In *Proceedings 9th International Joint Conference on AI,*
pp 995 -1001, Los Angeles, August 1985.

WHAT ENABLES A MACHINE TO UNDERSTAND?

by
Aaron Sloman
Cognitive Studies Programme
University of Sussex
(Now School of Computer Science, The University of Birmingham)

Abstract

The 'Strong AI' claim that suitably programmed computers can manipulate symbols that THEY understand is defended, and conditions for understanding discussed. Even computers without AI programs exhibit a significant subset of characteristics of human understanding. To argue about whether machines can REALLY understand is to argue about mere definitional matters. But there is a residual ethical question.

Topic area and keywords

Philosophical foundations, machines, language, meaning, understanding, reference, Strong AI.

Introduction

Filing cabinets contain information but understand nothing. Computers are more active than cabinets, but so are copiers and card-sorters, which understand nothing. Is there a real distinction between understanding and mere manipulation? Unlike cabinets and copiers, suitably programmed computers appear to understand. They respond to commands by performing tasks; they print out answers to questions; they paraphrase stories or answer questions about them. Does this show they attach meanings to symbols? Or are the meanings 'derivative' on OUR understanding them, as claimed by Searle([10])? Is real understanding missing from simulated understanding just as real wetness is missing from a simulated tornado? Or is a mental process like calculation: if simulated in detail, it is replicated?

I argue that there is no clear boundary between things that do and things that do not understand symbols. Our ordinary concept of 'understanding' denotes a complex cluster of capabilities, and different subsets of these may be exhibited in different people, animals or machines. To ask 'which are necessary for REAL understanding?' is to attribute spurious precision to a concept of ordinary language.

Instead of answering either 'YES' or 'NO' to the question whether suitably programmed computers can understand, we note that within the space of possible 'behaving systems' (including animals) there are infinitely many cases, some sharing more features with human minds, some fewer. The important task is to analyse the nature and the implications of these similarities and differences, without assuming existing English words can label the cases adequately.

Dennett [2] thinks we can justifiably take the 'intentional' stance towards any machine or organism whose behaviour thereby becomes easier to predict or explain. Searle [10], [11] retorts that behaviour is not enough, alleging that a suitable program could make a system appear to understand Chinese when it doesn't really, e.g. if Searle is inside executing the programs. In [18] I show that he actually attacks an extreme and implausible thesis, namely that ANY 'instantiation' of a suitable program would understand. But he is right in suggesting that actual behaviour is not what mental concepts refer to. How the behaviour is produced is relevant. There are significantly different ways in which the same behaviour might be generated. For instance a huge lookup table, prepared by an extraordinarily foresightful programmer who anticipated all our questions, could pass a collection of behavioural tests. But it might produce nasty surprises later, because no finite set of *actual* tests can establish the powers required for passing a wider range of *possible* tests. Since there are indefinitely many counterfactual conditional statements that are true of us, but which would not be true of such a machine, we would be unwise to rely on it in future simply because it has worked so far, without knowing the basis for success.

Attributions of mentality imply coherent behaviour and *reliability,* as friends, enemies, colleagues, or goal achievers. There are different kinds of unreliability. One kind would exist in a machine whose computations depended on co-operation of a (speeded up) human interpreter performing millions of steps, as in Searle's experiment. Tiredness, boredom, cussedness, and mere slips could easily interfere. This supports Searle's claim that mentality presupposes machinery with the right causal powers (though not his other conclusions).

The lookup table is unreliable in a deeper way: we cannot rely on it to deal with the unanticipated. The same applies, to a lesser degree, to less rigid programs: human-like performance in any finite set of tests does not justify the assumption that the behaviour would be convincing in other possible situations. This is painfully evident in AI programs to date.

So, taking the intentional stance on purely behavioural grounds (Turing's test), is potentially risky. We must adopt what Dennett calls the 'design stance' for a better justification of our ascriptions of intentionality, understanding, etc. A machine must not merely produce appropriate behaviour, but must satisfy the *design* requirements for understanding. Could a machine do this?

The main features of human understanding are sketched below. We'll find important aspects of our ordinary concept of 'understanding' in simple computers, even without AI programs. Requirements for richer human-like capacities are also described. There are no reasons for doubting that machines can satisfy them.

