artificial-life and cognitive researchers, who generally agree that the mind is simply the activity of the brain. Thus, the axis on which the two clusters differ is that the mind is isolated from the world in cognitivism but coupled to the world in artificial life.

To pursue this distinction, cognitivists view the mind as involving operations over internal representations and develop models of this abstract kind of thinking without giving serious consideration to the ongoing activities of the host organism in its environment. They assume that some sort of interface to sensory transducers and motor effectors will provide links to the environment, but that incorporating these links is not crucial to developing a good cognitive model. In contrast, artificial-life researchers take as their starting point an organism situated in an environment. The primary function of internal processes is to use sensation to control action, so the Cartesian separation of mental processes from worldly events is avoided. If more abstract processes develop, it is assumed that their form and functioning would be influenced by the more basic sensorimotor processes; they would not be studied in isolation. Moreover, insofar as the internal control systems constantly engage with the physical body and environment (e.g., by receiving new sensory input as a result of moving the body), the last vestiges of the Cartesian scheme are overcome. Instead of a Cartesian scheme, Wheeler suggests that such artificial life embodies a Heideggerian perspective wherein agents begin by being actively engaged with their world through skills and practices; whatever cognitive reflection occurs, it is grounded in this activity.

9.6.2 No representations in subsumption architectures?

For roboticist Rodney Brooks (1991), pursuing this program has generated some fresh ways of thinking about the mind's architecture. He builds a separate control system for each task a robot must perform; each is a hard-wired finite state machine which Brooks calls a *layer* (a very different use of the term than in connectionism). For example, in his simplest robot the first layer specializes in avoiding obstacles, the second layer generates wandering (randomly determined movements), and the third layer generates exploration (moving towards a particular location). Each layer is a complete system extending all the way from sensation to action and is capable of acting alone. Typically, however, they can be active simultaneously; minimal circuitry between layers manages when one layer will suppress another, or sometimes calculates a compromise direction of movement. Brooks calls this a *subsumption architecture*, and it follows from deciding to decompose activities horizontally by task rather than vertically by function. A function such as visual perception may be carried out separately and differently within several layers of the system; there is no overall, shared vision module in the sense either of Fodor or of traditional AI. This means there are no central representations. Even within a layer, information is accessed from the real world as needed, obviating the need to construct and keep updating an internal model of the world. Brooks argues, moreover, that the states of the various registers of the finite state machine constituting each layer do not even qualify as local representations: they do not involve variables or rules, and individual states do not have semantic interpretations (rather, the whole layer is grounded in the world it moves in).

Critics, however, are dubious of the ability of such a system to simulate all or even most forms of intelligent behavior (Brooks himself suggested that it will suffice for 97 percent). Kirsh (1991), for example, allowed that it could suffice for an activity such as solving a jigsaw puzzle. A person might look at the shape required at a particular gap and compare it to the shapes of available pieces, and might attempt to physically position each of the pieces in the gap if necessary; the task requires minimal cognitive activity and no complex models. But he argued that a host of important activities cannot be accomplished through such direct linkages to the environment – activities that require: (1) predicting the behavior of other agents; (2) taking into account information not presently available (e.g., precautionary activities to produce or avoid future outcomes); (3) taking an objective, not an egocentric point of view (e.g., obtaining advice from other agents); (4) problem-solving prior to action; and (5) creative activities (e.g., writing poetry).

The objections raised by Kirsh (and similarly by Clark and Toribio, 1994) seem telling against extreme antirepresentationalist positions such as that of Brooks and some others engaged in the artificial life movement. But what if a Brooks-style architecture were used as the starting point for a new system responsive to Kirsh's objections? Its designers could build sensorimotor controllers specialized for various activities in the world as a foundation, but then add the capacity to reason and solve problems in ways that partially decouple the internal system from the environment. Such a system could develop plans and strategies that enabled it to respond more effectively to situations that arose, and thereby enhance its evolutionary prospects (for a suggestion along these lines, see Grush, 1997a).

9.6.3 No intentionality in robots and Chinese rooms?

One attractive consequence of starting with an embodied system acting in the world is that whatever cognitive, representational processes are built on top of that will have their representations grounded in the world. Researchers taking this kind of hybrid approach might thereby hope to overcome the problem of accounting for intentionality in AI systems that was posed by John Searle (1980). The term *intentionality* refers to the character of linguistic utterances or thoughts as being about something and in that respect having meaning (e.g., the thought that John Searle teaches at Berkeley is about the actual person John Searle). Searle makes his case against AI systems by offering his Chinese Room thought experiment, in which a person or machine simulates the behavior of a speaker of Chinese using Chinese characters to engage in a written interaction. The simulation is accomplished by repeatedly consulting a handbook of rules for manipulating the Chinese characters as formal symbols (i.e., shapes with no meaning). Searle contends that the person or machine engaging in such symbol manipulation, because they do not understand the Chinese characters they are responding to or producing, does not have the (intentional) thoughts of the real Chinese speaker being simulated. If embodied artificial life systems are elaborated so as to develop representations that stand in for aspects of the environment with which they are interacting, then, unlike the purely formal system operating in Searle's Chinese room, these representations might be credited as exhibiting intentionality.

