ANDREW MELNYK

SEARLE'S ABSTRACT ARGUMENT AGAINST STRONG AI

ABSTRACT. Discussion of Searle's case against strong AI has usually focused upon his Chinese Room thought-experiment. In this paper, however, I expound and then try to refute what I call his abstract argument against strong AI, an argument which turns upon quite general considerations concerning programs, syntax, and semantics, and which seems not to depend on intuitions about the Chinese Room. I claim that this argument fails, since it assumes one particular account of what a program is. I suggest an alternative account which, however, cannot play a role in a Searle-type argument, and argue that Searle gives no good reason for favoring his account, which allows the abstract argument to work, over the alternative, which doesn't. This response to Searle's abstract argument also, incidentally, enables the Robot Reply to the Chinese Room to defend itself against objections Searle makes to it.

John Searle's case against strong AI is often thought to consist entirely in his notorious Chinese Room thought-experiment, originally published in 1980. But in subsequent work (e.g. Searle 1984, 39) Searle has been quite explicit in presenting an apparently independent argument for the same conclusion. In a recent and very succinct formulation (Searle 1991, 526), he expresses the argument in these words:

1. Programs are formal (syntactical).
2. Minds have contents (semantic contents).
3. Syntax is not identical with nor sufficient by itself for semantics.

From these we can derive:

Programs are not sufficient for nor identical with minds; i.e. strong AI is false.

Call this argument Searle's *abstract argument* against strong AI. My aim in this paper is first to reconstruct this argument as sympathetically as possible, and then to show that, and exactly where, it nevertheless goes wrong. The reader may be relieved to learn that I will be adding only incidentally to the already abundant, and often excellent, discussion of the Chinese Room.[1]

My refutation of Searle's abstract argument against strong AI will concentrate on its first premiss, which states that programs are formal (or syntactical). I shall claim that in this premiss Searle relies on a certain account of what a program is, and of what it is for a machine to run one: given the account, the premiss is true, and may be combined with the other two premisses to yield an argument that is probably sound. However, I shall describe an alternative account of a program and its implementation, which *cannot* in any obvious way be combined with the remaining premisses of the argument so as to yield a sound argument against strong AI. Yet Searle, I shall argue, gives no sufficient reason for preferring his favored account of a program over the alternative which does not serve his argumentative purpose. So Searle's abstract argument, I shall conclude, rests on the undefended assumption that a particular account of a program and of what it takes to implement one is correct. Since I take it that, by definition, a machine is computing iff it is implementing some program, I could express my conclusion, equivalently, by saying that his argument rests on an undefended assumption about the nature of computation; but I shall continue to speak of programs and of implementing (or running) them.[2]

The abstract argument is, I believe, Searle's best argument against strong AI, and in fact I suspect that he may only ever have intended the Chinese Room as a vivid dramatization of it. If this suspicion is correct, then the Chinese Room may only be laid to rest for good once the abstract argument itself has been refuted. And I aim, of course, to supply a refutation.

Though I will not discuss the Chinese Room as such, it will turn out that my refutation of the abstract argument is interestingly related to the so-called Robot Reply to the Chinese Room argument (see Searle 1980 and 1984, Ch. 2). One important connection is that my refutation of the abstract argument enables an advocate of the Robot Reply to answer two objections to it which Searle makes. In consequence of this connection, I shall set out my refutation of the abstract argument indirectly, by way of responding to these two objections. One advantage of this indirect strategy is that it makes clear that those objections to the Robot Reply cannot be successfully wielded against my reply to the abstract argument.

Here is how the paper will go. In the first section I present my reconstruction of Searle's abstract argument against strong AI, emphasizing the role played by the key premiss that *syntax is not sufficient for semantics.* I must apologize in advance for the length of the exposition, but I have two excuses. First, I want to show that even when the abstract argument is reconstructed as sympathetically as possible, it can still be shown to be defective. Secondly, I need to present the argument in enough detail to be able to show exactly where the defect lies. With the reconstruction of the abstract argument in place, I then sketch the Robot Reply to the Chinese

Room argument, together with the two responses to it that Searle makes. In the three sections to follow, which form the heart of my paper, I argue that neither of those responses succeeds. It is in the course of doing this that I hope to identify exactly what is wrong with Searle's abstract argument, namely, its dependence upon an unmotivated account of what a program is and of what it is to run one. For all that Searle has said, strong AI may still be true, and thought still a kind of computation.

1.

I begin, then, by presenting my reconstruction of the abstract argument, my reconstruction, that is, of the argument I quoted from Searle a few paragraphs back. Since my reconstruction, of necessity, greatly expands upon, and hence deviates from, Searle's exact words, the question naturally arises whether it is faithful to Searle's intentions, as expressed both in that quotation and elsewhere. My confidence that it is a faithful reconstruction rests upon three considerations. The first is the principle of charity: if my reconstruction is accepted, then Searle may be credited with an extremely interesting and plausible (albeit subtly flawed) argument. The second is that if my reconstruction is rejected, we are left, in effect, with no detailed understanding of the abstract argument at all, even though its author evidently deems it to be of the utmost importance to his assessment of strong AI; so my reconstruction seems to be, at present, the only game in town. The third consideration is that, aside from making sense of the quotation at the start of this paper, my reconstruction also makes sense of various other passages from Searle's writings; as I proceed I shall cite these passages, sometimes in the body of the text, though sometimes in footnotes.

Suppose, then, that we wish to construct a machine that is in a mental state which exhibits intentionality, a machine that is, say, *thinking about Vienna.* Then Searle's opponent, the advocate of strong AI, will maintain that in order to do this *all* we have to do is to find the right program and build a machine to run it; this is all we have to do in the sense that it is necessarily true that if a machine is running the right program, it is thinking of Vienna.[3] Of course, such a program may be highly complex, and we may not currently know what it is; but if strong AI is correct, there is such a program. According to strong AI, then, some program, P, is such that running P is sufficient – metaphysically – for thinking of Vienna; that is, some program, P, is such that in every possible world in which an object, x, is running P, x is in that world thinking of Vienna.[4]

The conclusion of the abstract argument is, of course, the denial of this thesis. But how is this denial to be supported? It is obvious, from the stress which Searle himself lays on it (e.g. 1984, 34 & 39), that the key premiss in the supporting argument is the assertion that *syntax is not sufficient for semantics.* Three questions arise at once. (i) What does this assertion mean? (ii) Why should we suppose it to be true? And (iii) how exactly does it combine with the other premisses of the abstract argument to yield the conclusion that strong AI is false? Let me address each question in turn.

(i) Consider a set of physical symbols, i.e. physical symbol-types. These symbols might be inscribed or spoken words in some natural language, or they might be formulae in a logical system, or primitive symbols in that system, or they might be kinds of marks in sand, or kinds of patterns of smoke, or whatever. 'Symbol' is not being used here in a sense which presupposes or implies that all symbols have to be meaningful, or interpreted; indeed, in this sense of 'symbol', symbol-types are little more than types (including *state*-types) that can be individuated in some non-semantic way (though perhaps they must in some sense have the potential to possess semantic properties). Now to specify a *syntax* for a set of symbol-types is to specify a set of rules which lay down how tokens of those types are to be combined with, or related to, one another; examples of rules of syntax in the case of logical systems would be formation rules which tell us what is and what is not a well-formed formula, or syntactic rules of inference which tell us something about which formulae may follow which others. All syntactic rules specify relations in which tokens of symbol-types may or must stand to one another. By contrast, to specify a *semantics* for a set of symbol-types is to state what tokens of the symbol-types, whether alone or in combination, *mean*, or are *about.*[5]

So what does it mean to claim that syntax is not sufficient for semantics? Presumably that, for any set of symbol-types, no facts merely about how tokens of those symbol-types are, or may be, related to one another are sufficient – metaphysically sufficient – for it to be the case that any of those symbol-tokens gets to *mean* anything, or to be *about* anything. Or, to put it differently, two sets of symbol-types might be exactly alike syntactically (i.e. in respect of the rules governing their inter-relations), and yet differ semantically (e.g. in respect of what strings of them are about): the syntactic facts do not determine the semantic facts (or even that there are any semantic facts). So no matter how fancily we specify the syntax of a set of symbol-types, it will remain an open question what the symbol-tokens of those types are about (or, indeed, whether they are about anything at all).