The Semantic Linkage Problem

A central issue is the 'semantic linkage problem': how can a person, or machine, take one thing as referring to or describing another? AI work on language and image understanding often relies on translation into some internal representation. But if the machine itself does not understand the internal representation, we have not progressed much beyond filing cabinets. If all understanding requires translation we risk an infinite regress. Ultimately something must be interpreted as meaningful in its own right. How? It is implausible that existing AI story 'understanders' really can think about parties, political events, or passionate murders, despite printing out sentences about them after reading stories. If a symbol-user U uses a symbol S to refer to some object O, then it seems that U must have some *other* way of relating to O, attending to O, thinking of O, etc., besides using S. This 'semantic linkage' problem pervades recent analytical philosophy

(E.g. See [17], [6], [3]). It is ignored in work on formal semantics, and both linguistics and psychology seem to have little to say about it. It is complicated by the fact that O can be remote from U, or even long dead, or imaginary, which rules out direct causal connections between U, S and O, as necessary. We shall see that when O is *part* of U (e.g. a location in U's memory, an internal action U can perform, an internal pattern U can test for), the link may be a comparatively simple causal relationship. My conjecture is that more sophisticated types of meaning and reference are possible only on the basis of this 'internal' semantics.

What is understanding a language?

I use the word 'language' loosely as equivalent to 'notation', 'representational scheme', 'symbol system' etc. Very roughly, a language L is a system of symbols used by some agent U in relation to a world W. A full analysis would distinguish different kinds of: (a) symbol media, (b) symbol systems, (c) mechanisms for manipulating symbols, (d) symbol users, (e) worlds, and (f) purposes for which symbols might be used. This paper discusses only a subset of this rich array of possibilities.

Symbols are structures that can be stored, compared with other structures, searched for, etc. They may be physical structures, like the marks on a piece of paper, or virtual symbols, i.e. abstract structures in a virtual machine, like 2-D arrays in a computer (See [15]). They may be internal or external. They need not be separable physical objects or events, since a single travelling wave may 'carry' different signals simultaneously, and a network of active nodes may have several patterns superimposed in its current state. Symbols include maps, descriptions, representations, of all kinds, including computer programs, and non-denoting symbols, like parentheses and other syntactic devices. (In fact, anything at all can be used as a symbol.)

A language L contains symbols used by U to represent or refer to entities, properties, relations, events, processes, or actions in some world W. The word 'used' may suggest that U has goals or purposes. However, this is not a necessary condition, since a plant "uses" water in photosynthesis without having any explicit goal, or purpose. We can tell that U uses a symbol S to refer to object O, by discovering that some significant subset of the conditions listed below are satisfied. We shall see that in the more elaborate cases goals are involved.

The symbols need not be used for external communication. Meaning and understanding are often assumed (e.g. [7]) to be essentially concerned with communication between language users. As argued in [13], this is a mistake, since understanding of an external language is secondary to the use of an internal symbolism for storing information, reasoning, making plans, forming percepts and motives, etc. This is prior in (a) evolutionary terms, (b) in relation to individual learning, and (c) insofar as the use of an external language requires internal computations. In short:

'Representation is prior to communication'.

Objects in the world W may be concrete (e.g. physical objects) or abstract (e.g. numbers, grammatical rules). They may be external, or internal to U. Like symbols, the objects may exist in a *virtual* world, embodied in a lower level world, like a virtual machine implemented in a lower level computer. Many programming languages refer to objects in a virtual world, such as lists, arrays, procedures, etc. Similarly social systems form a virtual world embedded in a psychological and physical world.

The structure of the concept ‘understanding’

Instead of fruitlessly trying to identify a set of defining conditions for U to use symbols with understanding, I offer a prototypical set of conditions for saying that U uses some collection of symbols as a language L referring to objects in a world W. Different combinations of these conditions define different concepts of ‘language’, ‘meaning’, ‘understanding’, etc. Asking which is the ‘RIGHT’ concept is pointless.

For each condition I comment on how it might be satisfied by a machine, ignoring, for brevity, the difference between internal and external representations of computer languages. The discussion will appear to be question-begging, as fragments of evidence will be presented as if the case had been made. The fact that so many fragments can be presented this way, is what makes the case! It shows that events and processes in a machine can constitute a model for a significant subset of the ‘axioms’ implicitly defining mentalistic concepts. Unlike simulations of (e.g.) tornadoes, people outside the model can relate to the model as to the real thing (though some may find this distasteful). A robot may obey commands, answer questions, teach you things. But a simulated tornado will not make you wet or cold. Anyone who objects that this is not enough can be challenged to describe precisely what is missing. Appeals to mystery, or to unanalysable kinds of mental or spiritual stuff are undiscussable.

