9

Strong Artificial Intelligence

David Chalmers recently gave a talk at NYU on whether Language Models, like GPT-3, can be conscious. You can watch it here: www.shorturl.at/ceAKN

1. Machine Consciousness

Could a machine be conscious? Could an appropriately programmed computer truly possess a mind? These questions have been the subject of an enormous amount of debate over the last few decades. The field of *artificial intelligence* (or *AI*) is devoted in large part to the goal of reproducing mentality in computational machines.So far progress has been limited, but supporters argue that we have every reason to believe that eventually computers will truly have minds. At the same time, opponents argue that computers are limited in a way that human beings are not, so that it is out of the question for a conscious mind to arise merely in virtue of computation.

Objections to artificial intelligence typically take one of two forms. First, there are *external* objections, which try to establish that computational systems could never even *behave* like cognitive systems. According to these objections, there are certain functional capacities that humans have that no computer could ever have. For example, sometimes it is argued that because these systems follow rules, they could not exhibit the creative or flexible behavior that humans exhibit (e.g., Dreyfus 1972). Others have argued that computers could never duplicate human mathematical insight, as computational systems are limited by Gödel's theorem in a way that humans are not (Lucas 1961; Penrose 1989).

External objections have been difficult to carry through, given the success of computational simulation of physical processes in general. In particular, it seems that we have good reason to believe that the laws of physics are computable, so that we at least ought to be able to *simulate* human behavior computationally. Sometimes this is disputed, by arguing for a noncomputable

element in physical laws (as Penrose does), or by arguing for nonphysical causation (as Lucas does), but it is clear that those putting forward these objections are fighting an uphill battle.

More prevalent have been what I call *internal* objections. These objections concede at least for the sake of argument that computers might simulate human behavior, but argue that they would lack minds all the same. In particular, it is suggested that they would have no inner life: no conscious experience, no true understanding. At best, a computer might provide a *simulation* of mentality, not a replication. The best known objection in this class is John Searle's "Chinese room" argument (Searle 1980). According to these objections, computational systems would at best have the hollow shell of a mind: they would be silicon versions of a zombie.

Those who take a nonreductive view of conscious experience have often been attracted to internal objections to artificial intelligence, with many arguing that no mere computer could be conscious. Indeed, those who have been impressed by the problem of consciousness have sometimes characterized the problem by pointing to consciousness as the feature that we have but that any computer would lack! Many have found it hard to believe that an artificial, nonbiological system could be the sort of thing that could give rise to conscious experience.

A nonreductive view of consciousness does not automatically lead to a pessimistic view of AI, however. The two issues are quite separate. The first concerns the *strength* of the connection between physical systems and consciousness: Is consciousness constituted by physical processes, or does it merely arise from physical processes? The second concerns the *shape* of the connection: Just *which* physical systems give rise to consciousness? Certainly it is not *obvious* that executing the right sort of computation should give rise to consciousness; but it is not obvious that neural processes in a brain should give rise to consciousness, either. On the face of it, there is no clear reason why computers should be any worse off than brains in this regard. Given that we have accepted the surprising fact that brains give rise to consciousness, it would not be a *further* sort of surprise to find that computation might give rise to consciousness. So the mere embrace of a nonreductive view of consciousness ought to leave the matter open.

In this chapter, I will take things further and argue that the ambitions of artificial intelligence are reasonable (Figure 9.1). In particular, I will argue for the view that Searle calls *strong artificial intelligence*: that there is a nonempty class of computations such that the implementation of any computation in that class is sufficient for a mind, and in particular, is sufficient for the existence of conscious experience. This sufficiency holds only with natural necessity, of course: it is *logically* possible that any computation might take place in the absence of consciousness. But the same goes for brains, as we

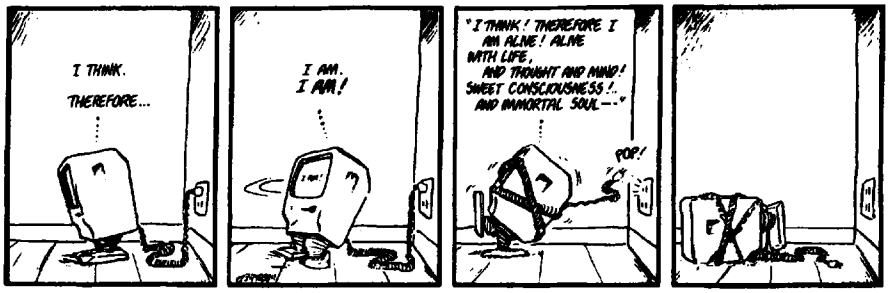

Figure 9.1. Bloom County on strong AI. (© 1985, Washington Post Writers Group. Reprinted with permission)

have seen. In evaluating the prospects of machine consciousness in the actual world, it is natural possibility and necessity we are concerned with.

(Lest this conclusion be thought a triviality, given the panpsychist suggestions in the last chapter, I note that nothing in this chapter rests on those considerations. Indeed, I will argue not just that implementing the right computation suffices for consciousness, but that implementing the right computation suffices for rich conscious experience like our own.)