(ii) Searle (1984, 39) describes the premiss that syntax is not sufficient for semantics as "a conceptual truth", and, perhaps for that reason, offers no argument for it. But the claim, whether or not a conceptual truth, is surely very plausible, as reflection on a couple of examples (not Searle's) may show. Consider a natural language, such as Russian. Assuming you know no Russian, I could teach you everything syntactical about Russian: which words are words of Russian, how Russian words may be combined with one another so as to yield grammatical sentences, and so on. Armed with this knowledge, you would presumably be able to recognize Russian sentences, and to distinguish grammatical from ungrammatical sentences in Russian. And yet you need not know the meaning of a single Russian word or sentence. (Imagine that I replaced traditional grammatical terms such as 'verb' and 'noun', which provide some clue as to the meaning of words which are verbs and nouns, with coinages which do not, such as 'Type A Symbol' or 'Type B Symbol'.) In fact, you would only have my word for it that what you had learnt was a language at all. For if I had been in a mischievous mood, I might have taught you the syntax, not of Russian, but of some quite meaningless formal system of my own invention! Knowledge, at least, of the syntax of a set of symbols seems not to be metaphysically sufficient, in general, for knowledge of its semantics.

Someone might object that *knowing* the syntactic facts does not guarantee *knowing* the semantic facts merely because semantic facts cannot be *explicitly defined* in terms of syntactic facts, so that no *inference* from syntactic facts to semantic facts is possible, even though syntactic facts do actually determine semantic facts, metaphysically, in the sense that any two possible worlds exactly alike syntactically are exactly alike semantically.[6] But a different example casts considerable doubt on this objection. Take some relatively simple logical system, such as the first-order predicate calculus. Notoriously, students of this system are apt to supply no interpretation for the predicate letters in their formalizations of natural language arguments, perhaps on the assumption that it is just obvious what the interpretation is. But of course it is not obvious: we need to *supply* an interpretation. Moreover, we can give any interpretation we like to our predicate letters, and indeed give *different* interpretations to the *same* letters on different occasions. And of course we can make all these semantic changes while the formation and inference rules of the predicate calculus – its syntax – remain quite unchanged. Since the formulae of first-order logic can bear many different interpretations, we have a case in which semantic facts can vary even while the syntactic ones do not, so that it cannot in general be true that syntactic facts determine – are metaphysically sufficient for – semantic facts.[7] The point is more than merely epistemological.

(iii) I take it as very plausible, then, that syntax is not metaphysically sufficient for semantics. But how can this claim be combined with the remaining premisses of the abstract argument to yield the conclusion that strong AI is false? Well, suppose, in accordance with strong AI, that a machine *could* be programmed to think about Vienna; then, since (Premiss 2) thinking of Vienna, like other mental states, is a state with a *semantic* property (in this case, that of *being about Vienna)*[8], some internal state of the machine, by virtue *solely* of the programming, would have to have been endowed with precisely that semantic property. But what it means to say that (Premiss 1) programs are purely formal (or syntactic) is merely that all that programming a machine amounts to, as such, is providing the internal states of the machine with a *syntax*. So the supposition that a machine could be programmed to think about Vienna (i.e. that programming alone would be metaphysically sufficient) entails that providing the internal states of a machine with a *syntax* (via programming) would be metaphysically sufficient for providing at least one of those internal states with a *semantic* property, namely that of being about Vienna. However, since (Premiss 3) syntax is *not* metaphysically sufficient for semantics, that is, no amount of determining the syntax of a set of states is metaphysically sufficient for any of those states to possess any semantic property at all, it follows that a machine could *not* be programmed to think about Vienna; the programming all by itself would not be metaphysically sufficient. In short: the internal states of a computer are just symbols, and if it is true that syntax is not metaphysically sufficient for semantics, then no amount of relating those symbols to one another – which is all that programming can achieve – is going to guarantee that those symbols mean, or are about, anything at all; and if those symbols (= the computer's internal states) are not guaranteed to be about anything at all, then they cannot be guaranteed to be (intentional) mental states, contrary to the thesis of strong AI.

As I have reconstructed the abstract argument here, it evidently depends crucially upon the premiss that programs are formal (or syntactical), which I have glossed as meaning that all that programming a computer amounts to, as such, is providing a machine's internal states with a syntax.[9] But this premiss, I suggest, gains whatever plausibility it has (and Searle gives no argument for it) from a certain account of what a program *is*, and of what it takes for a machine to be running one, which we need to suppose that Searle is assuming. So we must make this account explicit, and show how it supports the claim.

Before doing so, let me briefly say why I think we should suppose Searle to be assuming this account. I note, to begin with, that we surely must suppose Searle to be assuming *some* account of what a program is; for how else, in principle, could the claim that programs are formal (or syntactical) be defended than by appeal to some understanding of what a program is? My reasons, however, for attributing to Searle the specific account to follow are three-fold. First, if he is taken to be assuming the account to follow, then he can be credited with a plausible reason for believing his important premiss that programs are formal (or syntactical). Secondly, the account of what a program is, and of what is involved in a machine's running one, that I shall shortly expound is not eccentric or peculiar. It is presented in a textbook, apparently with endorsement, by John Haugeland (1985, Ch. 2), and a very close relative of it is presented (though without endorsement) by William Lycan (1987, 28–30), and described by him as "the going notion" (italics removed). These facts surely constitute *some* reason to suppose that Searle is assuming just this account; certainly he gives no indication whatsoever that the account he assumes is in any way unusual or unconventional. My third reason for supposing Searle to assume the account is an assortment of remarks of his that can be well explained by making that supposition; I shall cite these remarks, when appropriate, in footnotes.

Everyone will allow that to program a machine is, roughly, to cause the machine to be running a program.[10] What, then, is a program? And what is it for a machine, M, to run (or realize, or implement) a program, P? (a) We may say that, on the account I take Searle to assume, to specify a program is to lay down a set of rules[11] which describe the permissible and obligatory transitions from one kind of what I shall call *abstract program state* to another. A rule in such a set might take the following form: if the machine is in a state of type S_1, and goes into an input-state of type I_1, then it must move to a state of type S_2 and go into an output-state of type O_1.[12] I call the states referred to by such expressions as 'S_1', 'I_1', 'S_2', and 'O_1' abstract program states, since, in order to allow for the possibility that the same program be run by physically very different machines, these expressions cannot be taken to refer to physical states (e.g. voltage levels); the specification must abstract away from the particular physical states that implement the program states.[13] Such abstract program states are to be individuated solely by reference to their place in the abstract mathematical structure which is the program itself.[14]

(b) What is it, then, for a machine, M, to *run* a program, P? M must be a machine which is capable of being in tokens of a number of different (presumably physical) state-types, where the state-types can be individuated non-semantically. Let us suppose that we can set up a one-one mapping between every program state-type mentioned in a specification of P and every member of some set of state-types of the machine, M[15]; we may call members of this set of state-types of the machine, each of which is paired with exactly one program state-type, *program relevant* machine state-types. (Usually, of course, there will be many state-types of a given machine which are not program relevant.) Then, according to the account

I take Searle to assume, a logically necessary and sufficient condition for a machine, M, to be running program P is that the transitions, both actual and counterfactual, between tokens of the different program relevant state-types that the machine can occupy exactly parallel (given the mapping) those transitions between tokens of abstract program state-types which are allowed or required by program P.[16]

I said earlier that if Searle is taken to be assuming this account of programs and of running them, then he can be credited with a plausible reason for believing his important premiss that programs are formal (or syntactical). Here is why. If the account is correct, it certainly seems to follow that programming a computer is, in itself, merely a matter of causing its internal states to be related to one another in certain rule-governed ways, i.e. merely a matter of endowing those states with a syntax. For, according to the account, a machine's *running* any given program is merely a matter of its states' being related to one another in certain rule-governed ways; and so *causing* a machine to run a program (i.e. programming it) is merely a matter of *causing* its states to be related to one another in certain rule-governed ways.

I have proposed, on a variety of grounds, that we should take Searle to be assuming this account of programs in his abstract argument. But let me now stress a vitally important feature that his assumption must have, if it is to do him any good: since Searle's final conclusion is that *no* kind of program is such that running that program metaphysically suffices for thinking of Vienna, he has to assume that the above account of programs is a correct account of *every* kind of program that there is; his sweeping conclusion about the limitations in principle of *all* programs, of whatever kind, could not otherwise be reached. But exactly this assumption, of course, is what I shall in the end challenge.