We’ll see that computers can manipulate internal structures and use them as symbols associated with a world W consisting of both entities within the machine and more abstract entities like numbers and symbol-patterns. Later, the discussion addresses reference to an ‘external’ world.

Prototypical conditions for U to use L to refer to W

* L is a set containing simple and complex symbols, the latter being composed of the former, in a principled fashion, according to syntactic rules.

This condition is satisfied by most computer languages, though machine codes generally have very simple syntactic rules and structures. Rules may be implicit in procedures.

* U associates some symbols of L with objects in W, and other symbols with properties, relations, or actions in W.

A computer can associate ‘addresses’ with a world W containing locations in its memory (or in a virtual machine) and their contents and relationships. The symbols cause processes to be directed to or influenced by specific parts of the system. Some of the symbols specify which processes - i.e. they name actions.

Various sorts of properties and relations may be symbolised in a machine language, e.g. equality of content, neighbourhood in the machine, arithmetic relations, having a bit set, etc. Symbols indicating tests that produce a boolean result, name properties and relationships.

So, if U is a simple computer, the basic semantic relation is causal:

> 'S refers to O for U' =
>
> > 'S makes U's activities relate to or involve O',

where O may be an object, property, relation or type of action.

Instructions have imperative meanings because they systematically cause actions to occur. Roughly,

> 'S denotes action A to U' = 'S makes U do A'

Depending on how rich the language is, S and A may have independently variable components, e.g. object, instrument, manner, location, time, etc.

In computers imperative meaning is basic: even denoting expressions are often instructions to compute a value. This low level meaning depends on direct causal connections within the machine. Later we discuss non-imperative denotation.

* Some of the objects referred to in world W are abstract, like numbers.

Computers can use certain symbols to denote numbers because they are manipulated by arithmetical procedures and used as loop counters, address increments, array subscripts etc. Thus the machine can count its own operations, or the elements of a list that satisfy some test. The way a machine does this is typically very close to the core of a young child's understanding of number words - they are just a memorised sequence used in certain counting activities. So:

> 'S refers to a number, for U' =
>
> > 'S belongs to a class of symbols which U manipulates in a manner characteristic of counting, adding, etc.'

* What a complex symbol S expresses for U depends on its structure, its more primitive components and some set of interpretation rules related to the syntactic rules U uses for L. ([5])

This is true of many computer languages. E.g. what is denoted by a complex arithmetical expression, or a complex instruction, depends on what the parts denote, and how they are put together according to the syntactic rules of the language.

* U can treat the symbols of L as 'objects', i.e. can examine them, compare them, change them, etc., though not necessarily consciously.

This applies to computers. Symbolic patterns used to refer can also be referred to, compared, transformed, copied, etc. E.g. two patterns may be tested for equality, or overlap, or set inclusion. An address can be incremented to get the next location. It is not clear whether other animals can or need to treat their internal symbols as objects. This may be a pre-requisite for some kinds of learning.

* Certain symbols in L express conditionality.

This is the key to much creative thinking or planning, and to flexibility of action. We can distinguish (a) ‘if’ used in conditional imperatives, (b) ‘if’ used as the standard boolean (truth-functional) operator and (c) ‘if’ used in conditional assertions. (c) is not found in the simplest computer languages.

Conditional imperatives are found in machines since ‘if’ (or some equivalent) when combined with evaluable expressions permits or suppresses actions, depending on the evaluation.

* By examining W, U can distinguish formulas in L that assert something true from those asserting something false.

Computers typically use symbols to denote ‘truth-values’ (‘true’ and ‘false’ or ‘1’ and ‘0’). Boolean operations e.g. ‘or’, ‘and’, ‘not’ are also represented, by symbols that trigger actions transforming inputs to outputs consistently with truth-tables. The ‘result’ is taken as a truth-value partly because of its role in conditional imperatives. The sense in which computers can examine their internal states to assign a truth-value is fairly clear, though how they check arithmetical statements requires deeper analysis.

If U assigns truth-values to symbols in a manner that depends on the state of world W, the symbols can be thought of as representing factual propositions, that so and so is the case in W. More generally,

> ‘For U, S means P is the case’ =
>
> > ‘in certain contexts the expression S causes U to do certain things only if P is the case, otherwise not’

We have yet to see how a machine can treat ‘true’ and ‘false’ as more than just formal duals.