I have already done most of the work required for this defense of strong AI, in arguing for the principle of organizational invariance in Chapter 7. If that argument is correct, it establishes that any system with the right sort of functional organization is conscious, no matter what it is made out of. So we already know that being made of silicon, say, is no bar to the possession of consciousness. What remains to be done is to clarify the link between computation and functional organization, in order to establish that implementing an appropriate computation is sufficient to ensure the presence of the relevant functional organization. Once this is done, strong AI falls out as a consequence. I will also answer a number of objections that have been put forward against the strong AI enterprise.

2. On Implementing a Computation

In its standard form, the theory of computation deals wholly with *abstract* objects: Turing machines, Pascal programs, finite-state automata, and so on. These are mathematical entities inhabiting mathematical space. Cognitive systems in the real world, on the other hand, are *concrete* objects, physically embodied and interacting causally with other objects in the physical world. But often we want to use the theory of computation to draw conclusions about concrete objects in the real world. To do this, we need a bridge between the abstract and concrete domains.[1]

This bridge is the notion of *implementation*: the relation between abstract computational objects—"computations" for short—and physical systems that holds when a physical system "realizes" a computation, or when a computation "describes" a physical system. Computations are often implemented on synthetic, silicon-based computers, but they can be implemented in other ways. Natural systems such as the human brain are often said to implement computations, for example. Computational descriptions are used to make sense of physical systems in all sorts of domains. Whenever this happens, a notion of implementation is implicitly or explicitly doing the work.

The notion of implementation is rarely analyzed in detail; it is usually simply taken for granted. But to defend strong AI, we need a detailed account of it. The strong AI thesis is cast in terms of computation, telling us that implementation of the appropriate computation suffices for consciousness. To evaluate this claim, we need to know just what it is for a physical system to implement a computation. Once we know this, we can combine it with our earlier analysis of psychophysical laws to determine whether the conclusion might follow.

Some have argued that no useful account of implementation can be given. In particular, Searle (1990b) has argued that implementation is not an objective matter, but instead is "observer-relative": any system can be seen to implement any computation if interpreted appropriately. Searle holds, for example, that his wall can be seen to implement the Wordstar word processing program. If this were so, it would be hard to see how computational notions could play any foundational role in a theory that ultimately deals with concrete systems. As for strong AI, it would either be emptied of content or would imply a strong form of panpsychism. But I think this sort of pessimism is misplaced: an objective account of implementation can straightforwardly be given. In this section I will outline such an account. (The account is a little technical, but the rest of the chapter should make sense even if the details here are skimmed.)

Any account of what it is for a computation to be implemented will depend on the class of computations in question. There are many different computational formalisms, with correspondingly different classes of computations: Turing machines, finite-state automata, Pascal programs, connectionist networks, cellular automata, and so on. In principle, we need an account of implementation for each of these formalisms. I will give an account of implementation for a single formalism, that of *combinatorial-state automata*. This class of computations is sufficiently general that the associated account of implementation can be easily extended to apply to other classes.

A combinatorial-state automaton is a more sophisticated cousin of a *finite-state* automaton. A finite-state automaton (FSA) is specified by giving a finite set of inputs, a finite set of internal states, and a finite set of outputs, and

by giving an associated set of *state transition* relations. An internal state of an FSA is a simple element S_i without any internal structure; the same goes for inputs and outputs. The state transition relations specify, for every possible pair of inputs and internal states, a new internal state and an output. If the initial state of an FSA is given, these state transition relations specify how it will evolve over time and what outputs it will produce, depending on what inputs are received. The computational structure of an FSA consists in this relatively simple set of state transition relations among a set of unstructured states.

Finite-state automata are inadequate to represent the structure of most computations that are relevant in practice, as the states and state transition relations in these computations generally have complex internal structure. No FSA description can capture all the structure present in a Pascal program, for example, or a Turing machine, or a cellular automaton. It is therefore more useful to concentrate on a class of automata that have structured internal states.

Combinatorial-state automata (CSAs) are just like FSAs, except that their internal states are structured. A state of a CSA is a *vector* $[S^1, S^2, \ldots, S^n]$. This vector can be either finite or infinite, but I will focus on the finite case. The elements of this vector can be thought of as the *components* of the internal state; they correspond to the cells in a cellular automaton or the tape-squares and head-state in a Turing machine. Each element S^i, can take on a finite number of values S^i_j, where S^i_j is the jth possible value of the ith element. These values can be thought of as "substates" of the overall state. Inputs and outputs have a similar sort of complex structure: an input is a vector $[I^1, \ldots, I^k]$, and an output is a vector $[O^1, \ldots, O^m]$.

A CSA is determined by specifying the set of internal state vectors and input and output vectors, and by specifying a set of *state transition rules* that determine how the state of the CSA evolves with time. For each element of the internal-state vector, a state transition rule determines how its new value depends on old values of the input and internal state vectors. For each element of the output vector, a state transition rule determines how its new value depends on old values of the internal state vector. Every finite CSA can be represented as an FSA with equal computational power, but the FSA description will sacrifice most of the structure that is crucial to a CSA. That structure is central in using CSAs to capture the organization that underlies a mind.