Let me now pull together all the threads of my reconstruction of Searle's abstract argument against strong AI. If it is true, as strong AI alleges, that a computer could be programmed to think of Vienna (i.e. that programming is all that it would take), then, given that it is running the right program, the computer's being in a token of some program relevant state-type must be metaphysically sufficient for the machine to be in a token of the semantic state-type, *thinking of Vienna*. But how could this possibly be metaphysically sufficient? For the partisan of strong AI must regard those program relevant state-types as *symbol*-types, and their tokens as *symbol*-tokens. Moreover, given Searle's account of running a program, the only thing that makes those program relevant state-types program relevant is that their tokens are related to one another, actually and counterfactually, in certain ways, i.e. those allowed by the rules of the program. But if syntax is not metaphysically sufficient for semantics, that is, if no amount of specifying permissible relations among tokens of symbol-types metaphysically suffices to endow tokens of those types with semantic properties, then no token of any program relevant state-type which the computer is in is *automatically*, just in virtue of its being a token of a program relevant state-type, a token of any semantic state-type such as *being about Vienna.* And so strong AI is false: no program is such that running that program is metaphysically sufficient for thinking of Vienna. There may exist thinking machines, but their thinking cannot be a metaphysically necessary consequence of their running some appropriate program.

2.

The best way of developing my refutation of the abstract argument will be to discuss critically Searle's responses to the so-called Robot Reply to the Chinese Room argument, a reply which Searle himself explicitly considered in his original paper. According to the Robot Reply, in order to make a computer think about Vienna, or be in any intentional state, it would be metaphysically sufficient to build the computer into a robot; that is, to locate the computer inside some sort of artificial body in such a way that the computer's inputs would derive, causally, from the outside world, via artificial sense-organs, and (perhaps) the computer's outputs would causally affect the outside world, via artificial limbs.[17] Talk of building the computer into a robot, however, is best regarded merely as a way of dramatizing the philosophical idea that what is needed to ensure that a computer is thinking about Vienna is *appropriate causal connections* to the outside world and its contents (e.g. to Vienna); perhaps certain internal states of the computer must actually be caused, via a particular kind of causal chain, by states of the outside world, or perhaps the internal states must just be nomically dependent in some suitable way on outside states.[18]

Searle seems to make two responses to the Robot Reply. The first is quite explicit (Searle 1980, 420). He charges the Robot Reply with having in effect conceded his main point, viz. that merely implementing a program, whatever the program, does not suffice for thinking about anything. His point is conceded, he says, because the reply seems to insist that, *in addition to* the computer's running the right program, some of the computer's states must be *causally related* in some suitable way to the world external to the computer; so running the program is not enough all by itself to ensure mental states with intentionality.[19] Searle's second response to the Robot Reply is to claim that artificially embodying the computer as described above in any case makes no difference, since "the same thought experiment [the Chinese Room] applies to the robot case" (Searle 1980, 293). And the root of the problem once again is "the inexorability of the semantics-syntax distinction" (Searle 1984, 34–5); somehow the insight that syntax is not sufficient for semantics also refutes the idea that being a suitably-programmed *and suitably-embodied* computer suffices for having thoughts about things. So, as a repetition of the Chinese Room thought experiment shows, embodying the computer fails to get us any nearer to a non-biological artifact that is thinking of Vienna; and it fails because it does not overcome the old problem that syntax is not sufficient for semantics. I shall discuss this second response first, aiming both to illuminate and to criticize it.

3.

Searle's second response to the Robot Reply is bold and full of interest, not least because it seems to imply that one could not give a naturalistic account of intentionality in terms of appropriate causal or nomic connections between mental symbols on the one hand and things in the outside world on the other.[20] But what exactly does this response come to? And does it succeed?

Part of the response is clearly to urge a repetition, with appropriate changes, of the Chinese Room thought-experiment. And his claim is that, if we repeat the thought-experiment, but now insist that the batches of Chinese script which Searle receives as input and emits as output are causally connected, in some appropriate way, to things in the outside world, we get the same result: the conditions alleged to be metaphysically sufficient for intentionality are present, and yet there still seems, intuitively, to be no intentionality. But since Searle's intuitions about this particular case have not been universally shared[21], and since in any case an appeal to intuition of this sort will always be questionable, it would be much better if Searle could give an abstract argument to support this response to the Robot Reply, similar to the one expounded in Section 1, and similarly independent of one's intuitions about such cases as the Chinese Room; and indeed the remark I quoted from him in the last section about the inexorability of the semantics-syntax distinction perhaps suggests he would wish to give such an argument. But could he do so successfully? I shall argue that if he does so by simply extending the abstract argument of Section 1, then he will not succeed. And if he does not simply extend that argument, I do not see how else he can mount an argument of the required sort.

Let us consider, then, what would happen if Searle were to try to extend the abstract argument of Section 1 to the case of an artificially embodied computer – a *robot* for short – in the hope of showing that no kind of robot is such that being a robot of that kind is metaphysically sufficient for thinking of Vienna (where being a robot of a given kind would be a matter not merely of running a certain program but also of being causally related in certain ways to the outside world). Specifically, let us concentrate upon the role in that extended argument of the premiss, so crucial to the original abstract argument, that syntax is not sufficient for semantics.

Notice, first, that the premiss in its original form is useless to Searle. For, interpreted so as to serve the needs of the original argument, it states that no amount of specifying the permissible relations *among tokens of symbol-types* metaphysically suffices to endow any of those tokens with semantic properties. But owing to the clause italicized, the premiss thus interpreted proves quite irrelevant to the case of the robot. For the idea that, in order to be thinking of Vienna, a computer needs somehow to be embodied is *precisely* the idea (as Searle himself agrees) that, in order to be thinking of Vienna, the internal state-tokens of a computer need to be related not just to *one another* but also to *other things*, i.e. to things in the outside world which are not symbols.[22] Therefore, even if syntax is not sufficient for semantics in the sense that no amount of relating symbols *to one another* metaphysically suffices to endow any of the symbols with semantic properties, it might still be true that there is some way of programming and embodying a computer such that being a computer programmed and embodied in those ways is metaphysically sufficient for being in a state with semantic properties; for this might be true in virtue of relations holding between token-states of the computer (i.e. symbols) and token-states of the world *external* to the computer (i.e. non-symbols), relations which the embodiment of the computer is designed precisely to ensure.

So if Searle is to show that being a robot never suffices for thinking of Vienna, he requires a suitably-strengthened version of the claim that syntax is not sufficient for semantics. The minimal version that would meet his needs appears to be this: no amount of specifying the permissible relations among tokens of symbol-types, *and between tokens of symbol-types and other things*, metaphysically suffices for any of those tokens to possess semantic properties. If it is true, in this strengthened sense, that syntax does not suffice for semantics, then Searle can indeed have the premiss he needs for an abstract argument against the idea that a suitably-embodied computer running a suitable program would necessarily be thinking of Vienna. (For all that embodying the computer seems to add is certain causal relations between token-states of the computer and token-states of things outside the computer.) But *is* it true, in the strengthened sense, that syntax is not sufficient for semantics?

I shall not argue that it is false. Rather, I contend merely that Searle gives us no reason at all, still less a conclusive reason, for supposing that this key premiss is true; hence his anti-robot argument is inconclusive. Two considerations are crucial here. First of all, the original version of the claim that syntax does not suffice for semantics (i.e. the claim as originally interpreted) does not entail the strengthened version of the claim. That no amount of specifying the relations among tokens of symbol-types metaphysically suffices to endow any of the symbol-tokens with semantic properties does not entail that no amount of specifying the relations among tokens of symbol-types *and tokens of other types* metaphysically suffices to endow any of the symbol-tokens with semantic properties. So Searle cannot support the latter (strengthened) claim by deducing it from the former (original) claim. Secondly, the rather powerful intuitive considerations I presented on Searle's behalf in Section 1 to support the original version of the claim that syntax does not suffice for semantics do not carry over to support also the strengthened version of the claim. For neither of the examples I gave (Russian syntax and the uninterpreted predicate calculus) involved symbol-tokens which were related to other things as well as to one another; the relations were purely inter-symbolic. So, as far as I can see, Searle has no support for the strengthened version of the claim that syntax does not suffice for semantics, and hence no support for the premiss he needs in order to mount a satisfying abstract argument, analogous to the original abstract argument, against the idea that there is some way of programming and embodying a computer such that being programmed and embodied in that way metaphysically suffices for thinking about things. But without that argument, he seems to have no good argument against that idea at all.[23]

4.

