10

Computing with Neurons

In this chapter, we give preliminary consideration to how the components of a computing machine might be implemented with neurobiological elements. We return to this question in a final chapter of a much more speculative character. We stress, however, that *all* contemporary attempts to spell out how computations are realized in the brain are speculative. Neuroscientists do not know with any confidence the answers to even the most basic questions about how computations are carried out in neural tissue. There is, for example, no consensus about what the primitive computational operations are, out of which other computations are constructed. Indeed, the question, "What are the primitive computational operations in neural tissue?" has only rarely been posed (Koch, 1999; Koch & Poggio, 1987), let alone answered to general satisfaction (Mel, 1994). Although this question strikes most engineers charged with building a machine that computes as about as elementary and basic and unavoidable a question as one could well ask, it strikes many neurobiologists as an odd question. It strikes them as odd because, broadly speaking, neurobiologists are not focused on understanding the nervous system from a computational point of view. If forced to answer, many would probably suggest that there is no set of primitive computational mechanisms. That is, of course, one possible answer to the question (cf. Koch, 1999, pp. 471ff.). It is, however, tantamount to saying that every time a different computation has appeared in the evolution of the brain's computational capacity, it has arisen *de novo*. Unlike other aspects of biological structure, more complex computational structures have not been fashioned from computational components already available. That's possible, but why should the principles that govern the evolution of computational structure be different from the principles that govern the evolution of other structures? All other biological structures are seen to arise by recombination of basic elements.

There is also no consensus about even such a basic preliminary question as how information is encoded in spike trains. It is generally supposed that spike trains serve as the inputs to the brain's computational machinery. As we have seen, the form of a mechanism that implements a computational operation is fundamentally dependent on the form in which the relevant information is encoded in the symbols on which it operates. If there is no consensus about how the values of simple variables are encoded in spike trains, there is *a fortiori* no secure knowledge about

how computational mechanisms in the brain combine the information from two or more such trains. If, for example, we do not know how the durations of intervals are encoded in the input, we cannot know how the brain computes their sum, difference, or ratio.[1]

Transducers and Conductors

There are, however, elements of the problem of neural computation for which there are widely agreed upon answers. Sensory receptors are the transducers, and muscles and exocrine glands are the effectors. About that, there is no doubt. There is also no doubt that the action potential is the mechanism by which information is rapidly transmitted over "long" distances (distances greater than 10 microns). However, it is important to appreciate how slow this transmission is relative to what the builders of computational machines are accustomed to work with. It takes an action potential on the order of 10–100 microseconds to travel 100 microns, which is a more or less typical distance separating the neurons within a local circuit, such as a cortical column. The time taken for information to travel between similarly spaced electronic elements is shorter by a factor of about 100,000,000. This is a sobering difference when one reflects that minimizing the time for information to travel between computational elements is an important consideration in the design of modern computers. It makes one wonder whether it is plausible to assume that elementary computations in neural tissue are implemented by neural circuits in which the relaying of action potentials between the neurons is an essential part of the operation. This assumption is all but universal in contemporary neural network modeling (see, for example, the neural network model of dead reckoning reviewed in Chapter 14). Any such implementation will be almost unimaginably slow by the standards of contemporary electronic computing machines. The alternatives are that the elementary computational operations are implemented within single neurons, either by subcellular structures like dendritic spines (Koch, 1999, ch. 12) or at the molecular level (see Chapter 16).

On the other hand, the axonal conduction mechanism readily implements a function that it is awkward to implement within the marble machines we used to illustrate the physical realization of basic computational functions, namely, the transmission of a signal from a single source to multiple destinations. Axons branch prolifically. When an action potential arrives at a branch point, action potentials propagate down both branches. By contrast, a falling marble must fall in either one branch

[1] Readers interested in further pursuing the question of how information is encoded in spike trains may want to begin with the much praised text by Rieke et al. (1997) or the short treatment in Koch (1999, ch. 14). For more recent work, see Averbeck, Latham, & Pouget, 2006; Bialek & Setayeshagar, 2005; Brenner, Agam, Bialek, & de Ruyter van Steveninck (2002); Brenner, Bialek, & de Ruyter van Steveninck (2000); Brenner, Strong, Koberle, Bialek, & de Ruyter van Steveninck (2000); Deneve, Latham, & Pouget (2001); Fairhall, Lewen, Bialek, & de Ruyter van Steveninck (2001); Latham & Nirenberg (2005); Nirenberg & Latham (2003); Schneidman, Berry, Segev, & Bialek (2006); Simmons & de Ruyter van Steveninck (2005); Strong, de Ruyter van Steveninck, Bialek, & Koberle (1998).

or the other; it cannot give rise to two marbles falling in two different branches. Sherrington (1947 [1906]) emphasized the functional importance of this *divergence*, the ability to send the same signal to many different processors. Computer scientists often call this "fan out."

Sherrington also recognized the importance of the fact that action potentials *converge* at synaptic junctions. When two or more axons synapse on the same postsynaptic neuron, the effects of signals in these axons combine to determine whether the postsynaptic neuron does or does not fire. From a computational perspective, this convergence and the mechanism of signal transmission across synaptic junctions makes possible the implementation of the AND, OR, and NOT functions. Computer scientists call this property of a computing machine's architecture "fan in." The brain exhibits massive fan out and fan in relative to what is seen in contemporary computing machines, and this may be a key to understanding how it is able to make complex computations as rapidly as it does.