We are now in a position to give an account of implementation. Computations such as CSAs are abstract objects, with a *formal structure* determined by their states and state transition relations. Physical systems are concrete objects, with a *causal structure* determined by their internal states and the causal relations between the states. Informally, we say that a physical system *implements* a computation when the causal structure of the system mirrors

the formal structure of the computation. That is, the system implements the computation if there is a way of mapping states of the system onto states of the computation so that physical states that are causally related map onto formal states that are correspondingly formally related.

This intuitive idea can be straightforwardly applied to yield an account of implementation for CSAs. A physical system implements a CSA if there is a decomposition of internal states of the system into substates, a decomposition of the system's inputs and outputs into input and output substates, and a mapping from substates of the system onto substates of the CSA, such that the causal state transition relations between physical states, inputs, and outputs reflect the formal state transition relations between the corresponding formal states, inputs, and outputs.

The formal criterion for implementing a CSA is as follows:

> A physical system P implements a CSA M if there is a decomposition of internal states of P into components $[s^1, \ldots, s^n]$, and a mapping f from the substates s^j into corresponding substates S^j of M, along with similar decompositions and mappings for inputs and outputs, such that for every state transition rule $([I^1, \ldots, I^k], [S^1, \ldots, S^n]) \rightarrow ([S'^1, \ldots, S'^n], [O^1, \ldots, O^l])$ of M: if P is in internal state $[s^1, \ldots, s^n]$ and receives input $[i^1, \ldots, i^n]$, which map to formal state and input $[S^1, \ldots, S^n]$ and $[I^1, \ldots, I^k]$ respectively, this reliably causes it to enter an internal state and produce an output that map to $[S'^1, \ldots, S'^n]$ and $[O^1, \ldots, O^l]$ respectively.

We may stipulate that in a decomposition of the state of a physical system into a vector of substates, the value of each element of the vector must supervene on a separate region of the physical system, to ensure that the causal organization relates distinct components of the system. Otherwise, it is not clear that the detailed causal structure is really present within the physical system. There is room to tinker with this and with other details in the definition above. The notion of implementation is not written in stone, and it might be tightened or loosened for various purposes. But this gives the basic shape that will be shared by any account of implementation.

It may seem that CSAs are not much of an advance on FSAs. After all, for any finite CSA, we can find a corresponding FSA with the same input–output behavior. But there are some crucial differences. First and foremost, the *implementation* conditions on a CSA are much more constrained than those of the corresponding FSA. An implementation of a CSA is required to consist in a complex causal interaction among a number of separate parts; a CSA description can therefore capture the causal organization of a system to a much finer grain. Second, CSAs provide a unified account of the implementation conditions for both finite and infinite machines. And third, a CSA can directly reflect the complex formal organization of computa-

tional objects such as Turing machines and cellular automata. In the corresponding FSA, much of this structure would be lost.

Indeed, we can use this definition of implementation to straightforwardly provide implementation criteria for other sorts of computations. To specify what it takes to implement a Turing machine, for example, we need merely redescribe a Turing machine as a CSA and apply the definition above. To do this, we describe the state of the Turing machine as a giant vector. One element of the vector represents the state of the machine head, and there is an element for each square of the tape, representing the symbol in the square and also indicating whether or not the machine head occupies that square. The state transition rules between the vectors are those derived naturally from the mechanisms specifying the behavior of the machine head and the tape. Of course, the vectors here are infinite, but the implementation conditions in the infinite case are a straightforward extension of those in the finite case. Given this translation from the Turing machine formalism to the CSA formalism, we can say that a Turing machine is implemented whenever the corresponding CSA is implemented. We can give similar translations of computations in other formalisms, such as cellular automata or Pascal programs, yielding implementation conditions for computations in each of these classes.

This yields a perfectly objective criterion for implementing a computation. Implementation of a computation does not collapse into vacuity in the way that Searle suggests. It is true that *some* computations will be implemented by every system. For example, the single-element, single-state CSA will be implemented by every system, and a two-state CSA will be implemented almost as widely. It is also true that most systems will implement more than one computation, depending on how we carve up that system's states. There is nothing surprising about this: it is only to be expected that my workstation implements a number of computations, as does my brain.

What is crucial is that there is no reason to believe that *every* CSA will be implemented by *every* system. For any given complex CSA, very few physical systems will have the causal organization required to implement it. If we take a CSA whose state vectors have one thousand elements, with ten possibilities for each element, then arguments along the lines of those presented in Chapter 7 suggest that the chance of an arbitrary set of physical states having the requisite causal relations is something less than one in $(10^{1000})^{10^{1000}}$ (actually much less than this, because of the requirement that the transition relations be reliable).[2]

What of Searle's claim that computational descriptions are "observer-relative," then? It is true that there is a limited degree of observer relativity: any given physical system will implement a number of computations, and which one of these an observer chooses to focus on will depend on her purposes. But this is not threatening to AI or computational cognitive sci-

ence. It remains the case that for any given computation, there is a fact of the matter about whether or not a given system implements it, and there will be only a limited class of systems that qualify as implementations. For computational accounts to have metaphysical and explanatory bite, this is all that the fields require.

To say that a physical system implements a given complex computation P is to say something very substantial about the causal structure of that system, something that may be quite useful in providing cognitive explanations and perhaps in understanding the basis of consciousness. Only systems with a very specific sort of causal organization will have a hope of satisfying the strong constraints of implementation. So there is no danger of vacuity, and there is room to hope that the notion of computation can provide a substantial foundation for the analysis of cognitive systems.

