FORM AND CONTENTS IN SEMANTICS

Yorick Wilks
1989
No. 56

WORKING PAPERS
Fondazione Dalle Molle
Form and Content in Semantics

Yorick Wilks

ABSTRACT

This paper continues a strain of intellectual complaint against the presumptions of certain kinds of formal semantics (the qualification is important) and their bad effects on those areas of artificial intelligence concerned with machine understanding of human language. The paper begins with a critical examination of Lifschitz' (out of McCarthy) use of epistemological adequacy. The paper then moves, rather more positively, to contrast forms of formal semantics with a possible alternative: commonsense semantics. Finally, as an in-between case of considerable interest, it examines various positions held by McDermott on these issues and concludes, reluctantly, that, although he has reversed himself on the issue, there was no time at which he was right.
1. Introduction

This paper is written from the point of view of one who works in artificial intelligence (AI): the attempt to reproduce interesting and distinctive aspects of human behavior with a computer, which, in my own case, means an interest in human language use.

There may seem little of immediate relevance to cognition or epistemology in that activity. And yet it hardly needs demonstration that AI, as an aspiration and in practice, has always been of interest to philosophers, even to those who may not accept the view that AI is, essentially, the pursuit of metaphysical goals by non-traditional means.

As to cognition in particular, it is also a commonplace nowadays, and at the basis of cognitive science, that the structures underlying AI programs are a guide to psychologists in their empirical investigations of cognition. That does not mean that AI researchers are in the business of cognition, nor that there is any direct inference from how a machine does a task, say translating a sentence from English to Chinese, to how a human does it. It is, however, suggestive, and may be the best intellectual model we currently have of how the task is done. So far, so well known and much discussed in the various literatures that make up cognitive science.

My first task in this paper concerns epistemology but in a rather narrow way and does not directly address the large topics I have named above. It is to observe and criticise the fact that one school of AI researchers has, in effect, hijacked the word “epistemology” and used it to mean something quite unrelated to its traditional meaning: the study of what we know and how we know it. The term has been used within the ongoing dispute in AI about how we represent knowledge (facts, generalizations, performances, etc.) in AI programs so that machines can be said to know things, or rather, how they can be programmed so as to perform as if they know things, such as telling you, about the trains to Washington at a station where you type a question to a publicly-available computer.

The AI researchers who use the word “epistemology” are part of what is frequently call the “Logic Approach to AI”: the claim that the representations required by the task just mentioned are those of first-order predicate logic, and its associated model theoretic semantics. My overall task in this paper is to examine that use of “epistemology” and then move to a criticism of that whole tradition of logic-based-representation in AI. My own view is that we do need representations (as opposed to the current trend of connectionism (e.g., Smolensky 1988, Waltz & Pollack 1985) who deny that), but that their form, if interpretable, is largely arbitrary, and we may be confident it has little relationship to logic. I shall restate the view that the key contribution of AI in unravelling how such complex tasks as “understanding” might be simulated by a machine lies not in representations at all but in particular kinds of procedures (that much at least, my view shares with connectionism). It would be the most extraordinary coincidence, cultural, evolutionary, and intellectual, if what was needed for the computational task should turn out to be formal logic, a structure derived for something else entirely. Although, it must be admitted, strange coincidences have been known in the history of science.
The view under criticism here, then, the "Logic Approach to AI", is not merely being accused of misusing a word ("epistemology"), nor of getting its representations wrong. Its whole larger dream is under attack: that concentrating on the deductive relations of propositions can yield a theory of mind or even machine performance.

There is a well-known tradition, going back to Plato at least, that what we can be said to know includes the deductive consequences of other things we know. But that inquiry alone has never been thought to yield a theory of mind, or an epistemology. If anything in AI were ever to bear on, or contribute to, the study normally indicated by that tired old word, it would surely come when a computer not only behaved as if it knew things, (and in a complex, coherent way, not as a recitation of facts) but could also relate them directly to its own physical manipulations, as we can. Moreover, it might (so as to qualify) also have to come to know new things under roughly the conditions we do, and, I suggest, not know things that we cannot in principle know, such as aspects of our internal functioning at the level of the brain (see Wilks 1984a).

All these possibilities for a serious "electronic metaphysics" are a long way off. My task here is not to advance such endeavors but is merely domestic, perhaps only housecleaning, by showing that the AI uses of "epistemology" and the "Logic Approach to AI" have nothing to do with that task.

2. Epistemological adequacy and the logic-based approach to AI and natural language processing.

Let us advance scholarship in a video age by hunting our intellectual quarry on videotapes, rather than in journals, though serious scholars will find the distinctions discussed here also in (McCarthy 1977). Lifschitz (1987) in setting out what he calls the Logic Approach to AI distinguishes epistemological and heuristic adequacy as follows:

An **Epistemological adequate** model is one such that a solution to a given problem follows from the model.

A **Heuristically adequate** model is one which provides, in addition, a method for finding that solution.

This he explains best in terms of chess, where the rules of chess give a solution to a board problem, and hence form an epistemologically adequate model; but only additional heuristics give an effective solution, if one is available.