Synapses and the Logic Gates

Synapses are the junctions between neurons, with a presynaptic ending on one side of a very narrow cleft between two neurons and a postsynaptic membrane on the other side. Action potentials do not propagate across synapses. Rather, the arrival of an action potential at a presynaptic ending releases transmitter substance into the cleft. The binding of the transmitter substance to receptor molecules embedded in the postsynaptic membrane on the other side of the cleft alters ionic conductances in the postsynaptic neuron. The alteration is direct, in the case of ionotropic receptors. That is, the ion gate is a component of the transmembrane receptor to which the released transmitter substance binds. In the case of metabotropic receptors, the transient change in one or more ionic conductances is produced indirectly by way of an intracellular cascade of molecular reactions. In other words, it is mediated by a molecular signal transmission process within the postsynaptic neuron. The existence of this mechanism links extracellular signaling to intracellular signaling, allowing the presynaptic signals to alter the internal operations of the postsynaptic neuron. Whether directly or indirectly produced, the alteration in ion-channel conductances may either transiently depolarize the postsynaptic membrane (an Excitatory PostSynaptic Potential, or EPSP) or transiently hyperpolarize it (an Inhibitory PostSynaptic Potential, or IPSP).

Inhibitory postsynaptic conductance changes cancel excitatory changes, thereby implementing the NOT function.

The depolarizing effects of contemporaneous EPSPs sum to produce a greater depolarization. Whether this depolarization is or is not sufficient to initiate a conducted action potential in the postsynaptic neuron depends on how large it is relative to the threshold for action potential initiation. If either of two EPSPs acting alone depolarizes the membrane beyond the threshold for initiating an action potential, the circuit comprised of those two presynaptic neurons and their common postsynaptic neuron implements the OR function.

If neither EPSP acting alone is suprathreshold but the two together are, the circuit implements the AND function.

The nonlinear thresholding that plays a critical role in implementing these indispensable logic functions need not be the threshold for the initiation of a spike at the axon hillock (where the axon originates); it may occur more locally within single dendrites (Koch, 1997). Thus single dendrites may act as logic gates. Dendrites are tree-like structures branching out from the cell bodies of neurons and covered with presynaptic endings.

The Slowness of It All

Synaptic integration processes are an order of magnitude slower than the already slow signal transmission process. From the time that an action potential arrives at the presynaptic ending until the EPSP builds to its peak is on the order of 500 microseconds (half a millisecond), while the decay from the peak response back to the resting level of polarization typically takes several milliseconds. A millisecond is an eternity in a modern computer. The computer on your desktop executes an entire floating point operation (e.g., adding two real numbers) in a thousandth of a microsecond or less. The execution of a floating point operation requires a great many ANDs, ORs and NOTs, the retrieval of symbols for both quantities from memory, and the writing of the symbol for the result back to memory.

We do not understand how computational elements operating so slowly, with such long communication delays, can execute what would seem to be extremely complex computations as fast as the brain in fact does execute them. The brain can, for example, process to the point of scene recognition and analysis ("picnic," "wedding," "baseball game," etc.) streams of disparate images coming at one image every 100 milliseconds (Potter, Staub, & O'Connor, 2004). We do not currently know any algorithm capable of recognizing and analyzing the content of these complex images anywhere near as well as the human observer does, even when given as much time as the algorithm likes to process each image. However, insofar as we currently understand this problem (image parsing, recognition, and analysis), a remotely adequate algorithm would appear to require accessing a great deal of diverse stored information and making a large number of high-level computations, each involving an even larger number of elementary computations.

How it is possible for something as slow as the brain to process 10 scenes per second is a major mystery. Contemporary neural network models are by and large models of recognition or categorization. They model a content addressable memory, in which the input of a portion of a previously experienced input vector induces the net to settle into a pattern of activity characteristic of the category to which that input belongs. The settling into a distinct pattern of activity is taken to be an implementation of the recognition or categorization computation. This settling into a stable pattern of activity (a so-called "attractor state") is an example of a computational mechanism that is fundamentally dependent on repeated (recurrent) signal transmission between thousands of neuronal elements. No one has any idea

how a system based on this mechanism would be able to categorize images at the rate of 10 images per second.

The standard neurobiological answer to how it is possible for the brain to do seemingly very complex computations so fast is "massive parallelism": it breaks the problem down into innumerable very simple computations, all of which are carried out simultaneously by many different microcircuits. There may be an important element of truth in this answer. However, at this time in the development of cognitive neuroscience, this answer is mostly a cloak for ignorance. We do not know to what extent it is possible even in principle to break this and many other computational problems down into elements that may be processed simultaneously. Much less do we know how such a breakdown could be accomplished, how the elements of the problem could be directed to the appropriate microcircuits, and how the results reported from all these circuits could be collated and integrated to arrive at an answer, all with such rapidity that a throughput of 10 images per second is possible. Computer scientists have been studying massively parallel computation for decades. One thing they have learned is that it is not easy to implement, except in the case of some rather special problems. Thus, if massive parallelism is the answer, then this poses the question what the architecture of the brain tells us about how to implement massively parallel computation. To that question, there is at present no clear answer, although one suspects that the above-mentioned massive fan out and fan in are part of the answer.