3. In Defense of Strong AI

What it takes to implement a CSA is strikingly similar to what it takes to realize a functional organization. Recall that a functional organization is determined by specifying a number of abstract components, a number of states for each component, and a system of dependency relations indicating how the states of each component depend on previous states and on inputs, and how outputs depend on previous states. The notion of a CSA is effectively a direct formalization of this notion.

Indeed, given any functional organization of the sort described in Chapter 7, it can be straightforwardly abstracted into a CSA. We need only stipulate that the CSA's state vectors have an element for each component of the organization, and that the formal state transitions between the CSA states correspond to the causal dependency relations between components. To realize the functional organization comes to almost exactly the same thing as implementing the corresponding CSA. There are a few small differences, such as different treatments of inputs and outputs, but these are not significant.

The account of implementation that I have given thus makes clear the link between causal and computational organization. This way, we can see that when computational descriptions are applied to physical systems, they effectively provide a formal description of the systems' causal organization. The language of computation provides a perfect language in which this sort of abstract causal organization can be specified. Indeed, it can be argued that this is precisely why computational notions have had such wide application throughout cognitive science. What is most relevant to the explanation of the behavior of a complex cognitive system is the abstract causal organization of the system, and computational formalisms provide an ideal

framework within which this sort of organization can be described and analyzed.[3]

This link makes the defense of strong artificial intelligence straightforward. I have already argued for the principle of organizational invariance, which tells us that for any system with conscious experiences, a system with the same fine-grained functional organization will have qualitatively identical conscious experiences. But we know that any given functional organization can be abstracted into a CSA that is implemented whenever the organization is realized. It follows that for a given conscious system *M*, its fine-grained functional organization can be abstracted into a CSA *M*, such that any system that implements *M* will realize the same functional organization, and will therefore have conscious experiences qualitatively indistinguishable from those of the original system. This establishes the thesis of strong artificial intelligence.

For example, we might abstract a neural description of the brain into a CSA, with an element of the state vector for each neuron and with substates for each element reflecting the relevant range of each neuron's states. The state transition rules of the CSA reflect the way in which the state of each neuron depends on the state of other neurons, and the way in which neural states are related to inputs and output. If nonneural components of the brain are relevant, we can include components for those, too. Any physical system that implements this CSA will have a fine-grained functional organization that duplicates the neuron-level functional organization of the brain. By the invariance principle, this system will have experiences indistinguishable from those associated with the brain.

It is easy to think of a computer as simply an input–output device, with nothing in between except for some formal mathematical manipulations. This way of looking at things, however, leaves out the key fact that there are rich causal dynamics inside a computer, just as there are in the brain. Indeed, in an ordinary computer that implements a neuron-by-neuron simulation of my brain, there will be real causation going on between voltages in various circuits, precisely mirroring patterns of causation between the neurons. For each neuron, there will be a memory location that represents the neuron, and each of these locations will be physically realized in a voltage at some physical location. It is the causal patterns among these circuits, just as it is the causal patterns among the neurons in the brain, that are responsible for any conscious experience that arises.

We can also defend the strong AI thesis directly, using the fading qualia and dancing qualia arguments. Given any two implementations of a CSA, there will be a spectrum of cases between them, in which physical components of the implementations are replaced one at a time while the pattern of their causal interaction with the rest of the system is preserved. If one of the systems is conscious, and if the CSA abstracts its fine-grained functional

organization, then the arguments in question imply that the other system must be conscious and that it must have indistinguishable conscious experiences. If the other system were not conscious, there would be an intermediate system with fading qualia. If the other system were not conscious or had different conscious experiences, then we could construct an intermediate system with dancing qualia. These consequences are implausible, for the reasons outlined in Chapter 7. Given that qualia cannot fade or dance in this way, it follows that the second of the original systems has experiences indistinguishable from the first, and that the strong AI thesis holds.

There is a small caveat. The argument assumes that the brain's organization can be abstracted into a CSA description. This requires only that the relevant organization can be described in terms of a finite number of parts each having a finite number of relevant states. Nevertheless, some might dispute this. For example, perhaps an infinite number of states are needed for each neuron, to capture the vital role of continuous processing. And some might claim that the transitions between these infinite states might be uncomputable. I will discuss this sort of objection later; for now, I am happy to embrace the conclusion that *if* cognitive dynamics are computable, then the right sort of computational organization will give rise to consciousness. That is, I am more concerned with internal than external objections here. All the same, I will address some external objections later in the chapter.

4. The Chinese Room and Other Objections

Of course, opponents of strong AI have sometimes put forward concrete arguments against the position. The best known of these are due to John Searle, in his 1980 paper, "Minds, Brains, and Programs," and elsewhere. Here I will use the framework I have outlined to answer these arguments.

The Chinese room

In a celebrated argument against strong AI, Searle (1980) argues that any program can be implemented without giving rise to a mind. He does this by exhibiting what he takes to be a *counterexample* to the strong AI claim: the Chinese room, inside which a person manipulating symbols simulates someone who understands Chinese. The Chinese room is intended to provide an example, for any given program, of a system that implements that program but that lacks the relevant conscious experience.