So much for Searle's second response to the Robot Reply. Let me turn now to his first, which claimed, of course, that the Robot Reply concedes his main point, since, by adding a requirement of embodiment, it in effect accepts that running a program does not metaphysically suffice all by itself for thinking about things, no matter what the character of the program. It will turn out that this response assumes the correctness of precisely that account of a program and its implementation which was expounded in Section 1 in the course of reconstructing the abstract argument. The task of this section is to show that *rejecting* (or at least not accepting) that account of programs and their implementation permits not only a convincing rebuttal to Searle's first response to the Robot Reply, but also, and more importantly, a plausible reply to his abstract argument against strong AI.

Let us begin, though, by recalling the account of a program and its implementation that, according to my reconstruction in Section 1, Searle assumes in his abstract argument. Specifying a program consists of presenting a set of rules of the following kind:

(R) If a machine is in a state of type S_1, and goes into an input-state of type I_1, then it must move to a state of type S_2 and go into an output-state of type O_1

And such rules, to the extent that they can be said to describe anything, do not describe physical features of machines (not even high-level features of them) but rather the relations among what I called abstract program states. A physical machine can then be said to be running a program specified like this just in case, for some one-one mapping of abstract program states onto discrete physical states of the machine, the transitions, both actual and counterfactual, between tokens of the physical state-types the machine can occupy exactly parallel (given the mapping) the transitions between abstract program state-types allowed or required by the program. As Searle himself (1990, 26) puts it, "The physics is irrelevant except insofar as it admits of the assignments of 0's and 1's and of state transitions between them."

Now this account of a program as a whole and its implementation by a whole system has an important implication for what is involved in the implementation (or realization) of a *particular state* of the program (e.g. an input state) by a *particular state* of the system. The account implies that, as I shall often express it, *implementing (or realizing) states* (i.e. the program-relevant states of a machine which is running or implementing some program) *are specified purely abstractly.* What I mean is that, on this account, *all* that is required for a particular physical state-token to implement a particular program state is that it be a token of some program relevant state-type of a physical *system* which can correctly be said to be running the program in virtue of some one-one mapping between some sub-set of the system's states and all the abstract program states mentioned in the program. To be sure, satisfying this condition is not trivial – some physical token-states may be disqualified from being implementers of the particular abstract program states of certain programs because they are states of systems too inflexible to count as running those programs – but it is surely very easy. So long as a one-one mapping can be set up between abstract program states, on the one hand, and internal states of the machine, on the other, and so long as, relative to this mapping, the program relevant states of the machine are related to one another, actually and counterfactually, in a way which exactly mirrors the relations specified by the program to hold between its abstract program states, the machine is running the program. No further conditions must be satisfied by the machine's internal states; there are no intrinsic physical characteristics they must possess, nor relations they must stand in, above and beyond those necessitated by fulfilling the sufficient condition for implementing a program just stated. A token of pretty much *any* kind of physical state of a system is in principle a candidate for implementing an abstract program state.[24]

Searle himself is well aware of this implication of the definition of a program and its implementation that he assumes. He remarks, in his (1990, 26–7), that "On the standard textbook definition of computation", with which he evidently feels obliged to operate, "... the wall behind my back is right now implementing the Wordstar program, because there is some pattern of molecule movements which is isomorphic with the formal structure of Wordstar." His point, of course, is that, viewed sub-atomically, a wall is such a complex thing that there is just *bound* to be some way of mapping its physical states (which presumably might include gerrymandered disjunctive states) onto the abstract program states of Wordstar in such a fashion that exactly those relations obtain among the physical states as must hold, according to Wordstar, among the program states; so, given the above account of a program and its implementation, the wall satisfies a sufficient condition for running Wordstar.[25]

The account of a program and its implementation that Searle assumes, then, implies that implementing states are specified purely abstractly. Consequently, *if* we accept that account, then we must grant that his first response to the Robot Reply is quite correct: the Robot Reply does indeed concede, in effect, that running a program is not metaphysically sufficient for intentionality. For the Robot Reply concedes this, according to Searle, since it insists that, in order to make a computer that thinks, it would be necessary, not *only* to ensure that it was running the right program, but *also* to embody it, so as to ensure that its internal states were appropriately connected, causally, to the outside world. But this requirement of embodiment only appears to Searle to be an *additional* requirement, over and above that of running the right program, because, according to his account of a program and its implementation, implementing states are specified purely abstractly, so that a machine's internal states would not need to be connected to the outside world, in whatever ways are required by embodiment, for it to count as running the program; the isomorphism is all that matters. So

Searle's first response to the Robot Reply rests squarely upon the account he assumes of a program and its implementation.

But that is not all, of course. For we saw in Section 1 how the abstract argument against strong AI relies upon the very same account; according to my reconstruction, the account is assumed by its first premiss. But the dependence of the argument's first premiss on this account of a program and its implementation has a further consequence: in order to avoid the fallacy of equivocation, the argument's *conclusion* must be understood in light of the same account. The conclusion states, of course, that no program, P, is such that running P is metaphysically sufficient for thinking of Vienna. But this conclusion will not be validly derivable unless 'program' and 'running', which appear in it, are understood as they are in the first premiss. Properly understood, then, what the abstract argument concludes is that no program, *as understood by the above account of programs and their implementation,* is such that running it is sufficient for intentionality. And, since this conclusion is supposed to be equivalent to the negation of strong AI, the abstract argument must also assume that the *affirmation* of strong AI should be understood in light of the same account: strong AI must be taken to claim, not merely that some program is such that implementing that program is metaphysically sufficient for intentionality, but that some program, *as understood by the above account of programs and their implementation,* is metaphysically sufficient for intentionality.

So both Searle's first response to the Robot Reply and his abstract argument against strong AI assume the correctness of a particular account of a program and its implementation, one implying that implementing states are specified purely abstractly, so that a machine can run a given program merely in virtue of a suitable isomorphism between its own internal states and the abstract program states mentioned in the program. By rejecting that account, we can undermine both the response to the Robot Reply and, more significantly, the abstract argument. In what follows, I will first sketch an alternative account of a program and its implementation; then show exactly how it undermines Searle's claims; and finally argue that this alternative account is a plausible alternative to Searle's, and one which he gives no good reason for rejecting. This will leave Searle in the position of arguing against strong AI on the basis of an account of programs and their implementation which is essentially arbitrary.[26]

5.

The alternative account of a program and its implementation resembles the one Searle assumes very closely; in fact, its account of a program is exactly the same. The difference lies in its account of what conditions are necessary and sufficient for the implementation of a program. For, although, according to the alternative account, it is still a *necessary* condition of a machine's implementing a program that there be a suitable isomorphism between its internal states and the abstract program states mentioned in the program, it is not a *sufficient* condition.[27] To achieve a sufficient condition, it is required in addition that, in the case of some program states (e.g. some or all of the input states), any states of a machine implementing them have to be causally connected, in some appropriate way, to states of the outside world. So, for instance, it might be specified that, in addition to being appropriately related to other states, any machine state realizing input state I_1 must be a state which is causally connected, in such-and-such a way, to a particular kind of object or state in the outside world.[28] We might hope to get the details about the exact nature of the causal connection from our complete naturalistic theory of intentionality; but suppose, for the sake of a crude illustration, that the machine state realizing input I_1 must be such that its tokens are nomically dependent upon red objects in the external world, the state realizing I_2 must be such that its tokens are nomically dependent on round objects, and so on. Now if implementing states are specified in this richer, less than purely abstract, way, then implementation of a program by a machine will require more than it does on the account Searle assumes. In particular, even if there is a one-one mapping between program states, on the one hand, and machine states, on the other, such that, relative to the mapping, the relations between tokens of the program relevant machine states exactly mirror the relations specified to hold between program states by the program, the machine may still fail to be running the program; it may fail because those of its states supposedly realizing the program's (say) input states are not causally connected in the right ways to states of the outside world. Notice, however, that even if implementing states are not specified purely abstractly, there is still a sense of 'formal' in which programs are formal. For even on this alternative account of a program, it remains true that program-relevant states of a machine running a program fall under those causal laws in virtue of which the machine passes from one state to another only because of their non-semantic features – their 'shape', as Searle puts it.[29] (One might even speculate that perhaps Searle has mistakenly inferred formality in the sense of purely abstractly specified implementing states from formality in this other sense.)