* U can detect that stored symbols contain errors and take corrective action, e.g. noting that two descriptions are inconsistent and finding out which to reject.

Something like this occurs in programs that attempt to eliminate wrong inferences derived from noisy data, e.g. in vision, and in plan-executors that check whether the assumptions underlying the current plan are still true. Here we find support for a richer conception of a truth-value than just a pair of arbitrarily chosen symbols, if ‘true’ connotes surviving tests, and ‘false’ rejection. More on this later.

* A complex symbol S with a boolean value may be used for different purposes by U, for instance: questioning (specifying information to be found by lookup, computation, or external sensing), instructing (specifying actions), asserting (storing information for future use).

We have seen how, in a computer, S can function as a primitive question, in a conditional instruction where action depends on the answer to the question. In low level machine languages there is not usually the possibility of using the same symbol to express the *content* of an imperative as in “Make S true”. I.e. machine codes do not have ‘indirect imperatives’ with embedded propositions. However, AI planning systems have shown how in principle this can be done, at least in simple cases, assuming the initial availability of direct imperatives.

Apart from a few exceptions like Planner, Conniver and Prolog, most computer languages include requests and instructions, but not assertions: factual statements assimilated to some store of beliefs. However, it is easy to allow programs to record results of computations or externally sensed data, or even results of self-monitoring. Recomputable information may be stored simply for easy access, as people store multiplication tables.

Whether U uses S as a question, an assertion, or an instruction, will depend on context. S may specify the content of an assertion in one context (‘store(S)’), a question in another (‘if S then...’ or ‘lookup(S)’), and an instruction in a third (‘achieve(S)’). I.e. role is determined by *use* rather than form or content.

* U can make *inferences* by deriving new symbols in L from old ones, in order to determine some semantic relation (e.g. proofs preserve truth, refutations demonstrate falsity).

Work in AI has demonstrated mechanisms for doing this, albeit in a restricted and mostly uncreative fashion so far. Human forms of inference require some of the functional architecture discussed below in connection with motives, and also require use of a much wider range of representations than AI has so far addressed ([15]).

* L need not be a fixed, static, system: it should be extendable, to cope with expanding requirements.

One source of language change in people is communication with others using different dialects. A deeper source is situations that prove hard to describe.

Many computer languages are extendable. Adaptive dialogue systems are beginning to show how a machine may extend its own language according to need. But deep concept formation is still some way off. It is not clear which animals can and which cannot extend their internal languages. Without this, certain other forms of learning may be impossible. (More on language change below.)

* U may use symbols of L to formulate goals, purposes, or intentions; or to represent hypothetical possibilities for purposes of planning or prediction.

Simple versions of this sort of thing are found in existing AI planning systems.

Without a functional architecture supporting distinctions between beliefs, desires, plans, suppositions, etc., a machine cannot assign meanings in the way that we do. Merely storing information, and deriving consequences, or executing instructions, leaves out a major component of human understanding, i.e. that what we understand *matters* to us. For information to matter to a machine it would have to have its own desires, preferences, likes, dislikes, etc. This presupposes that there are modules whose function is to create or modify goals - motive generators. Full flexibility requires motive-generator generators. Deciding and planning require motive comparators and motive-comparator-generators. This is a complex story, spelled out in a little more detail in [14]. When desires, intentions, plans, preferences, etc. are generated through experience, perhaps over many years, this undermines the claim that a machine can exhibit only desires of the programmer or user. Such a machine, unlike existing computers, would use symbols in L for *its* purposes.

This takes us across yet another boundary in the space of behaving systems. Does a machine 'REALLY' understand without all this? Well, it could 'understand' well enough to be an utterly slavish servant. It could not, however, be entrusted with tasks requiring creativity and drive, like managing a large company or a battle force, or minding children.

* A language may be used for communication between individuals. This adds new requirements [18]), which are irrelevant to our present concerns.

Recapitulation

All the conditions so far listed for U to use a language L in relation to a world W are consistent with U being a computer. Several do not even require AI programs, since modern computers are built able to use symbols to refer to a world W containing numbers, locations in memory, the patterns of symbols found in those locations, properties and relations of such patterns, and actions that change W.