In the original version, Searle directs the argument against machine *intentionality* rather than machine consciousness, arguing that it is "understanding" that the Chinese room lacks. All the same, it is fairly clear that consciousness is at the root of the matter. What the core of the argument establishes directly, if it succeeds, is that the Chinese room system lacks

conscious states, such as the conscious experience of understanding Chinese. On Searle's view, intentionality requires consciousness, so this is enough to see that the room lacks intentionality also. Others deny this, however. In any case we can factor out the issue about the connection between consciousness and intentionality, and cast the issue solely in terms of consciousness. The issues may be somewhat clearer this way.

(That is, we can separate Searle's conclusions into two parts: (1) no program suffices for consciousness; and (2) no program suffices for intentionality. Searle believes that (1) implies (2), but others deny this. Things are clearest if the argument about strong AI is taken to focus on (1): all parties will accept that if (1) is true, then the most interesting form of strong AI is doomed, and even Searle would accept that refuting (1) would show that the Chinese room argument fails. The link between consciousness and intentionality can then be set aside as a separate issue, not crucial to the argument against AI.

This way one avoids the situation in which opponents argue against (2) without bothering to argue against (1). For example, replies that focus on the connection between the Chinese room and its environment [Fodor 1980; Rey 1986] and replies that give procedural or functional accounts of intentionality [Boden 1988; Thagard 1986] may or may not shed light on the issue of intentionality, but they do nothing to make it more plausible that the Chinese room is conscious. Consequently, they leave one with the feeling that the problem the scenario poses for AI has not been addressed. At best, what has been disputed is the auxiliary premise that intentionality requires consciousness.)[4]

The Chinese room argument runs as follows. Take any program that is supposed to capture some aspect of consciousness, such as understanding Chinese or having a sensation of red. Then this program can be implemented by a monolingual English speaker—who we will call the *demon*—in a black-and-white room. The demon follows all the rules specified by the program manually, keeping a record of all the relevant internal states and variables on slips of paper, erasing and updating them as necessary. We can imagine that the demon is also connected to a robotic body, receiving digital inputs from perceptual transducers, manipulating them according to the program's specifications, and sending digital outputs to motor effectors. In this way, the program is implemented perfectly. Nevertheless, it seems clear that the demon does not consciously understand Chinese, and that the demon is not having a sensation of red. Therefore implementing a program is not sufficient for these conscious experiences. Consciousness must require something more than the implementation of a relevant program.

Proponents of strong AI have typically replied by conceding that the *demon* does not understand Chinese, and arguing that understanding and consciousness should instead be attributed to the *system* consisting of the demon and the pieces of paper. Searle has declared this reply manifestly

implausible. Certainly, there is something counterintuitive about the claim that a system of an agent and associated pieces of paper has a collective consciousness. At this point, the argument reaches an impasse. Proponents of AI argue that the system is conscious, opponents find the conclusion ridiculous, and it seems difficult to proceed any further. I think that the arguments already given provide grounds for breaking the impasse in favor of strong AI, however.

Let us assume that the relevant program is in fact a combinatorial-state automaton that reflects the neuron-level organization of a Chinese speaker who is looking at a juicy red apple. The demon in the room is implementing the CSA by maintaining a slip of paper for each element of the state vector, and updating the slips of paper at every time-step according to the state transition rules. We may run the fading and dancing qualia arguments by constructing a spectrum of cases between the original Chinese speaker and the Chinese room.[5] This is not difficult to do. First, we can imagine that the neurons in the Chinese speaker's head are replaced one at a time by tiny demons, each of whom duplicates the input–output function of a neuron.[6] Upon receiving stimulation from neighboring neurons, a demon makes the appropriate calculations and stimulates neighboring neurons in turn. As more and more neurons are replaced, demons take over, until the skull is filled with billions of demons reacting to each others' signals and to sensory inputs, making calculations, and signaling other demons and stimulating motor outputs in turn. (If someone objects that all those demons could never fit in a skull, we can imagine a scenario with radio transmission equipment outside the skull instead.)

Next, we gradually cut down on the number of demons by allowing them to double up on their work. At first, we replace two neighboring demons with a single demon doing the job of both of them. The new demon will keep a record of the internal state of both neurons he is simulating—we can imagine that this record is kept on a piece of paper at each location. Each piece of paper will be updated depending on signals from neighboring demons and also on the state of the other piece of paper. The demons are consolidated further, until eventually there is just a single demon, and billions of tiny slips of paper. We may imagine that each of these slips is at the original location of its corresponding neuron, and that the demon dashes around the brain, updating each slip of paper as a function of the states of neighboring slips, and of sensory inputs where necessary.

Despite all these changes, the resulting system shares the functional organization of the original brain. The causal relations between neurons in the original case are mirrored by the causal relations between demons in the intermediate case, and by the causal relations between slips of paper in the final case. In the final case, the causal relations are mediated by the actions of a demon—a piece of paper affects the state of the demon, which affects a neighboring piece of paper—but they are causal relations nevertheless. If

we watch the system function at a speeded-up rate, we will see a whir of causal interaction that corresponds precisely to the whir among the neurons.