It might be objected to this alternative account of a program and its implementation that it squanders one major advantage of the account Searle assumes, viz. the advantage of allowing for the multiple realizability of computational states. But this objection is largely mistaken. It is indeed harder for a physical system to implement a program according to the new account than it is to implement one according to the old. But programs, even according to the new account, nevertheless admit of very considerable multiple realizability, and for two reasons: first, it may only be the states that implement some program states (e.g. the input states), and not all, which, according to the new account, may be specified in a less than purely abstract way; and secondly, the causal connections to external things which a machine state must have in order to count as realizing a program state will presumably be describable at a high level (relative to fundamental physics), and will therefore themselves be multiply realizable by fundamental physical states of very different types. So the alternative account allows for as much multiple realizability as anyone could decently expect.

But how would adoption of this alternative account of a program undermine Searle's claim that the Robot Reply concedes his point that no program is such that running that program is metaphysically sufficient for intentionality? Well, Searle's argument for this claim was that the requirement of embodiment is *additional* to the requirement of running the right program – as indeed it is if one accepts the account of a program and its implementation that Searle assumes. But it would *not* be additional, if one accepted instead the alternative account, according to which, for a machine to run a program, some of its states might have to be causally connected to the outside world in certain ways. For, we may recall, being embodied just *is* being causally connected to the world in certain ways. Thus adoption of the alternative account of a program and its implementation enables the Robot Reply to avoid having to concede that running a program fails to be sufficient metaphysically for intentionality.

How would adopting the alternative account undermine the abstract argument, as reconstructed in Section 1? Adopting it permits one to reject Searle's definition of strong AI, and instead to understand strong AI as claiming that there is some program, *as understood by the alternative account of programs and their implementation,* such that running that program is metaphysically sufficient for thinking about Vienna (or whatever). Understood in this way, however, strong AI seems to be quite immune to Searle's abstract argument against it. For if strong AI is understood in line with the alternative account, it can insist that, in order to be running any program sufficient for semantic properties, a machine must *not only* be such that its program-relevant states are properly related to one another, but *also* such that some tokens of its program relevant state-types are causally connected to the outside world in certain, specified ways. It will then no longer be possible to claim that programs are formal (or syntactic), i.e. that all that programming a machine does is to ensure that its program relevant states are related to one another in a certain way; for programming will also involve ensuring the holding of appropriate causal connections between certain machine states and external things. But in that case, the pivotal premiss that syntax is not sufficient for semantics, i.e. that no amount of specifying the relations among tokens of a set of symbol-types suffices, metaphysically, to endow any of those tokens with semantic properties, will no longer conflict with the thesis of strong AI that programming a machine can be sufficient to endow the machine with semantic states; for programming will involve *more than* specifying relations among symbols (= program relevant machine states). Could Searle perhaps defend a strengthened version of his claim that syntax does not suffice for semantics which does conflict with what the newly-defined strong AI claims? He could easily formulate such a claim, but a main moral of Section 3 was that he could defend it neither by inferring it from the original claim nor by appealing to the intuitive cases offered in support of the original claim; defense would therefore require some new argument. So Searle's abstract argument against strong AI cannot be used, at least in any straightforward way, to refute strong AI as understood in light of the alternative account of a program and its implementation.

We have arrived at the following position: the abstract argument works if strong AI is formulated using the account of a program and its implementation that Searle assumes, but fails if strong AI is formulated using the alternative account. Evidently, then, Searle needs it to be the case that the account he assumes is correct (and, indeed, correct to the exclusion of any alternative account – it must correctly characterize the only kind of program there is). So I must argue that my alternative account is a genuine alternative (i.e. plausible, and certainly not just a totally arbitrary invention), but that Searle has given no adequate reason for rejecting it in favor of his own. If I am right, then we can say that *Searle's abstract argument against strong AI is inconclusive because it rests upon a definitional assumption that we have been given no reason to accept.* Notice that this objection to Searle does not require proving the *superiority* of the alternative account, since the point is that the account Searle assumes is undefended, not that it is definitely false; so the mere plausibility of an alternative is enough. Indeed, this objection to Searle does not even need it to be true that there *is* any objectively superior account of a program and its implementation; for if there is not, the objection to Searle simply becomes the charge that he has arbitrarily precisified an inherently vague notion to suit his argumentative purposes. Notice, also, that if the alternative account can be made plausible, the objection to Searle will not be open to the charge of being a thoroughly *ad hoc* maneuver, a merely arbitrary redefinition of a key term; for the alternative account will have something going for it *independently* of the present dispute.

My first task, then, is to argue that the alternative account of a program and its implementation offered above is plausible enough for it to rate as a genuine alternative to Searle's. Now, to the extent that we have an intuitive concept of a program, it is, I suggest, that of a set of rules such that, if strictly followed, some task (paradigmatically, perhaps, an arithmetical computation) is accomplished. In addition, the following of a *single* rule must be a simple affair, at least in comparison with the complexity of the task. Correlatively, a machine implements a program when it follows such a set of rules, and can be programmed to perform some task when it can be made to perform the task as a result of following some set of rules.[30] These ideas, as I say, appear to represent our commonsense or intuitive notion of a program. But if they do, then, owing to their very vagueness, they leave plenty of room for *different* ways of making them more precise and rigorous. In particular, nothing in these intuitive ideas requires that implementing states be specified in a purely abstract way; for it is perfectly possible to frame rules concerning states whose implementation requires more than the mere holding of an isomorphism. So nothing precludes our regarding the alternative account of a program and its implementation as being one legitimate way of precisifying the intuitive concept of a program and its implementation; for the alternative account does, after all, retain the core idea of a program as a set of rules. But in fact the alternative account can claim an intuitive advantage over the one Searle assumes, having to do with the relative ease with which a physical system can implement a program according to the two accounts. As earlier observed, it is harder for a system to implement a program according to the alternative account. But this is good. Consider again Searle's claim that, on the account of a program and its implementation he assumes, the wall behind him was running Wordstar. That claim seems just false, and if the account he assumes entails it, then so much the worse for that account. And yet we can, I think, make sense of the idea that a wall is running Wordstar. But to do so we need to imagine the wall causally hooked up to – or at least capable of being causally hooked up to – appropriate input and output devices, such as a keyboard and a printer, perhaps via some apparatus sensitive to quantum phenomena. However, the imaginability of this seems to play into the hands of the alternative account, for it suggests that even ordinarily we impose requirements on implementing a program which go beyond those that the account Searle assumes would include, requirements which may include appropriate causal connections to external things.

Suppose, then, that the alternative account is at least plausible enough that Searle is under some obligation to say why he assumes the account he does assume. Do we find in his writings any adequate reason for this preference? The answer is not at all clear; Searle clearly seems to think that he *has* to accept this account, but it is not clear why. However, there is some indication that he thinks that he finds it in the writings of Alan Turing, and that such a pedigree is enough to establish the propriety of relying on it.[31] Leaving aside the apparent appeal to authority here, I will attend just to Searle's claim that the account he assumes of a program and its implementation is Turing's. For Turing's importance in the history of computing is sufficient to invest this claim with some interest. I will argue that no support for the account Searle assumes, *as compared to the alternative account,* can be found in Turing's best known writings (i.e. his 1936 and 1950).

The question, then, is whether Turing required that all implementing states be specified in a purely abstract way, so that all it takes for a physical device to implement a program is the obtaining of a certain isomorphism between the program states mentioned in the program and the (program relevant) physical states of the device. But since Turing did not speak of programs, or of program states, or indeed of computers (except in the sense of persons who compute), the question needs to be translated into something closer to his terminology. When this is done, the question becomes this: did Turing require that the states (whether input, output or internal) mentioned in a machine table describing a given computing machine be such that their implementation is nothing more than the holding of a suitable isomorphism between them and the physical states of some system?