Associations between program elements and things in the computer's world define a primitive type of meaning that *the computer itself* attaches to symbols. Its use of the symbols has features analogous to simpler cases of human understanding, and quite unmatched by filing cabinets. So, it does not interpret symbols merely derivatively: the causal relations justify our using simplified intentional descriptions, without anthropomorphism.

Reference to inaccessible objects

We have seen how machines can refer to their own internal states, to numbers, and to symbolic patterns, i.e. what Woods [18] calls a 'completely accessible' world. In order to be useful as robots, or friends, they will need to refer to external objects, events, locations, etc. The problem of *external* semantic linkage is harder to deal with. Can a system use symbols to describe objects, properties, and relationships in a domain to which it has no direct access, and only incomplete evidence, so that it can never completely verify or falsify statements about the domain? (Compare philosophers on unobservables in science, e.g. [8])).

A key idea is that implicit, partial, definitions (e.g. in the form of an axiom system) enable new undefined concepts to be added to a language. (Compare [1]) on 'meaning postulates'. Woods' 'abstract procedures' seem to be the same thing.) For instance, a collection of axioms for Euclidean geometry, in the context of a set of inference procedures, can partially and implicitly define concepts like 'line', 'point', 'intersects', etc. The axioms constrain the set of permissible models. Similarly, a congenitally blind person may attach meanings to colour words not too different from those of a sighted person, because much of the meaning resides in rich interconnections with concepts shared by both, such as 'surface', 'edge', 'pattern', 'cover', 'stripe', 'harmonise', etc.

We can generalise this. In A.I. vision programs, instead of assertions and inference rules we often find data-structures and procedures for manipulating them. If the structures are also used to guide actions and predict their consequences, that implicitly gives them semantic content, by constraining the class of possible environments that could coherently close the feedback loops, just as a set of axioms restricts the set of possible models. As with axioms, the constraints may not define a unique model.

Causal embedding in an environment

Does external reference require external causal links? One may be able to use sensors detecting light, sound or pressure from external objects, and mechanical devices that act on objects. But direct links are often not possible. For instance we can refer to events remote in space and time, and even to hypothetical objects in hypothetical situations. So direct causal connections to X are not necessary for reference to X.

Causal links may differ in kind. Consider two machines running programs P1 and P2, the former connected to TV cameras and mechanical arms, as well as a VDU, and the latter only to a VDU. Suppose P1 is able to use its sensory-motor links in referring to the external world, and P2 contains all of P1 except portions of the program required for interacting with the cameras and arms. P1 can learn about the world either through its cameras, or from another agent through the VDU. P2 has only the VDU, but can think about the same world, like a blind and paralysed person who can talk and listen; and like paleontologists talking about pre-history. Causal links can be more or less direct, and can convey more or less rich information. Communication via another agent is indirect, and generally provides limited but abstract information, but it is still a causal link, like fossil records.

So, using symbols to formulate descriptions of an external world does not require that the world actually be directly sensed and acted on by the specific symbol-user, though the internal symbols and procedures must be rich enough to support such processes. However, some causal link is required if symbols are to refer to particular physical objects, like the Tower of London, or physical properties found in our world, such as magnetism. Without causal connections with the environment a thinker could only think (existentially quantified) thoughts about an abstract possible world, perhaps a generalisation of our world, but not about *this* world, or things in it. Causal links, whether via sense organs or other agents, can help to pin the reference down to this world. They can reduce the extent of ambiguity of reference, though they never totally remove it, as shown by old philosophical arguments in support of scepticism (see Strawson).

Extending ‘mentalese’: concept learning

A language may be extended by the addition of new axioms and procedures, partially and implicitly defining some new primitive symbols, and modifying the meanings of old ones. The history of concepts of science and mathematics shows that not all newly-acquired concepts need be *translatable* into one’s previous symbolism. E.g. ‘mass’ in Einstein’s physics is not definable in Newtonian terms. Physicists use concepts not explicitly definable in terms of tests that may be applied to sensory data. Using theories and inconclusive tests, they infer descriptions including symbols that are only partially defined. An intelligent machine or organism is in the same sort of relation to the world as is a scientific community.

So new symbols may be learnt without being *translatable* into old ones. After such learning, there is no clear functional distinction between the original concepts and the accreted language: we can memorise facts, formulas and instructions in English, instead of always having to translate into ‘mentalese’. Hence, contrary to Fodor, different humans (or machines) may use different ‘mentalese’ even if they all started off the same.