We can therefore apply the fading and dancing qualia arguments. If the final system lacks conscious experience, then there must be an intermediate system with faded conscious experiences. This is implausible for just the same reasons as before. We can also imagine switching between a neural circuit and a corresponding backup circuit implemented with demons, or with a single demon and pieces of paper. As before, this would lead to dancing qualia with constant functional organization, so that the system could never notice the difference. Once again, it is much more plausible to suppose that qualia stay constant throughout.

It is therefore reasonable to conclude that the final system has precisely the conscious experiences of the original system. If the neural system gave rise to experiences of bright red, so will the system of demons, and so will the network of pieces of paper mediated by a demon. But of course, this final case is just a copy of the system in the Chinese room. We have therefore given a positive reason to believe that that system really has conscious experiences, such as that of understanding Chinese or of experiencing red.

This way of looking at things makes clear two things that may be obscured by Searle's description of the Chinese room. First, the "slips of paper" in the room are not a mere pile of formal symbols. They constitute a concrete dynamical system with a causal organization that corresponds directly to the organization of the original brain. The slow pace that we associate with symbol manipulation obscures this, as does the presence of the demon manipulating the symbols, but nevertheless it is the concrete dynamics among the pieces of paper that gives rise to conscious experience. Second, the role of the demon is entirely secondary. The interesting causal dynamics are those that take place among the pieces of paper, which correspond to the neurons in the original case. The demon simply acts as a kind of causal facilitator. The image of a demon scurrying around in the skull makes it clear that to attribute the experiences of the system to the *demon* would be a serious confusion of levels. The fact that the demon is a conscious agent may tempt one to suppose that if the system's experiences are anywhere, they are in the demon; but in fact the consciousness of the demon is entirely irrelevant to the functioning of the system. The demon's job could be performed by a simple look-up table. The crucial aspect of the system is the dynamics among the symbols.

Searle's argument gains its purchase on our intuitions by implementing the program in a bizarre way that obscures the realization of the relevant causal dynamics. Once we look past the images brought to mind by the presence of the irrelevant demon and by the slow speed of symbol shuffling, however, we see that the causal dynamics in the room precisely reflect the causal dynamics in the skull. This way, it no longer seems so implausible to suppose that the system gives rise to experience.

Searle also gives a version of the argument in which the demon *memorizes* the rules of the computation, and implements the program internally. Of course, in practice people cannot memorize even one hundred rules and symbols, let alone many billions, but we can imagine that a demon with a supermemory module might be able to memorize all the rules and the states of all the symbols. In this case, we can again expect the system to give rise to conscious experiences that are not the demon's experiences. Searle argues that the demon must have the experiences if anyone does, as all the processing is internal to the demon, but this should instead be regarded as an example of two mental systems realized within the same physical space. The organization that gives rise to the Chinese experiences is quite distinct from the organization that gives rise to the demon's experiences. The Chinese-understanding organization lies in the causal relations between billions of locations in the supermemory module; once again, the demon only acts as a kind of causal facilitator. This is made clear if we consider a spectrum of cases in which the demon scurrying around the skull gradually memorizes the rules and symbols, until everything is internalized. The relevant structure is gradually moved from the skull to the demon's supermemory, but experience remains constant throughout, and entirely separate from the experiences of the demon.

Some may suppose that because my argument relies on duplicating the neuron-level organization of the brain, it establishes only a weak form of strong AI, one that is closely tied to biology. (In discussing what he calls the "Brain Simulator" reply, Searle expresses surprise that a supporter of AI would give a reply that depends on the detailed simulation of human biology.) This would be to miss the force of the argument, however. The brain simulation program merely serves as the thin end of the wedge. Once we know that *one* program can give rise to a mind even when implemented Chinese-room style, the force of Searle's in-principle argument is entirely removed: we know that the demon and the paper in a Chinese room can indeed support an independent mind. The floodgates are then opened to a whole range of programs that might be candidates to generate conscious experience. The extent of this range is an open question, but the Chinese room is not an obstacle.

Syntax and semantics

A second argument, put forward by Searle (1984), runs as follows:

1. A computer program is syntactic.
2. Syntax is not sufficient for semantics.
3. Minds have semantics.
4. Therefore, implementing a program is insufficient for a mind.

Once again, this is put forward as an argument about intentionality, but it can also be taken as an argument about consciousness. For Searle, the central sort of intentionality is phenomenological intentionality, the kind that is inherent in consciousness.

There are various ways in which this argument can be interpreted and criticized, but the main problem is that the argument does not respect the crucial role of implementation. *Programs* are abstract computational objects and are purely syntactic. Certainly, no mere program is a candidate for possession of a mind. *Implementations of programs,* on the other hand, are concrete systems with causal dynamics, and are not purely syntactic. An implementation has causal heft in the real world, and it is in virtue of this causal heft that consciousness and intentionality arise. It is the program that is syntactic; it is the implementation that has semantic content.

Searle might argue that there is a sense in which even implementations are syntactic, perhaps because the dynamics of implementations are determined by formal properties. Any sense of "syntax" in which implementations are syntactic, however, loses touch with the sense in which it is plausible that syntax is not sufficient for semantics. While it may be plausible that static sets of abstract symbols do not have intrinsic semantic properties, it is much less clear that formally specified causal processes cannot support a mind.