The answer to this question, I claim, is No: for all that Turing said, or needed to say, some of the states mentioned in a machine table might perfectly well be such that their implementers are not specifiable purely abstractly, so that any computing machine truly said to be describable by the table would have to do more than merely stand in some suitable isomorphism to the states mentioned in the table. In defense of this negative answer I will make three points. The first is that, as far as I can see, nowhere in his two best known papers (including his seminal 1936 paper) does Turing insist that the implementers of all machine table states must be specified purely abstractly. Short of providing a blow-by-blow analysis of those papers, all I can do to persuade any reader skeptical of my claim is to urge them to return to Turing's papers and to try to identity *exactly* where he states the requirement I deny he imposes; of course, I expect such a search to be in vain.

My second point is that sometimes Turing seems rather to take it for granted that a machine table might specify inputs and outputs in such a way that their implementation must involve more than the mere holding of an isomorphism. For example, throughout the 1936 paper, in which Turing's concern is with computing numbers, it seems clear that machine tables must (and hence may) specify outputs as states to be realized only by states which are (or include) expressions standing for numbers (i.e. as states whose implementers are not specified purely abstractly); for instance, in the "operations" column of the simple illustrative table on p. 233 we find the instruction, "P1", which (it is clear) means 'Print an expression which stands for, or which at least can be thought of as standing for, the number, 1!'. If the instruction is not understood in this way (that is, if it is understood as saying just 'Print a mark with such and such a shape!'), then by what right could Turing claim (as he does) that the table describes a machine capable of computing a certain *number*? A machine whose states were suitably isomorphic to the states mentioned in this table, but whose outputs could *not* (for some bizarre reason) be regarded as expressions for numbers, could surely not be computing any numbers.

Similarly, the whole of Turing's 1950 paper rests on the assumption that a computing machine in his sense could successfully play the Imitation Game – could take English questions as inputs and yield sensible English answers to those questions as outputs – *just in virtue of* being a computing machine, i.e. just in virtue of being describable by some table. But surely the table for a machine capable of doing this would have to specify the inputs and outputs in such a way that their implementers had to be English sentences (and so not specified purely abstractly). For if it did not, then the table would, in virtue of some isomorphism, equally well describe a machine which, since its inputs and outputs were *not* English sentences, could *not* play the Imitation Game; in which case one could not say what Turing seems to have wanted to say, namely, that a machine which *did* play the game well did so *just in virtue of* being describable by some table, and hence being a computer. So there may be grounds for holding that Turing just took it for granted that implementing states might be specified in some more than purely abstract way.

My third (and most important) point is that Turing originally introduced his concept of a computing machine with a specific purpose in mind, and that for this purpose he did not *need* to require machine tables to describe states whose implementers are specified in a purely abstract way. So we should not be surprised to find, as I have claimed we do find, no trace of such a requirement in his most famous papers.

The subject of Turing's classic 1936 paper is computable numbers, and his preliminary definition of computable numbers is that they are "those whose decimals are calculable by finite means" (p. 231). It is for the sake of making this definition more precise that he introduces the concept of a computing machine; having done so, he can then say that, "A number is computable if it differs by an integer from the number computed by a circle-free machine" (p. 233). In addition to making precise the notion of being "calculable by finite means", the concept of a computing machine must also, it seems, remain faithful to that of a *human* computer, e.g. a human computing a real number; for Turing explicitly introduces (and in part justifies) his concept of a computing machine by analogy with that of a computing human (see p. 231, and, especially, pp. 249–252). So a computing machine must compute – in the pre-1936 sense in which a human could be said to be computing – and it must do so "by finite means".

The point of mentioning these constraints which Turing imposes upon the concept of a computing machine is simple: it seems they can be satisfied, and Turing's intellectual purposes achieved, *without* insisting that all the states mentioned in a machine table be such that their implementers are specified in a purely abstract way. Now Turing evidently regarded a human computer, e.g. one doing a subtraction, as proceeding by rigidly following a smallish set of rules which specified exactly one thing to do for any situation the human might be in, where the following of a rule on a specific occasion would require the performance of some very simple operation, such as striking out a "9", writing "8" in its place, and writing a superscript "1" just to the right of it; and this way of regarding a computing *human* inspired Turing's conception of a computing *machine* as a machine capable of being in a finite number of states and of performing a finite number of primitive operations, whose transitions from state to state and performances of operations are in strict accordance with the set of (deterministic) rules codified in a machine table. Note, however, that it need *not* be required that the machine table describe states (or operations) whose implementers are specified in a purely abstract way. The machine will *still* be finite (finite states, finite number of primitive operations, finite set of rules), and its behavior will *still* be determined by the rules (given its input and current state), even if the machine table describes some states or operations whose implementers are specified in some more than purely abstract way. Neither the goal of precisifying the notion of a computable number, nor the constraint that a computing machine remain relevantly analogous to a computing human, requires machine tables (i.e. descriptions of programs, in effect) to describe only states whose implementers are specified in a purely abstract way. Turing, I claim, had no motivation for imposing this requirement.[32]

Let me now sum up my discussion of Turing. Searle hints that the account of a program which he presupposes in his formulation of the thesis of strong AI, in his abstract argument against it, and in his charge that the Robot Reply concedes his main point can at least be recommended on the grounds that it is the account of a program and its implementation to be found in the seminal work of Turing. I have claimed, in response, that this account is one which (a) Turing did not in fact give, which (b) he seems, at least implicitly, to have rejected, and which (c) he had no philosophical motivation to give. If I am right, then Searle cannot appeal to Turing in order to support the claim he needs, namely, that the account of a program and its implementation that he assumes, with its requirement that all implementing states be specified purely abstractly, should be preferred to the alternative account which lacks this requirement. But if the appeal to Turing fails, Searle's preference seems entirely unsupported. And so he is left without support for the central pillar of his abstract argument against strong AI, the claim that all that programming can do is something metaphysically insufficient for intentionality.

6.

I will conclude with a brief account of what I hope I have achieved. While reconstructing Searle's abstract argument against strong AI as charitably as possible, I have tried to show that, and exactly how, it still fails. For Searle's argument presupposes a certain account of a program and its implementation; and his insight that syntax does not suffice for semantics refutes strong AI only if the programs of which strong AI speaks conform to that account. I have challenged that account, on the grounds that there is no reason to prefer it to an account which leaves the viability of strong AI an open question (at least as far as Searle's argument goes).

So do we learn nothing from Searle on AI? Not at all. First of all, we learn, I think, that *if* programs are understood *à la* Searle, then there is indeed no program such that anything running that program is bound, metaphysically, to have a mind. Partisans of strong AI, if such there be, who accept the Searle-assumed account are therefore in error.[33] Secondly, Searle drew forceful attention to the important point, perhaps in some circles in danger of being forgotten, that non-eliminativist materialists must provide some suitably naturalistic account of the fact that people's thoughts are thoughts *about* things, and that providing such an account is not likely to be easy. In doing so, Searle performed a valuable service.

Finally, however, if what I have been arguing is correct, then we learn also from Searle that the notion of a program and its implementation need careful philosophical attention, no doubt much more than I have given in this paper. For this insight we should also be grateful to Searle. But we need not abandon strong AI.[34]

NOTES

[1] See, for example, David Cole (1984) for a very nice piece, in which he argues (amongst other things) that it does not follow from the fact that Searle, when inside the Chinese Room, denies *in English* that he understands Chinese, that he does not in fact understand Chinese. In a later paper (Cole 1991), however, he apparently repudiates this objection, replacing it with a different one according to which it does not follow from the fact that *Searle*, when in the Chinese Room, does not understand Chinese, that *no* person understands Chinese; the implementation of the program might produce a person *distinct* from Searle, a person who *did* understand Chinese; thus, for all that Searle has shown, running the right program might still suffice for there being *some* Chinese-understander. For interesting critical discussion of this objection, though, see Selmer Bringsjord (1992, 194–202). In contrast to the Chinese Room, however, the abstract argument has attracted remarkably little attention, perhaps because it is regarded as depending for its cogency on the Chinese Room. (This is the dependence that I deny in calling it *in*dependent of the Chinese Room.) William Rapaport (1988a, 596–7; see also his 1988b) is one of the few writers I have seen who discusses the abstract argument, but even he supposes that it draws support from the Chinese Room, and hence is not independent. (I think it more likely that the abstract argument provides a rationale for Searle's intuitive response to the Chinese Room.) It is confusing, however, that the problematic mental state in the Chinese Room is that of *understanding Chinese*, whereas in the abstract argument it is any propositional attitude; it is far from obvious that understanding Chinese is, or involves, a propositional attitude.
[2] I discovered as this paper was nearing completion that Daniel Dennett, right at the end of a discussion of the abstract argument (1987, 336–7), also suggests, against Searle, that programs may not be purely formal. Since he makes this suggestion in the space of a single 15-line paragraph, it is hard to say how far his suggestion and mine coincide, or how far he might approve of my working out of the suggestion. At any rate, he says that "Whether a program is to be identified by its purely formal characteristics is a hotly contested issue in the law these days. . . . If details of 'embodiment' are included in the specification of a program, and are considered essential to it, then the program is not a purely formal object at all. . . '. With this last conditional I agree (though I cannot comment on the legal claim Dennett makes); and of course I agree that the abstract argument is to be faulted for assuming that details of 'embodiment' need *not* be included in the specification of a program and its implementation. But Dennett adds a remark whose meaning, I confess, escapes me entirely: ". . . without *some* details of embodiment being fixed – by the internal semantics of the machine language in which the program is ultimately written – a program is not even a syntactic object, but just a pattern of marks as inert as wallpaper".
[3] Compare Searle (1980, 296): "According to strong AI, instantiating a formal program with the right input and output is a sufficient condition of, indeed is constitutive of, intentionality."
[4] (i) Searle evidently understands the thesis of strong AI to make a modal claim (e.g. in his (1984, pp. 28–9) he says that, according to strong AI, "if [a computer] had the right