There is also the following puzzle. The modern computer carries out elementary computational operations a billion to a trillion times faster than the above numbers about the speeds at which axonal and synaptic transmission mechanisms operate suggest that the brain does. Doing computations in parallel rather than in sequence trades spatial resources (the number of computational mechanisms working on different parts of a problem at any given moment) for temporal resources (the number of different parts of the problem one mechanism can process in a unit of time). Because the modern computer appears to have many orders of magnitude greater temporal resources, why can it not outperform the brain by doing the same number of operations in sequence in less time than the brain can do them all at once in parallel? Either we are seriously deceived about the rate at which neural tissue can perform elementary computational operations, or the brain has discovered much more efficient algorithms (algorithms that minimize the required number of operations), or both.

The Time-Scale Problem

Nonetheless, if we set aside puzzles about how the brain is able to execute computational operations as fast as the behavioral evidence implies that it must, then in his book, *Biophysics of computation: Information processing in single neurons*, Koch (1999) shows how experimentally demonstrated mechanisms within individual neurons are sufficient to implement a variety of computational operations. He pays particular attention to multiplication because it is an elementary instance of the nonlinear operations that are an essential aspect of computation. What he mostly

avoids, however, is dealing with memory. In the opening paragraphs of his Introduction (p. 1), he writes:

> The brain computes! This is accepted as a truism by the majority of neuroscientists engaged in discovering the principles employed in the design and operation of nervous systems . . .
>
> The present book is dedicated to understanding in detail the biophysical mechanisms responsible for these computations. Its scope is the type of information processing underlying perception and motor control, occurring at the millisecond to fraction of a second time scale.

In limiting its scope to information processing on a time scale of less than a second, Koch avoids the biggest problem, the problem of relating the time scale of the known neurobiological mechanisms and processes to the time scale of behavior. The sub-second time scale is the time scale of neurobiological processes, but it is only the extreme short end of the time scale on which behavior unfolds. Most behavior is shaped by information that has been received over a time scale ranging from the most recent tenth of a second back over minutes, hours, days, months, and years. In order for behavior to be thus shaped, there must be mechanisms in nervous tissue that undergo changes of state that last minutes, hours, days, months, and years. These changes must encode information about where things are, where the animal is, how long properties last, what the surrounding scenery looks like, when in the past various events took place, and so on – all the information that demonstrably informs behavior. The function of the enduring physical changes that encode this information is to carry previously acquired information forward in time, so that it is available whenever needed in the computations that inform current behavior. Koch describes in great detail the biophysics of the spike propagation mechanism, which carries information from place to place within nervous tissue, making it available *where* it is needed. In striking contrast, he has relatively little to say about the mechanism that carries information forward in time, making it available *when* it is needed.

The time scale that Koch limits most of his discussion to is the time scale on which the changes of state in the processor component of a computing machine may be expected to endure. The processor of those computing machines whose operation we understand has only a modest number of states; it remains in any one of them only briefly; and it revisits its various states over and over again in the course of its operations. This is a reasonable first-order description of what we seem to observe in neurons. A given neuron can be in a modest number of different states. None of them typically lasts more than a fraction of a second once the stimulus or input that induced it is gone. The neuron revisits those states over and over again as new inputs come in.

Synaptic Plasticity

What is missing from the contemporary conception of neurobiological dynamics is what we see when we look at the memory elements of a computing machine. Memory

elements have three essential aspects, only one of which is widely appreciated by contemporary neuroscientists: (1) they undergo *enduring* changes, changes that last indefinitely; (2) the changes encode information; (3) the changes can be read, that is, they carry information forward in time within a mechanism that makes the information accessible to future computation.

The focus of neurobiological thinking about memory has been entirely on the first aspect, that is, on finding an enduring *functional* change. (The effects of, for example, vascular accidents, endure, but they are not functional; they do not enable the brain to better direct the animal's behavior.) Finding an enduring functional change is of central importance: a change can carry information forward only so long as it endures. However, the other two aspects are equally important. If the change cannot encode information in a readable form, then it cannot fulfill the role of the memory elements in computing machines. Enduring functional changes that do not encode information in a readable form are found in the state changes of the processor. There, their inability to encode information is not a problem, because their function is to alter the processor's response to inputs, not to carry information forward in time. And that is how in fact most contemporary neuroscientists conceive of memory: they conceive of it as a change in the state of the processor, a change that makes the processor respond differently to previously experienced inputs. That is why neuroscientists prefer to talk about mechanisms of *plasticity*, rather than mechanisms of memory. The conceptual framework behind the term "plasticity" goes back at least to the British empiricist philosophers of the seventeenth century and arguably to the conception of mental function advanced by Aristotle. The idea is that experience molds the brain so that it behaves more appropriately within the experienced environment. "Conceptually, this amounts to the input changing the transition function governing how the machine switches from one state to the next. Put differently, a nervous system will act like a machine that changes its instruction-set as a function of its input" (Koch, 1999, p. 470).