We can parody the argument as follows:

1. Recipes are syntactic.
2. Syntax is not sufficient for crumbliness.
3. Cakes are crumbly.
4. Therefore, implementing a recipe is insufficient for a cake.

In this form the flaw is immediately apparent. The argument does not distinguish between recipes, which are syntactic objects, and implementations of recipes, which are full-bodied physical systems in the real world. Again, all the work is done by the implementation relation, which relates the abstract and concrete domains. A recipe implicitly specifies a class of physical systems that qualify as *implementations* of the recipe, and it is these systems that have such features as crumbliness. Similarly, a program implicitly specifies a class of physical systems that qualify as implementations of the program, and it is these systems that give rise to such features as minds.

A simulation is just a simulation

A popular objection to artificial intelligence (e.g., Searle 1980, Harnad 1989) is that a simulation of a phenomenon is not the same as a replication. For example, when we simulate digestion computationally, no food is actually digested. A simulated hurricane is not a real hurricane; when a hurricane is

simulated on a computer, no one gets wet. When heat is simulated, no real heat is generated. So when a mind is simulated, why should we expect a real mind to result? Why should we expect that in this case but not others, a computational process is not just a simulation but the real thing?

It is certainly true that for many properties, simulation is not replication. Simulated heat is not real heat. On the other hand, for some properties, simulation *is* replication. For example, a simulation of a system with a causal loop *is* a system with a causal loop. So the real question here is, how do we distinguish those types X such that a simulation of an X really is an X from those such that it is not?

I suggest that the answer is as follows: A simulation of X is an X precisely when the property of being an X is an *organizational invariant.* The definition of an organizational invariant is as before: a property is an organizational invariant when it depends only on the functional organization of the underlying system, and not on any other details. A computational simulation of a physical system can capture its abstract causal organization, and ensure that that causal organization is replicated in any implementation, no matter what the implementation is made out of. Such an implementation will then *replicate* any organizational invariants of the original system, but other properties will be lost.

The property of being a hurricane is not an organizational invariant, as it depends partly on nonorganizational properties such as the velocity, shape, and physical composition of the underlying system (a system with the same causal interactions implemented very slowly among a large set of billiard balls would not be a hurricane). Similarly, digestion and heat depend on aspects of underlying physical makeup that are not wholly organizational. We could gradually replace the biological components in a digestive system so that acid-base reactions are replaced by causally isomorphic interactions among pieces of metal, and it would no longer count as an instance of digestion. So we should not expect a simulation of systems with these properties to itself have these properties.

But phenomenal properties are different. As I have argued in Chapter 7, these properties are organizational invariants. If so, it follows that the right sort of simulation of a system with phenomenal properties will itself have phenomenal properties, by virtue of replicating the original system's fine-grained functional organization. Organizational invariance makes consciousness different in principle from the other properties mentioned, and opens the way to strong AI.

5. External Objections

I have been most concerned with internal objections to strong artificial intelligence, as these are most relevant to the topic of this book, but I will

also at least mention some external objections. As I said earlier, the *prima facie* case against external objections to artificial intelligence is strong: there is every reason to believe that the laws of physics, at least as currently understood, are computable, and that human behavior is a consequence of physical laws. If so, then it follows that a computational system can simulate human behavior. Objections are occasionally mounted all the same, however, so I will discuss these briefly.

Objections from rule following

Perhaps the oldest external objection to AI is that computational systems always follow rules, so they will always lack certain human capacities, such as creativity or flexibility. This is in many ways the weakest of the external objections, partly as it is so vague and underspecified. Indeed, it can easily be replied in turn that at the neural level, the human brain may be quite mechanical and reflexive, but this is no bar to creativity and flexibility at the macroscopic level. Of course, an opponent could always choose to deny the thesis about mechanism at the neural level, but in any case there seems to be no good argument for the thesis that computational dynamics at a basic causal level is incompatible with creativity and flexibility at the macroscopic level.

This sort of objection may gain some leverage from the implicit identification of computational systems with *symbolic* computational systems: systems that perform symbolic manipulations of high-level conceptual representations—in the extreme case, systems that inflexibly draw conclusions from premises in first-order logic. Perhaps the objection has some force in these cases, although even that is disputable. But in any case, the class of computational systems is much broader than this. A low-level simulation of the brain is a computation, for example, but is not a symbolic computation of this sort. At an intermediate level, connectionist models in cognitive science have appealed to a sort of computation that does not consist in symbolic manipulation. In these cases, there may be a level at which the system follows rules, but this is not directly reflected at the level of behavior; indeed, connectionists often claim that theirs is a method of yielding high-level flexibility from low-level mechanicality. As Hofstadter (1979) has put it, the level at which I think is not necessarily the level at which I sum.[7]

Objections from Gödel's theorem

It is sometimes held that Gödel's theorem shows that computational systems are limited in a way that humans are not. Gödel's theorem tells us that for any consistent formal system powerful enough to do a certain sort of arithmetic, there will be a true sentence—the system's *Gödel sentence*—that the system cannot prove. But we can see that the Gödel sentence is true,

it is argued, so we have a capacity that the formal system lacks. It follows that no formal system can precisely capture human capacities. (Arguments of this sort are made by Lucas [1961] and Penrose [1989, 1994], among others.)