Beyond those projects using actual robots as embodied artificial organisms, much of the research in artificial life relies on simulation techniques. Even Nolfi, Miglino, and Parisi's (1994) work on evolving network controllers for Khepera relied upon off-line samples of the actual robot's experiences (a kind of simulation). If controllers for simulated robots are viewed as employing representations in their hidden layers (which Brooks might deny but many others would assert), Searle's objection would seem to arise again – the representations are completely formal and lack content. If indeed the network controllers evolved in such simulations turn out to be virtually equivalent to those evolved in real robots, on one view, this draws into question the intentionality of the representations in the real robots as well. This is likely the view Searle would adopt, since he uses the Chinese Room thought experiment to demonstrate, by contrast, the causal powers of real brains in producing intentional states. But an alternative interpretive stance on the simulated controllers would regard their internal states as representations insofar as they evolved in response to the evolutionary and learning conditions imposed by their simulated environment (the similarity to controllers evolved for real robots would be due to similarities in those conditions). We cannot hope to settle here the issues concerning the intentionality of representational states that develop in simulated and real robot controllers. However, this aspect of robotics clearly exemplifies how artificial life research can engage long-standing philosophical concerns.

9.6.4 No armchair when Dennett does philosophy?

Perhaps the most radical suggestion about artificial life is Daniel Dennett's (1995) proposal that it offers not just a new domain that philosophers can pounce upon and subject to their usual methods of analysis and criticism; more interestingly, it can provide a new method for doing philosophy itself. One of the traditional methods, as just illustrated by Searle's Chinese Room, is to pose thought experiments. The philosopher generates interesting circumstances (often fanciful or contrary-to-fact, but regarded as diagnostic) and reasons about their likely consequences. The results generally are far from definitive; philosophers with different intuitions, biases, and other limitations arrive at different consequences. Dennett suggests that building artificial life simulations offers an improvement upon this method, because it offers a way for elaborate thought experiments to be not merely created but also rigorously tested. If what the philosopher imagines can be realized (implemented) in the design of an artificial life simulation, then running the simulation will reveal the consequences. Philosophers inspired to add this technique to their toolkit will be able to break new ground in posing and testing ideas. As one example, Dennett points to the question of how a complex phenomenon like cooperation might emerge. If a philosopher's tentative answer can be realized in an artificial life simulation which indeed produces cooperative behavior (e.g., that of Ackley and Littman, 1994), this should be a more convincing way to evaluate the proposal than to reason about why the tentative answer might or might not succeed (the time-honored, perhaps time-worn, armchair method of doing philosophy). A similar issue that Dennett indicates might lend itself to investigation through artificial life modeling is whether highly purposive intentional systems with beliefs and desires might evolve gradually from simpler automata. The further the method is stretched beyond its biological roots, the wider is the range of traditional philosophical issues that can be opened to new insights.

We have touched on several ways in which successful simulations are likely to help advance philosophical inquiry; of these, Dennett's rethinking of thought experiments probably has the greatest implications for philosophy as a discipline. However, it can be expected that many philosophers will respond to simulation results by questioning whether the apparent cooperative behavior or intentionality are real instantiations of these constructs, or are pale imitations of uncertain consequence. Do the simplifications and abstractions involved in any simulation compromise the epistemic status of its outcome? Many rounds of argument could emerge from a skeptical comment of this kind, and they would remind us that simulation or any other new method would only augment, not replace, reasoning and debate as a way of doing philosophy.

9.7 Conclusion

The networks we discussed in previous chapters had been hand-crafted by humans and employed to solve problems that had been encoded by human designers on their input units. In this chapter we have explored early efforts to expand the scope of connectionist research so as to avoid these two constraints. By installing networks into robots as controllers, they are not restricted to what the researcher wants them to learn; they incorporate the regularities in an actual environment whatever those turn out to be. The genetic algorithm provides a way of evolving network designs; these evolved designs may turn out to be far more brain-like than those created by human designers if their early promise is realized. The final point considered here was that the lines of research discussed in this chapter are also ripe for philosophical analysis, both by raising philosophical questions such as whether the representations developed in embodied robot controllers achieve genuine intentionality and by offering a vehicle for carrying out philosophical thought experiments.

Sources and Suggested Readings

Adami, C. (1998) *Introduction to Artificial Life*. New York: Springer-Verlag.

Boden, M. A. (ed.) (1996) *Artificial Life*. Oxford: Oxford University Press.

Clark, A. (1997) *Being There: Putting Brain, Body, and World Together Again*. Cambridge, MA: MIT Press.

Langton, C. G. (ed.) (1995) *Artificial Life: An Overview*. Cambridge, MA: MIT Press.

Morris, R. (1999) *Artificial Worlds: Computers, Complexity, and the Riddle of Life*. New York: Plenum Trade.

Nolfi, S. and Florano, D. (2000) *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-organizing Machines*. Cambridge, MA: MIT Press.

Steels, L. and Brooks, R. (1995) *The Artificial Life Route to Artificial Intelligence: Building Embodied, Situated Agents*. Mahwah, NJ: Lawrence Erlbaum.