Contemporary neuroscience is committed to the thesis that the brain has the functional architecture of a finite-state automaton, rather than that of a Turing machine: it lacks a read/write memory. Koch (1999, p. 471) summarizes this view as follows:

> Memory is everywhere, intermixed with the computational elements, but it cannot be randomly accessed: in the concentration of free calcium in the presynaptic terminal and elsewhere, implementing various forms of short-term plasticity, in the density of particular molecules in the postsynaptic site for plasticity on a longer time scale, in the density and exact voltage dependency of the various ionic currents at the hour or day time scale, and ultimately, of course, in the genes at the cell's nucleus for lifetime memories.

A conspicuous feature of his catalog of memory mechanisms is the absence of any suggestion about how any of them could encode, for example, the duration of an event, or the direction and distance from the nest or hive, or the color and shape of a food source. All of the things that Koch lists are state variables, not memory elements, as a computer scientist understands memory.

The thesis that the brain has the architecture of a finite-state automaton rather than of a Turing machine makes it possible to ignore the coding problem as well as the problem of how the machine accesses the encoded information. The finite-state automaton does not read information from memory. Its current state simply determines its action. We illustrated this in Chapter 8 by showing how the previous sequence of marbles determines the positions of the rockers (that is, the state of the machine), and their position determines which channel an incoming marble will fall out of. The external observer, who is a god outside the machine, can determine from the position of the rockers what the preceding sequence must have been, but the machine has no need (and no mechanism) to read this from any memory; its current state determines its input–output function.

In Chapter 8, we were at pains to explain the limitations of the finite-state architecture. Our focus there was on the tendency of the finite-state architecture to be overwhelmed by the infinitude of the possible. A further difference that we did not mention is that in the Turing architecture, which stores the sequence in a read/write memory (see our marble shift register, Figure 8.12), a subsequent output can be made to depend on any part of the stored sequence. When the memory is read – when the retained marbles in our shift register are released, the machine recovers the whole sequence. It can use any part of that sequence in the next computation that it makes. By contrast, in the finite-state architecture, the action of the machine (the output channel in which the marble falls) depends entirely on the current state of the processor, which has been determined by the entire sequence. Thus, it is only possible to have differential responses to entire sequences of some necessarily pre-specified length. It is not possible, for example, to have a response that depends on whether the second half of a sequence (length not pre-specified) is the same as (or the inverse of, etc.) the first half. In other words, the absence of a readable memory radically limits computational power. The finite-state architecture requires the designer of the machine (or, in the neurobiological case, the epigenetic process) to provide in advance for every possible functionally relevant contingency. Historically, much of the inspiration behind this pervasive thesis about learning and memory has been empiricism, the belief that the brain's structure is largely determined by its experience, which has put the machine in the state it is in. There is little appreciation of the radical nativism implicit in it: the necessity of providing structurally in advance for every possible behaviorally relevant state of the world.

The plastic change that neuroscientists generally appeal to in order to explain the phenomena of memory is synaptic plasticity, an enduring functionally appropriate change in synaptic conductance produced by experience.

The idea that memory involves a change in synaptic conductance also has great appeal for psychologists, because it is the neurophysiological realization of a mechanism that has been at the center of psychological (and before that, philosophical) thinking about learning and memory for centuries, the mechanism of association (Fanselow, 1993; Gluck & Thompson, 1987; Hawkins & Kandel, 1984). Many contemporary psychologists feel strengthened in their belief that associative learning theory must be in some sense basically the correct psychological theory by what they take to be the relevant findings in neurobiology. They find in the neurobiological literature experimental evidence for a mechanism that produces enduring changes

in synaptic conductance when there is a temporal coincidence of pre- and postsynaptic signals, and they take this as support for associative theories of learning and memory in psychology.

It is important to realize that the evidence for the hypothesis that a change in synaptic conductance is the neurobiological basis of memory is weaker than many psychologists appreciate. Koch (1999, p. 308) accurately sums up the current state of the neurobiological evidence when he writes:

> For over a century (Tanzi, 1893; Ramon y Cahal, 1909, 1991), the leading hypothesis among both theoreticians and experimentalists has been that *synaptic* plasticity underlies most long-term behavioral plasticity. It has nevertheless been extremely difficult to establish a direct link between behavioral plasticity and its biophysical substrate, in part because most biophysical research is conducted with *in vitro* preparations in which a slice of the brain is removed from the organism, while behavior is best studied in the intact animal. In mammalian systems the problem is particularly acute . . . Even in 'simple' invertebrate systems, such as the sea slug *Aplysia* . . . it has been difficult to trace behavioral changes to their underlying physiological and molecular mechanisms. Thus, the notion that synaptic plasticity is the primary substrate of long-term learning and memory must at present be viewed as our most plausible hypothesis. (Milner, Squire & Kandel, 1998)

The evidence that psychologists have in mind when they appeal to neurobiology to justify associative theories of learning and memory comes from the voluminous literature on long-term potentiation (LTP), which is "a rapid and sustained increase in synaptic efficacy following a brief but potent stimulus" Koch (1999, p. 317). In capsule summary of the conceptual impetus behind this complex literature, Koch (1999, p. 317) writes:

> LTP research is very popular: between 1990 and 1997, over 2000 papers on LTP were published. The excitement stems in large part from the hope that LTP is a model for learning and memory, offering the most direct link from the molecular to the computational and behavioral levels of analysis. The field of LTP is also very controversial, so that there is only a surprisingly small number of completely accepted findings.