The short answer to these arguments is that there is no reason to believe that humans can see the truth of the relevant Gödel sentences, either. At best, we can see that *if* a system is consistent, then its Gödel sentence is true, but there is no reason to believe that we can determine the consistency of arbitrary formal systems.[8] This holds particularly in the case of complex formal systems, such as a system that simulates the output of a human brain: the task of determining whether such a system is consistent might well be beyond us. So it may well be that each of us can be simulated by a complex formal system *F*, such that we cannot determine whether *F* is consistent. If so, we will not be able to see the truth of our own Gödel sentences.

There are many variants on the Gödelian argument, with replies that an opponent might make to this suggestion and further byways that come up in turn. I will not discuss these here (although I discuss them at length in Chalmers 1995c). These issues lead to many stimulating points of interest, but I think it is fair to say that the case that Gödelian limitations do not apply to humans has never been convincingly made.

Objections from uncomputability and continuity

The objections above are "high-level" arguments that cognitive functioning is uncomputable. One might also try to attack the AI position at the low level, by arguing that physical functioning is not computable. Penrose (1994) argues that there may be a noncomputable element in a correct theory of quantum gravity, for example. His only evidence for this conclusion, however, lies in the Gödelian argument above. There is nothing in physical theory itself to support the conclusion; so if the Gödelian argument is overturned, any reason for believing in uncomputable physical laws disappears. Indeed, one might argue that given that every element of the brain, such as a neuron, has only a finite number of relevant states, and given that there are only a finite number of relevant elements, then the relevant causal structure of the brain *must* be capturable in a computational description.

This leads to the final objection, which is that brain processes may be essentially *continuous* where computational processes are discrete, and that this continuity may be essential to our cognitive competence, so that no discrete simulation could duplicate that competence. Perhaps in approximating a neuron by an element with only a finite number of states, one loses something vital to its functioning. An opponent might appeal, for example, to the presence of "sensitive dependence on initial conditions" in certain nonlinear systems, which implies that even a small round-off error at one

stage of processing can lead to major macroscopic differences at a later stage. If brain processing is like this, then any discrete simulation of the brain will yield results that differ from the continuous reality.

There is good reason to believe that absolute continuity cannot be essential to our cognitive competence, however. The presence of background noise in biological systems implies that no process can depend on requiring more than a certain amount of precision. Beyond a certain point (say, the 10^{-10} level on an appropriate scale), uncontrollable fluctuations in background noise will wash out any further precision. This means that if we approximate the state of the system to this level of precision (perhaps a little further to be on the safe side—to the 10^{-20} level, for example), then we will be doing as well as the system itself can reliably do. It is true that due to nonlinear effects, this approximation may lead to behavior different from the behavior produced by the system on a given occasion—but it will lead to behavior that the system *might* have produced, had biological noise been a little different. We can even approximate the noise process itself, if we want to.[9] The result will be that the simulating system will have the same behavioral *capacities* as the original system, even if it produces different specific behavior on specific occasions. The moral is that when it comes to duplicating our cognitive capacities, a close approximation is as good as the real thing.

It is true that a system with unlimited precision might have cognitive capacities that no discrete system could ever have. For example, one might encode an analog quantity corresponding to the real number whose nth binary digit is 1 if and only if the nth Turing machine halts on all inputs. Using this quantity, a perfect continuous system could solve the halting problem, something no discrete system can do. But the presence of noise implies that no biological process could reliably implement this system. Biological systems can rely on only a finite amount of precision, so human and animal brains must be limited to capacities that discrete systems can share.

6. Conclusion

The conclusion is that there do not appear to be any in-principle barriers to the ambitions of artificial intelligence. The external objections do not appear to carry much force. The internal objections may be more worrying, but none of the arguments for these objections seem to be compelling on analysis; and indeed if the arguments I have given in previous chapters are correct, then we have good positive reason to believe that implementation of an appropriate computation will bring conscious experience along with it. So the outlook for machine consciousness is good in principle, if not yet in practice.

I have said little about just what *sort* of computation is likely to suffice for conscious experience. In most of the arguments I have used a neuron-by-neuron simulation of the brain as an example; but it is likely that many other sorts of computations might also suffice. It might be, for example, that a computation that mirrors the causal organization of the brain at a much coarser level could still capture what is relevant for the emergence of conscious experience. And it is likely that computations of an entirely different form, corresponding to entirely different sorts of causal organization, could also give rise to rich conscious experiences when implemented.

This picture is equally compatible with the symbolic and connectionist approaches to cognition, and with other computational approaches as well. Indeed, one could argue that the centrality of computation in the study of cognition stems from the way that computational accounts can capture almost *any* sort of causal organization. We can see computational formalisms as providing an ideal formalism for the expression of patterns of causal organization, and indeed (in combination with implementational methods) as an ideal tool for their replication. Whatever causal organization turns out to be central to cognition and consciousness, we can expect that a computational account will be able to capture it. One might even argue that it is this flexibility that lies behind the often-cited *universality* of computational systems. Proponents of artificial intelligence are not committed to any one sort of computation as the sort that might suffice for mentality; the AI thesis is so plausible precisely because the class of computational systems is so wide.[10]

So it remains an open question just what class of computations is sufficient to replicate human mentality; but we have good reason to believe that the class is not empty.