program, it would *have* to have a mind"; my italics), but he is not at all clear about the kind of modality in question. In the text, I have construed the modality as metaphysical, in the sense introduced by Saul Kripke in his (1980, 35–6), mainly because to construe the modality as merely physical makes the thesis of strong AI too weak (since it would then be consistent with some kind of property dualism about intentionality), whereas to construe it as analyticity makes the thesis incredible (since it is surely not analytic that running the right program gives one a mind). The notions of metaphysical necessity and sufficiency, though widely invoked, are not uncontroversial, and I do not use them with an entirely clear conscience. But in fact nothing in my paper turns on the decision to construe the modality as metaphysical; it only matters (for the sake of formal validity) that having chosen a construal for the conclusion, we consistently use the same construal in interpreting the premisses. Georges Rey, who (1986, 183) very briefly discusses the abstract argument in a footnote, misses the point that Searle's conclusion is the negation of a *modal* claim, and as a result mistakenly accuses him of arguing invalidly.

(ii) My formulation of the thesis of strong AI (in line with Searle's) assumes that the very same object which is running the program is also in the resulting mental state; the variable, 'x', appears in both antecedent and consequent. But following Cole (1991), one might prefer to put the strong AI thesis this way: some program, P, is such that in every possible world in which an object, x, is running P, some object, y, is in that world thinking of Vienna (where y might $\neq x$). I do in fact prefer this formulation, but I shall use the cruder version in the text for the sake of a more readable exposition. The difference will not affect the points I wish to make.

[5] Here I articulate the distinction between syntax and semantics in a way which is, I think, quite standard in recent philosophy of mind and language; partly because it is standard (compare Searle's (1980, 300) remark: "*In the linguistic jargon*, . . . syntax . . . semantics"; my italics), and partly because Searle gives no indication that his usage is at all non-standard, it can safely be attributed to him. But this distinction between syntax and semantics seems to be quite different from that suggested by Rapaport (1986, 273–4) between syntactic understanding and semantic understanding. In a later paper (1988a, 597), Rapaport offers two suggestions concerning what Searle means by 'semantics', when he claims that programs lack semantics. Both suggestions (if I have grasped them; they are obscure) strike me as needlessly elaborate; surely Searle just wants to know what it is about running a program which ensures that the machine running it is in states, like beliefs that *represent* or are *about* the world.

[6] I would anticipate this kind of objection from a non-reductive materialist who holds that physical facts determine all facts, even though, because higher-level facts do not *reduce* to physical facts, the higher-level facts cannot be deduced or derived from the physical facts. See especially Ch. 4 of John Post's (1987).

[7] Note that it is *physically* possible for the semantic facts to vary while the syntactic facts do not. This shows that an opponent of Searle could not retreat to the claim that syntax is merely *physically* sufficient for semantics.

[8] Searle repeatedly expresses the view that members of a large class of mental states possess semantic (or intentional) properties (e.g. 1980, 288; 1984, 39).

[9] I gloss the premiss in this way partly because, thus glossed, it gives Searle what he needs to generate an interesting and plausible argument against strong AI, but partly also because of such remarks as this (1984, 33): "All that the computer has . . . is a formal program for manipulating uninterpreted Chinese symbols . . . a computer has a syntax, but no semantics." The essence of the abstract argument, I think, is that the states of a machine form nothing but an uninterpreted formal calculus if all you do is program it.

[10] So one way to program a machine, according to this definition, is to *hard-wire* it a certain way, which is, of course, how the first computers were programmed; so programming a machine, according to this definition, does not require installing a *stored* program (even though doing the latter is what many people today understand by 'programming').

[11] Talk of following rules in this connection should, of course, be understood as a mere *façon de parler;* see Cole (1984, 439–40) and indeed Searle himself (1984, 46–8).

[12] This account of a program may seem to treat inputs and outputs as states, rather than as symbols; indeed, it treats *all* symbols as states. But I think this is acceptable. States seem perfectly permissible candidates for being symbols; and anyway there could presumably be an elaborate paraphrase of the account in terms of symbols. Notice also that, on this account of a program, there is no deep or principled distinction between input/output states and other states; it's just an arbitrary convention to designate only some program states as input or output states. If this consequence were found too offensive to intuition, the account could be modified a little to avoid it: we could add that an input (program) state is one whose implementing (physical) state must be a causal consequence of *some* state *external* to the system implementing the program (though *what kind* of external state is left entirely unspecified); and that an output (program) state is one whose implementing (physical) state must cause *some* state *external* to the system implementing the program (though what kind of external state is also left entirely unspecified). I doubt that Searle would (or should) approve of this modification, but as far as my *criticisms* of Searle go, it makes no difference.

[13] Compare what Searle has to say here, in the course of setting out what he takes to be the standard view of computation (1990, 25; his italics): "To find out if an object is really a digital computer, it turns out that we do not actually have to look for 0's and 1's, etc.; rather we just have to look for something we could *treat as* or *count as* or *could be used to* function as 0's and 1's ... it turns out that this machine could be made out of just about anything."

[14] Here I represent a program as being, strictly speaking, an abstract, presumably mathematical, object; see Lycan (1987, 28–9) for an apparent attempt to nominalize away talk of programs and avoid commitment to abstract objects.

[15] Compare Searle's remarks here (1990, 26; his italics): "The class of computers is defined syntactically in terms of the *assignment* of 0's and 1's. ... The physics is irrelevant except insofar as it admits of the assignments of 0's and 1's and of state transitions between them."

[16] The relativity of a machine's running a program to some mapping between abstract program states and physical states of the machine is, I think, what Searle is referring to when he says (1990, 35) that "... computation is not discovered in the physics, it is assigned to it. Certain physical phenomena are assigned or used or programmed or interpreted syntactically. Syntax and symbols are observer relative."

[17] That the computer's output states would causally affect the outside world will play no part in what follows; what is crucial to the Robot Reply, I think, is the idea that the outside world causally affects the computer's input states. Contrast J. Christopher Maloney (1987, p. 352).

[18] Georges Rey (1986, 171–2) seems to interpret the point of the Robot Reply rather differently, claiming that the computer's internal states must be related not just to one another but also to the inputs and outputs of other *programs.* This requirement doesn't seem *in itself* to ensure the causal links with the outside world that I want to insist on.

[19] Rapaport (1986, 276 and 1988a, 598) argues against Searle in effect that since human brain states acquire semantic properties somehow, it is not at all clear why computers should not acquire them in the same way – whatever that way is. Fair enough. But will this way

turn out to be by running a program, and nothing else? The answer matters, since if it isn't solely by running a program, then Searle will say, as he says to the Robot Reply, that his main point has been conceded: running a program isn't sufficient all by itself for generating semantic (and hence mental) properties.

[20] I mean to allude to a category of accounts broad enough to include those of Fred Dretske (e.g. his 1988, Ch. 3) and Jerry Fodor (e.g. his 1990, Chs.3 & 4), as well as the Kripkean theories suggested by Kim Sterelny (1991, Ch. 6.3) and those of wide-role functionalists such as Frank Jackson and Philip Pettit (1988).