It is the lack of agreement about basic facts and parameters that accounts in some measure for the limited attention that Koch devotes to LTP in his account of computational mechanisms in the brain. He prefers to base his conjectures about basic neural mechanisms of computation on mechanisms for which there is substantial agreement about basic facts.

One might add to Koch's characterization of the current state of LTP research that a major reason why neuroscientists think that (some form of) LTP must be the mechanism of learning and memory is that this hypothesis is consistent with what they take to be established psychological fact, namely that learning and memory are associative processes. This extrinsic support from psychology and philosophy, rather than the murky and controversial evidence from neurobiological experiment, accounts in no small measure for the fact that the synaptic plasticity hypothesis is neurobiologists' "most plausible" hypothesis about memory. In short, neuroscientists

justify the study of LTP by citing associative learning as an established psychological fact, while psychologists justify associative learning models by citing "associative" LTP as an established neurobiological fact. Both groups and the progress of cognitive neuroscience would benefit from a fuller appreciation of the shakiness of the experimental foundations for the extrinsic hypotheses to which each field appeals as established fact in order to buttress weak support from the experimental evidence intrinsic to their own field. Neuroscientists would benefit from an appreciation of how weak the behavioral evidence for associative learning is (Gallistel, 1990, 1995, 1999, 2008; Gallistel & Gibbon, 2000), while psychologists would benefit from a fuller appreciation of the controversy and lack of agreement about basic facts that pervade the neurobiological literature on LTP.

Nonetheless, because long-term synaptic plasticity is the only widely accepted hypothesis when it comes to current ideas about the neurobiology of memory, it is instructive to consider how one might use this to make a neural machine that reacts to the current input in a way that depends on the sequence of previous inputs. Can we use this mechanism to make a neural machine with the same functionality as our finite-state marble machine? In considering this question, we assume that the interval that elapses between the previous inputs in a sequence and the current input may be of indefinite duration. Thus, the short-time-scale, rapidly decaying processes that Koch describes are not relevant. The relevant physical changes wrought by previous inputs must be like the changes that falling marbles produce when they change the tilt of a rocker, they must endure indefinitely. Among the many facts about which there is controversy is how long LTP in fact endures, but we will ignore this; we will assume a mechanism that produces a change in synaptic conductance that lasts indefinitely. Moreover, we will assume that this change can be bidirectional: synaptic conductance may be either increased or decreased. The latter phenomenon, an enduring decrease in synaptic conductance, is called *long-term depotentiation* (LTD).

We will, however, insist that the mechanism have one of the properties of LTP and LTD about which there is fairly general agreement (Koch, 1999, p. 318): "*induction requires (nearly) simultaneous presynaptic neurotransmitter release and postsynaptic depolarization.*" The reader may be puzzled about how changes in opposite directions may have the same causal antecedents. How can it be that transmitter release simultaneous with postsynaptic depolarization causes *both* LTP and LTD, both an increase in conductance and a decrease in conductance? There is no clear answer to this question. The usual answer is that whether one gets LTP or LTD depends on the degree to which the internal calcium ion concentration at the postsynaptic membrane is raised above its resting level by the postsynaptic depolarization. If it is only moderately elevated, LTD (a decrease in conductance) is said to result; if the elevation is greater, LTP is the result (an increase in conductance). However, there is much controversy about the specifics. All one can say with any confidence at present is that under some conditions one sees LTP, while under others one sees LTD.

The hedge about the simultaneity of pre- and postsynaptic activity conveyed by the parenthetical "nearly" is measured in tens of milliseconds: the presynaptic release of transmitter substance and strong postsynaptic depolarization must be separated

in time by no more than a few tens of milliseconds. In one often-cited report, deviations from absolute synchrony between presynaptic action potentials and postsynaptic action potentials of +10 milliseconds vs. −10 milliseconds determined whether synaptic conductance was increased (LTP) or decreased (LTD, Markram, Lübke, Frotscher, & Sakmann, 1997). Moreover, there was no effect on synaptic conductance unless these small differences in the timing of pre- and postsynaptic action potentials occurred while both neurons were firing 10 times per second or faster. (This is well above typical basal firing rates, which have been estimated to be less than one action potential per second.) Nor was there any effect when the strong postsynaptic depolarization was offset from the burst of presynaptic action potentials by more than 100 milliseconds in either direction. The very complexity of this result – the complexity of the conditions for inducing either LTP or LTD in the preparation used in this experiment – suggests something about how poorly both phenomena are currently understood.

Setting these qualifications aside, we ask whether this mechanism can be used to solve simple computational problems in which memory plays a critical role. We will find that it does not get us where we need to go. It does not enable us to construct even a finite-state automaton capable of looking back one step in a sequence of inputs. Thus, it does not give us even the computing power of a finite-state machine, let alone the power of a Turing machine. What this suggests is that the most plausible neurobiological hypothesis about the mechanism of memory is seriously inadequate from a computational perspective.