The essential incompleteness of semantics

Not *every* descriptive or referential symbol U understands must be one to which U can relate reality *directly,* using perceptual or other causal links. The symbol-system L may make contact with reality, e.g. through U's sense-organs and actions, only at relatively scattered points, and only in indirect ways (like the connection between reality and our concepts of 'atom', 'the remote future', 'another person's mind', 'Julius Caesar', 'the interior of the sun', and so on). People with different points of contact with reality store much the same general information about large chunks of the world, because their inference procedures permit them to extrapolate beyond what they have already learned, and we very likely have biological constraints built into us that, together with social processes, lead us to similar extrapolations from fragmentary evidence. However, convergence is clearly not guaranteed, and its absence may go undetected for some time [9]. If machines are to communicate successfully with us, the designers will have to understand these constraints and how they work.

If a new symbol is introduced using axioms that partially implicitly define it, then it can only be used with a partial meaning, and sentences containing it will not have determinate truth- and falsity-conditions. Such meanings may be inherently incomplete, if the concepts are indefinitely extendable by adding new theoretical assumptions about the nature of the reality referred to. This incompleteness is evident in theoretical concepts of science, but can also be demonstrated in ordinary concepts. This is an inevitable fact about the semantics of a language used to represent information about external objects, concerning which only partial, inferred, information is available, via sense organs, instruments, hearsay, books, fossil records, etc. In a sufficiently complex system, even the language used for describing its own *internal* state will have this kind of indeterminateness and incompleteness, because of the problems of internal access sketched in chapter 10 of [12].

How can truth and falsity be distinguished?

Although I have shown that computers can be said to use boolean operations and boolean values, it is not clear how to distinguish a 'true' from a 'false' boolean value, since their roles in a computer may be totally symmetrical. The manual may say that 1 stands for 'true' and 0 for 'false', and that certain symbols are interpreted as 'and', 'or' 'if', etc. But the duality of propositional logic implies that there is as much basis in the formal manipulations for treating 1 as 'false' and 0 as 'true', 'and' as 'or', 'or' as 'and' and 'if' as 'unless'. What else is required for there to be an asymmetry between the symbol for 'true' and the symbol for 'false'?

Assertions can be stored, but mere storage does not introduce an asymmetry between 'true' and 'false', since false as well as true statements could be stored, with explicit boolean indicators, or in different data-bases.

In Prolog-like languages, it might seem that there is a clear distinction between truth and falsity, between 'and' and 'or', and so on, with completed derivations signifying truth, failure signifying falsity. However, this is not sufficient to distinguish truth and falsity, since proving conclusion C on the basis of premisses P1 to Pn is equivalent to refuting the disjunction of P1 to Pn on the basis of the falsity of C.

We have seen one source of asymmetry, in mechanisms that can check stored assertions out, instead of always blindly assuming them correct: an elementary form of self-consciousness. Truth of a formula is then associated with having the capacity to survive thorough checking. But the connection is not simple, for the process of checking may include errors.

Another source of asymmetry is a ‘redundancy convention’. Instead of storing values of expressions explicitly, adopt a convention that one of the boolean indicators is redundant: it is signified merely by the presence of a formula in an information store or a communication. ‘True’ and ‘false’ then drop out of the ‘object language’ and become partly redundant metalinguistic concepts.

A deeper asymmetry lies in connections between beliefs and autonomous motives. Truth then is the boolean value of those beliefs (stored information) which (generally) enable desires to be satisfied by rational planning. Again the connection is not simple, for a true belief combined with other false premisses, or an invalid inference, can lead to a disastrous plan. Moreover, what fulfiles one desire may turn out to subvert another far more important one. I believe that further investigation will show that by adopting the design stance we can replace old and apparently empty philosophical disputes with new fruitful analyses with important implications for the design of intelligent systems.

Conclusion

By adopting a ‘design stance’, we can begin to clarify the question whether machines themselves can understand symbols, or whether meanings of symbols in a computer are only derivative. It is not enough that machines appear from the outside to mimic human understanding: there must be a *reliable* basis for assuming that they can display understanding in an open-ended range of situations, not all anticipated by the programmer. I have briefly described structural and functional design requirements for this, and argued that even the simplest computers use symbols in such a manner that, independently of how PEOPLE interpret the symbols, the machines themselves (unlike cabinets and copiers) associate meanings of a primitive sort with them. Internal uses of symbols are primary.