[21] See, for example, pp. 223–4 of Kim Sterelny's (1991), where Sterelny claims that it is just not clear that the whole system – Searle in the room plus causal connections – lacks understanding of Chinese, even if it is clear that Searle himself (in the example) does not understand Chinese. This combines the Robot Reply with the so-called Systems Reply, and I am inclined to agree with Sterelny that this is the best response to the Chinese Room thought-experiment. However, I wish to stress that the Systems-plus Robot Reply is *by itself* unable to overcome Searle's objection (to the Robot Reply) that it concedes his main point; the combined reply needs to be supplemented with the rejection of Searle's conception of a program, for which I go on to argue.

[22] Or if they are symbols, then at least they are not the symbols which we currently seek to endow with semantic properties. (If they were symbols, then that would be because we wanted our original symbols to be *about* other symbols. I do not mean to suggest that merely by linking symbols up with other symbols we could endow any of them with any semantic properties; to take that line would be to attack premiss three of the abstract argument, as Rapaport does in his (1988b), where he argues that in some sense syntax *does* suffice for semantics.)

[23] And no good argument either, I note, against aspirant naturalizers of intentionality such as those mentioned in note 20.

[24] The claims of this paragraph would require qualification if the account of a program under discussion were modified in the way mentioned in note 12. According to that modification, a program state only counts as an input (or output) state if it is such that the state of a physical system that implements it is caused by (or causes) some state *external* to the physical system, where (crucially) the external state is specified in no more detail than by saying that it *is* external. The effect of this modification is obviously to make it just a little bit harder for a system to implement a program (and hence for a particular state of a system to implement a particular program state) than I say it is in the text. On the other hand, it is equally obvious that, even with the modification, it remains very easy indeed for a system to implement a program, and for a particular state of a system to implement a particular program state.

[25] I should stress that Searle finds this feature of the account of a program and its implementation that he uses as absurd as I do; but he seems to think that he nevertheless *has* to use it, perhaps because there is no alternative, perhaps because he thinks it is Turing's definition. These assumptions are very doubtful, as we shall see.

[26] The alternative account may be arbitrary too, of course, but even if it is, it will not follow that Searle's account is *not* arbitrary. I lose nothing by arbitrariness; Searle loses his case against strong AI. Similarly, I can rest easy – though Searle can't – with the view that the concepts of young sciences shouldn't be given precise definitions, but should instead be permitted to co-evolve with the science itself.

[27] It is worth stressing that *any* account of a program and its implementation which makes the holding of a suitable isomorphism an *in*sufficient condition for implementing a program will pose a threat to Searle's abstract argument; so the *spirit* of my reply to the abstract

argument could be retained, even if the particular account of a program and its implementation I describe in the text were rejected.

[28] The reader will recall that in note 12 (see also note 24) I mentioned a possible modification to the account of a program that Searle uses, according to which a program state only counts as an input (or output) state if it is such that the state of a physical system that implements it is caused by (or causes) some state *external* to the physical system, where (crucially) the external state is specified in no more detail than by saying that it *is* external. That last clause is what marks the difference between that modification and the alternative account of implementation just laid out in the text. According to the modification mooted in note 12, the external cause for the realizer of an input program state is only required to be external, nothing more. But according to the alternative account of implementation in the text, much more is required: the state (of a physical system) implementing a particular input state must be causally dependent in some particular, specified way on some external state (or perhaps must be caused by an external state via some particular, specified kind of causal chain). The detailed specification that is required of the mode of dependence (or of the kind of causal chain) is what is crucial.

[29] Concerning himself in the Chinese Room, he says (1980, 284), "... all that 'formal' means here is that is that I can identify the symbols entirely by their shapes." Applied to the case where the CPU is not a human being, this idea seems equivalent to what I say in the text.

[30] As Cole (1984, 439–40) and Searle (1984, 46 – 8) both warn, one must not take talk of a machine's following rules too seriously. I do not think that I have done so, and it is a handy way of speaking.

[31] See p. 25 of his (1990). I suspect that Searle also holds that his account is the one actually used by AI researchers. But since, as far as I know, he never supports this attribution by citing textual evidence, I think that what we have here is probably an inference from (a) the claim that AI researchers say they rely on Turing's ideas about computation, and (b) the claim that the account is Turing's.

[32] An interesting straw in the wind: Ned Block's well-known introduction to functionalism illustrates the idea of a Turing machine with a machine table whose inputs and outputs are such that their implementers are *not* purely abstractly specified; see p. 173 of his (1980).

[33] It is not easy to say what definition of a program actual AI researchers have had in mind, since it is not a question that worries them very much: they are too busy getting on with the business of writing and testing programs. Or, at least, such is my impression, based on a very unsystematic survey.

[34] Thanks for comments on earlier drafts and talks based on this material to John Barker, Bob Gordon, Paul Humphreys, Peter Markie, Paul Roth, and an anonymous referee.

REFERENCES

Block, Ned: 1980, 'Introduction: What Is Functionalism?', in Ned Block (ed.), *Readings in Philosophy of Psychology*, Vol. 1, Methuen, London, pp. 171–184.

Bringsjord, Selmer: 1992, *What Robots Can and Can't Be*, Kluwer Academic Publishers, Dordrecht.

Cole, David: 1984, 'Thought and Thought Experiments', *Philosophical Studies* **45**, 431–44.

Cole, David: 1991, 'Artificial Intelligence and Personal Identity', *Synthese* **88**, 399–417.

Dennett, Daniel: 1987, *The Intentional Stance*, The MIT Press, Cambridge, MA.

Dretske, Fred: 1988, *Explaining Behavior: Reasons in a World of Causes*, The MIT Press, Cambridge, MA.

Fodor, Jerry: 1990, *A Theory of Content and Other Essays,* The MIT Press, Cambridge, MA.

Haugeland, John (ed.): 1981, *Mind Design: Philosophy, Psychology, Artificial Intelligence,* The MIT Press, Cambridge, MA.

Haugeland, John: 1985, *Artificial Intelligence: The Very Idea*, The MIT Press, Cambridge, MA.

Jackson, Frank and Philip Pettit: 1988, 'Functionalism and Broad Content', *Mind* **97**, 381-400.

Kripke, Saul: 1980, *Naming and Necessity*, Basil Blackwell, Oxford.

Lycan, William G.: 1987, *Consciousness*, The MIT Press, Cambridge, MA.

Mahoney, J. Christopher: 1987, 'The Right Stuff', *Synthese* **70**, 349–72.

Post, John: 1987, *The Faces of Existence,* Cornell University Press, Ithaca, New York.

Rapaport, William: 1986, 'Searle's Experiments With Thought', *Philosophy of Science* **53**, 271–9.

Rapaport, William: 1988a, 'Critical Review: John Searle, *Minds, Brains and Science*', *Nous* **22**, 585–609.

Rapaport, William: 1988b, 'Syntactic Semantics: Foundations of Computational Natural Language Understanding', in James H. Fetzer (ed.) *Aspects of Artificial Intelligence* (Dordrecht, Holland: Kluwer Academic Publishers), 81–131.

Rey, Georges: 1986, 'What's Really Going On In Searle's "Chinese Room"?', *Philosophical Studies* **50**, 169–85.

Searle, John R.: 1980, 'Minds, Brains, and Programs', *The Behavioral and Brain Sciences* **3**, 417–4. (Page references in the text are to the version reprinted in Haugeland (1981).)

Searle, John R.: 1984, *Minds Brains, and Science*, Penguin, London.

Searle, John R.: 1990, 'Is The Brain A Digital Computer?', *Proceedings and Addresses of the American Philosophical Association* **64**(3), 21–37.

Searle, John R.: 1991, 'Yin and Yang Strike Out', in David M. Rosenthal (ed.), *The Nature of Mind,* Oxford University Press, New York, pp. 525–6.

Sterelny, Kim: 1991, *The Representational Theory of Mind: An Introduction*, Basil Blackwell, Cambridge, MA.

Turing, Alan M.: 1936, 'On Computable Numbers, With An Application To The Entscheidungsproblem', *Proceedings of the London Mathematical Society (2)*, **42**, 230–65.

Turing, Alan M.: 1950, 'Computing Machinery and Intelligence', *Mind* **59**, 433–60.

Department of Philosophy
438 General Classroom Building
University of Missouri – Columbia
Columbia, MO 65211
USA