The instructive power of the look-back-one computation comes from its extreme simplicity. We want to make a neural machine with the same functionality as our finite-state marble machine whose output depended not just on the current input but also on the immediately preceding input. In the marble machine, the inputs were a marble falling in one of two input channels. In the neural machine, the inputs are an action potential conducted in one of two axons. In the marble machine, the outputs were a marble falling in one of four channels, depending jointly on which channel the marble came in on and which channel the previous marble came in on. In the neural machine, the outputs are an action potential conducted in one of four outgoing axons, depending jointly on which axon the incoming action potential arrives on and on which axon the preceding input action potential arrived on. If we can create a neural circuit with this functionality, we can combine copies of it to make a machine that can look back several steps, albeit at a cost in physical resources that increases exponentially with the number of look-back steps.

The key to creating a marble machine capable of looking back one event in the input sequence was a rocker-gate element with three properties: (1) It had two thermodynamically stable (hence, enduring) states; (2) the state it was in determined which of two output channels an input marble would be directed to; (3) it did or did not transition to its other state depending on whether the input did not or did match its current state. In attempting to create a functionally equivalent neural machine, we assume for the sake of argument that Hebbian synapses exist. We assume that they have (at least) two states that endure indefinitely, a conducting state and a non-conducting state. We assume that under some conditions, presynaptic transmitter release (by an arriving action potential) together with postsynaptic

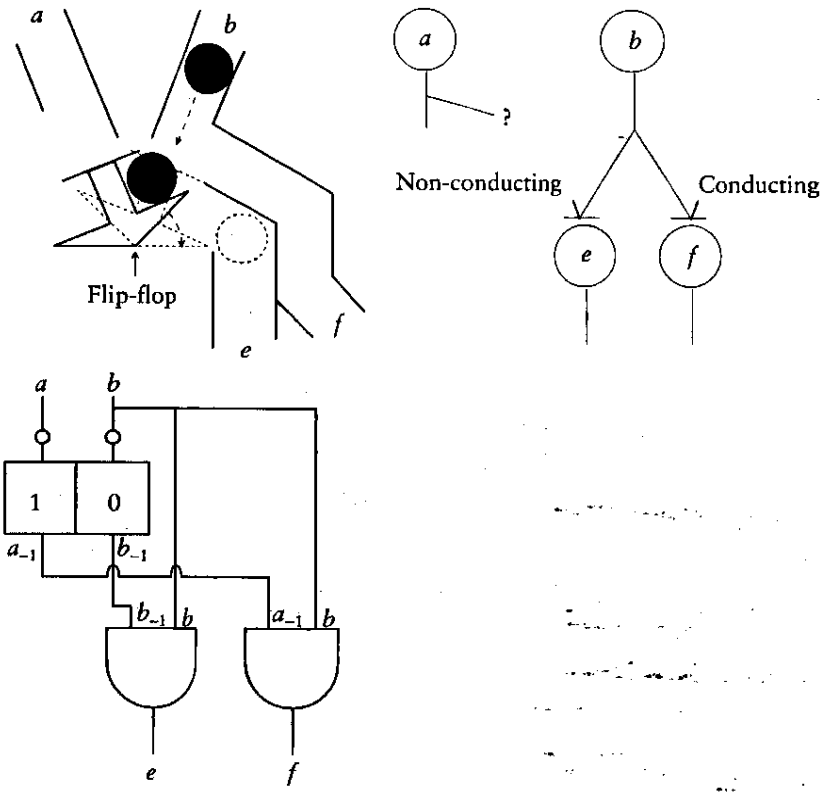

Figure 10.1 Attempting to construct a look-back element using Hebbian synapses as the memory elements. Only the right half of the symmetrical circuits are shown. On the left, as an aid to thought, we reproduce the structure of the marble machine (top) and the structure of a solid-state circuit (bottom), both having the requisite functionality. The conducting/non-conducting state of a synapse is indicated by the position of the "switch": when the switch is thrown right, the synapse is in its conducting state; when thrown left, it is in its non-conducting state. The only way to throw the switch is to have the postsynaptic neuron become depolarized at (nearly, i.e., +/– 10 ms) the same time as transmitter is released from the presynaptic neuron by the arrival of an action potential. By stipulation, action potentials cannot arrive simultaneously on *a* and *b*, because the challenge is to make an action potential arriving on *b* fire *f* if the preceding action potential also arrived on *b*, and *e* if the preceding arrived on a. Thus, one of the requirements is that an action potential on *a* reverse the states of the two synaptic switches.

depolarization (however realized) will put a synapse into its conducting state, while under slightly different conditions, the same confluence of pre- and postsynaptic events will put it into its non-conducting state.

To help us think about this problem, we make the sketch at the top right of Figure 10.1. It shows two input neurons (*a* and *b*) and two of the four output

neurons (*e* and *f*), namely, the two that are to be fired by an action potential arriving on presynaptic neuron *a*. We only show two of the four output neurons, because the problem is symmetrical. If we can get the two inputs to affect those two outputs in the required way, then, by symmetry, we can achieve the same result for the other two outputs. By stipulation, synapses between presynaptic neurons *a* and *b* and output neurons *e* and *f* can be placed in either a conducting or non-conducting state. In other words, they are single-pole switches. We indicate their state (conducting or non-conducting) by a switch thrown either to the right (conducting) or the left (non-conducting).