I have shown that a machine may use symbols to refer to its own internal states and to abstract objects; and indicated how it might refer to a world to which it has only limited access, relying on the use of axiom-systems to constrain possible models, and perception-action loops to constrain possible completions. These constraints leave meanings partly indeterminate and indefinitely extendable. Causal links reduce some of the indeterminacy. (All these topics require far more detailed discussion.)

The full range of meaningful uses of symbols by human beings requires a type of architectural complexity not yet achieved in AI systems. There is no known obstacle to such developments in principle, though further research may reveal insuperable difficulties.

Instead of listing necessary and sufficient conditions for understanding I argued that there is a complex set of prototypical conditions, different subsets of which may be exemplified in different animals or machines, yielding a complex space of possible systems which we are only just beginning to explore. Our ordinary concepts, like ‘understanding’ are not suited to drawing global boundaries within such a space. At best we can analyse the implications of various different designs, and the capabilities they produce, or fail to produce.

When we have shown in detail how like or unlike a human being some type of machine is, there remains a residual seductive question, namely whether such a machine really can be conscious, really can feel pain, really can think etc. Pointing inside yourself at your own pain (or other mental state) you ask ‘Does the machine really have THIS experience?’. This sort of question has much in common with the pre-Einsteinian question, uttered pointing at a location in space in front of you: ‘Will my finger really be in THIS location in five minutes time?’ In both cases it is a mistake to think that there really is an ‘entity’ with a continuing identity, rather than just a complex network of relationships. The question about machines has an extra dimension: despite appearances, it is ultimately an *ethical* question, not just a factual one. It requires not an answer but a practical decision on how to treat the machines of the future, if they leave us any choice.

Acknowledgements

The author has a fellowship from the GEC Research Laboratories, and has benefitted from discussions with members of and visitors to the Cognitive Studies Programme at Sussex University, especially Margaret Boden, Steve Torrance, and Bill Woods.

BIBLIOGRAPHY

----

[1] Carnap, R., *Meaning and Necessity* Phoenix Books 1956.

[2] Dennett, D.C., *Brainstorms,* Harvester Press 1978.

[3] Evans, Gareth, *The Varieties of Reference,* Oxford University Press, 1982.

[4] Fodor, J.A., *The Language of Thought* Harvester Press 1976.

[5] Frege, G., *Translations from the philosophical writings,* ed. P. Geach and M. Black. Blackwell, 1960.

[6] Hempel, C.G, ‘The Empiricist Criterion of Meaning’ in A.J. Ayer (Ed.) Logical Positivism, The Free Press, 1959. Originally in *Revue Int. de Philosophie,_Vol.4.* 1950.

[7] Lyons, John, *Semantics* Cambridge University Press. 1977.

[8] Pap, A., *An Introduction to the Philosophy of Science* Eyre and Spottiswoode (Chapters 2-3). 1963.

[9] Quine, W.V.O., ‘Two Dogmas of Empiricism’ in *From a Logical point of view* 1953.

[10]Searle, J.R., ‘Minds, Brains, and Programs’, with commentaries by other authors and Searle’s reply, in *The Behavioural and Brain Sciences* Vol 3 no 3, 417-457, 1980.

[11]Searle, J.R., *Minds Brains and Science,* Reith Lectures, BBC publications, 1984

[12]Sloman, A., *The Computer Revolution in Philosophy: Philosophy Science and Models of Mind, Harvester Press and* The Humanities Press, 1978.

[13]Sloman, A., ‘The primacy of non-communicative language’, in *The analysis of Meaning: Informatics 5,* Proceedings ASLIB/BCS conference Oxford, March 1979, Eds: M.MacCafferty and K.Gray, Published by Aslib.

[14]Sloman, A. and M. Croucher, ‘Why robots will have emotions’ in *Proc. IJCAI* Vancouver 1981.

[15]Sloman, A., ‘Why we need many knowledge representation formalisms’, in *Research and Development in Expert Systems,* ed M. Bramer, Cambridge University Press, 1985.

[16]Sloman, A., ‘Strong strong and weak strong AI’, *AISB Quarterly,* 1985.

[17]Strawson, P. F., *Individuals: An Essay in Descriptive Metaphysics,* Methuen. 1959.

[18]Woods, W.A., ‘Procedural semantics as a theory of meaning’, in *Elements of discourse understanding* Ed. A. Joshi, B. Webber, I. Sag, Cambridge University Press, 1981.