In its initial configuration, the *bf* synapse is in the conducting state and the *be* synapse in the non-conducting state. Thus, an action potential in *b* will fire *f*, which is what we want. Suppose now that the next action potential arrives on axon *a*. It must reverse the states of the two synapses, setting the *be* synapse to its conducting state and the *bf* to its non-conducting state. How are we to make this happen? These are Hebbian synapses; they only change state when their presynaptic and postsynaptic sides are simultaneously activated. But the presynaptic sides of the *be* and *bf* synapses are not activated, because the action potential has arrived on *a* not *b*. The previous action potential arrived on *b*, but that may have been an indefinitely long while ago. We can wire our system so that an action potential in *a* depolarizes output neurons *e* and *f*, but that establishes only one of the two things that must happen simultaneously in order for the conductance to change. Thus, there does not appear to be a way to achieve the necessary changes in conducting state when the input arrives on *a*. The Hebbian induction conditions appear to be inimical to our objective.

Suppose that the conducting states of the synapses have *somehow* been reversed, indicating that the preceding action potential arrived on *a*. Suppose further that the next action potential arrives on *b*. We can assume that this fires *e*, which is what we want. It is a bit of a stretch, but we can perhaps also assume that although the postsynaptic depolarization of *e* is sufficient to fire it, this is nonetheless the intermediate level of depolarization that, when accompanied by a presynaptic action potential, causes LTD rather than LTP. In other words, we assume that the *be* synapse is switched into its non-conducting state. This is in contrast to the assumption we made earlier concerning the *bf* synapse. There, we assumed that a presynaptic input sufficient to fire the postsynaptic neuron left the synapse in its conducting state. The murky facts about just what level of depolarization is required to get LTD as opposed to LTP perhaps allows us this latitude. But how are we to get the *bf* synapse to reverse its state? We have an action potential arriving at the presynaptic ending, but there is no simultaneous depolarization of the postsynaptic membrane, because the *bf* synapse is in its non-conducting state. There is no signal in *a*, because we are considering the case where the signal arrives on *b*. Again, the induction conditions on a Hebbian synapse are inimical to our objective.

We conjecture that the problem is unsolvable within the constraints we have stipulated. Our conjecture is not a proof. If a reader finds a solution, we will be genuinely grateful to learn of it, provided that it in fact honors the constraints we have stipulated. As an additional incentive to readers who like to measure their ingenuity against that of acknowledged experts, we add that we posed this problem

to a number of leading figures in the neural network modeling community. None could suggest an answer within the constraints we have stipulated.

One remarked that the problem required a "stack," that is, a read/write memory. We agree. Indeed, that comment brings us to the most popular mechanism for dealing with this problem within the neural network modeling community, which is to use reverberating activity as a read/write memory. The same expert who commented that it required a stack went on to suggest this approach.

Recurrent Loops in Which Activity Reverberates

A paper by Elman (1990) popularized the use of recurrent loops in the extraction of temporal structure. Elman is a linguist. Language unfolds in time. Any brain capable of understanding any human language must be able to extract temporal structure. Elman opened his paper by calling attention to the problem we are discussing: "[In the context of neural network theorizing] one of the most elementary facts about much of human activity – that it has temporal extent – is sometimes ignored and is often problematic" (p. 179). He then discusses solutions considered in previous modeling attempts and why they were unsatisfactory, which is basically because they begged the question by positing an input shift register that converted sequential inputs into simultaneous inputs.

Finally, he proposed the solution shown in Figure 10.2, which is taken from his paper. His model has become extremely influential. What it has that previous nets generally lacked is the "context units." As Elman explains in his caption, "activations are copied from hidden layer to context layer on a one-for-one basis, with fixed weight of 1.0." In other words, the signal values in the hidden units are written to the context units, so that the context units contain a copy of the previous signal values. The context units in turn project back to the hidden units. However, whereas one hidden unit projects to one and only one context unit, each context unit projects back to every hidden unit. Thus, the context units make the previous activity of any hidden unit available to all the other hidden units. Moreover, the back projections from the context units to the hidden units are "trainable," which is to say that the hidden unit can make of them whatever it likes (adjust the weight it gives to each different one).

Elman's recurrent loop from the hidden units to the context units and then from the context units back to the hidden units is a read/write memory. The untrainable one–one projection from the hidden units to the context units implements the write operation, while the one to all trainable projection back to the hidden units allows random access reading of any context unit by any hidden unit. As Elman writes, "The recurrent connections allow the network's hidden units to see its own previous output, so that the subsequent behavior can be shaped by previous responses. These recurrent connections are what give the network memory" (p. 182), and "the context units remember the previous internal state. Thus, the hidden units have the task of mapping both an external input and also the previous internal state to some desired output" (p. 184). This last is, of course, what we require in a machine that can look back: access to both the current input and previous input.

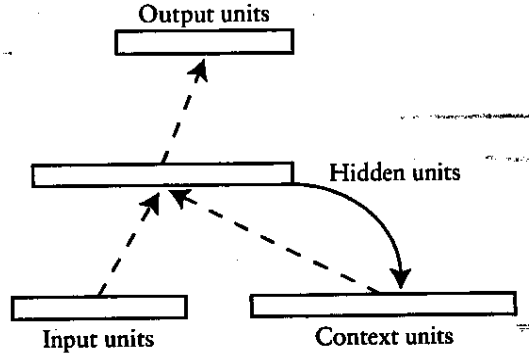

Figure 10.2 A simple recurrent network in which activations are copied from hidden layer to context layer on a one-for-one basis, with fixed weight of 1.0. Dotted lines represent trainable connections. (Reproduced by permission from Elman, 1990, p. 183.)

The essential thing to realize about Elman's solution, which has become the most common solution in the neural network literature to the problem of responding differentially to different temporal structure in the input, is that it does *not* use Hebbian synapses as memory elements. (Indeed, in Elman's model even the trainable synapses are trained by back-propagation, which is an omniscient god outside the machine.) Elman's proposal assumes instead the existence of elements to which activity can be "copied" (that is, written) and which hold that activity until the next time step, making it available to processing by the hidden units (that is, it is held in readable form). In Chapter 14, we will study in some detail the use of this same idea in a model of dead reckoning.

For the moment, we note only that this reverberating activity model of memory is implausible as a model for memory on a time scale longer than a fraction of a minute. That the model works when implemented using the noise-free read/write memory elements of a conventional computer is no surprise. On that kind of machine, the assumption that one can copy "activity" (that is, a floating point number) to memory with gain of 1 (so that the number in memory is exactly the same as the number that was copied), which memory then makes the number available to any and all other computations, is entirely reasonable. That is the essential function of memory, as a computer scientist understands it. But how should we imagine that this can be achieved in the nervous system whose biophysical reality Koch is concerned to depict? Elman does not say. But among those who have followed his lead, the assumption is that a signal somehow encoding a real number (that is, a graded quantity) is copied into a reverberating loop, where it travels around and around, neither gaining nor losing in value.

One certainly can trace loops within the connectivity of the nervous system. The hypothesis that reverberating activity in such loops can serve a critical short-term memory function goes back at least to Lorente de No (1932). It played a significant role in the earliest attempts to think about neural activity in computational terms (McCulloch & Pitts, 1943). Moreover, there is experimental evidence for it in at

least one model system, the oculomotor integrator of the goldfish (Aksay, Baker, Seung, & Tank, 2000; Major et al., 2004). However, even in the goldfish oculomotor system, the activity in a loop typically declines by more than half in less than two minutes. In that loop, the rate of firing specifies the position of the eye in its socket. As the activity in that loop declines, the position of the eye drifts back to its null position. This is not a problem for the oculomotor system because it works on a time scale of seconds. In a few seconds, the drift due to the decay in the loop is negligible. If, however, activity in such a loop were to represent, say, the goldfish's distance from its nest or from its favorite hiding place, the goldfish would not – so far as its brain was concerned – need to do anything in order to get home. No matter how far it was from home, if it simply held still for a few hundred seconds, its brain would represent it as having reduced its distance to home to zero, because the circulating signal representing that distance would have decayed to the basal firing rate of the circuit. Thus, the first and perhaps foremost objection to the idea that reverberating activity can serve as a memory over indefinite intervals is that it would be difficult to prevent its becoming degraded over intervals measured in minutes at most.

The problem is that a reverberating signal must cross synapses 100 or more times per second. Synapses are thought to be inherently noisy, because the process of releasing transmitter from the presynaptic ending is stochastic (the number of quanta released by an action potential varies), and because the process of binding to postsynaptic receptor molecules is also stochastic (the fraction of transmitter released that succeeds in binding to a postsynaptic receptor varies). To preserve a reverberating signal for any length of time, there can be very little noise at the synapses that it must traverse over and over again.

A further problem is the coding: Is the magnitude of the stored value coded by the number of action potentials that pass a given point in the circuit in a given unit of time? Or in some other way? If the first way, must the action potentials be evenly spaced around the loop, so that the instantaneous firing rate (the reciprocal of the interval between any two action potentials) is constant? How could the even spacing be achieved? Or, are action potentials allowed to bunch up within the loop, so that they pass any given point in a burst? In the latter case, how does the process that reads the reverberating activity know whether it is reading within the burst or between bursts? And, if it has a way of recognizing the onset of a burst, does it know how long it must wait for the burst to come by again?

Another problem is the energetic costliness of this approach to information preservation. Signaling accounts for more than half the brain's energy demand and the adult human brain's demand for energy is 20 percent of the body's total demand (Laughlin, 2004). Storing large amounts of information for long periods in reverberating loops would tie up much of the brain's circuitry, be extravagantly wasteful of energy, and create a serious cooling problem. Neural signal transmission is not immune to the laws of thermodynamics. A substantial fraction of the energy released to do the work of signal transmission is lost as heat. Dissipating that heat would be no small problem.

These are among the many reasons why we believe that no biophysically informed neuroscientist believes that storing information in reverberatory loops can

be the brain's solution to preserving large amounts of information for indefinite amounts of time. Thus, neither of the two mechanisms that have traditionally been imagined to implement memory – neither Hebbian synapses, nor reverberating activity – is suited to serve the function of carrying information forward in time in a computationally accessible form. And that brings us back to the point with which we started this chapter, the fact that at this time we do not know the answers to many of the most basic questions about how computations are implemented in neural tissue. A mechanism for carrying information forward in time is indispensable in the fabrication of an effective computing machine. Contemporary neurobiology has yet to glimpse a mechanism at all well suited to the physical realization of this function.