The last phrase is crucial since Lifschitz's illustration ignores a crucial fact about chess: that the game is decidable only from a certain range of positions, usually known as "saddle points" (Botvinnik 1971), and hence, in no serious sense, are the rules of chess an epistemologically adequate model on Lifschitz's own definition. But let us treat that point as mere carping, even though its flavor will continue to permeate our discussion of those key notions and their impact and significance for the representation of knowledge in AI.
Lifschitz's distinction, under various names and guises, has been central to the "Logic Approach to AI" since the publication of McCarthy and Hayes (1969). My concern in this paper will be with its impact on the area of processing in AI that I know best, namely natural language processing. But that will not be merely an arbitrary focussing of this paper, within the general area of epistemology and cognition, for two reasons.

First, the areas of natural language representation and processing have been major ones in recent years for developments of formal representations (under influence from the McCarthy & Hayes work in AI, and its successors, and from the work of Montague (1974) and its successors in formal linguistic semantics). Secondly, much of the work in the philosophy of language in this century has been to recapitulate, if not absorb, other areas of philosophy, such as the philosophy of mind. So then, to discuss the issue of epistemology from a language-oriented viewpoint is in no way revolutionary.

Lifschitz quotes Bundy approvingly to the effect that AI abounds in formalisms that are plausible but lack a proper semantics. This has been a familiar line in the McCarthy and Hayes tradition in AI: Hayes himself (1985) applied this stern medicine to a range of AI formalisms including that of the present author.

That, says, Lifschitz is what happens when you have only a heuristics, but do not have the epistemological part properly worked out. As we shall see, this last means no more or less than having a classic model-theoretic semantics, which has no natural relation to any normal meaning of "epistemological" or to "knowledge" at all.

Mere appeal to logic is not enough, he then warns us, since even some users of logic are guilty of insufficient attention to epistemological adequacy, and he cites negation-by-failure in logic prgramming as an example (one to which we shall return below).

Let us at this point pause and ask again what is this epistemological adequacy? It is still a fair question since, as we saw, the chess example, which was intended to explain the notion, actually raised as more doubts than it assuaged. Certainly, the notion has little to do with traditional epistemology: the part of metaphysics that deals with human knowledge, with what we know and how we come to know it and, in the British tradition at least, the intimate relation of those two. Lifschitz's epistemological adequacy (let us call it EA) has nothing to do with that: he never considers or displays any interest in what we know, its degree of certainty, its possible limitations, its contents, or how we come to know anything. If EA has philosophical antecedents, and its proponents usually assume this without any evidence, it is probably Plato, with his view that we know all consequences of what we know innately. But, again, Plato, unlike our AI colleagues, gave a great deal of thought to what and how we know innately, and it was that that made him an epistemologist.

Nor has EA anything to do with cognition, in the sense of the psychology of what we know or how we come to know it: in the chess example, a psychologist would focus almost entirely on what Lifschitz dismisses as the heuristics of the
game: the abstract sense in which the rules condition possible moves would probably be of very little interest to a cognitive psychologist. Unlike Chomsky in a similar dilemma over cognition and universal grammar, it will be difficult for Lifschitz to separate himself from these cognitive problems since he clearly does believe that, in some magical way, and independently of any evidence, logic is at the basis of cognition. I suppose this certainly is clearer, even if false, than Chomsky's (1965) resort to an inscrutable competence-performance distinction.

So, what is EA about? It is simply another way of putting clothes on the thin skeleton of model theoretic semantics, applied to areas that are not a prima facie appropriate for its attentions: common sense knowledge, natural language, even chess. We need logic, says Lifschitz, because the facts are logically complex. But that is precisely the point at issue: in natural language representation it is not agreed what the facts are, although it is agreed that, whatever they are, they are complex. But why logically complex? What possible evidence is there for that, given that it would be very foolish to deny, at this stage of AI research, that the "facts of language" can be represented on a (connectionist, non-logical) network of arbitrary complexity (cf Smolensky 1988). That is a representational claim; as to the corresponding cognitive claim, the disarray is even greater.

But let us stick to man-made domains that might seem to suit Lifschitz's case: chess, above all. EA for chess, as stated by him, establishes the opposite of what he intended; in that it cannot lead to a chess player, artificial or human. That must require heuristics, as practical research has amply shown. And, as I noted, chess does not even support the abstract claim, about derivability of positions by formal methods. What can the content of EA then be? I suggest, nothing: it is no more than a disguised statement of faith that a set of logical statements capture a situation and that a semantics will be made available at some point to give computational decidability to any putative consequence of the axioms and rules of inference. To know that some model is EA is to know just that, and yet can we be given some non-trivial domain for which that can be known of a model? If not, then all this may be magnificent, but it is not knowledge representation, let alone AI.

Let us stay, for a moment longer, with another of Lifschitz's chosen domains: the one of negation-by-failure, as implemented in standard Prolog. It is especially revealing for our efforts to find out what the real content of EA is.

Lifschitz maintains that negation as failure (which we shall refer to as NAF) is not EA, which might seem plausible if NAF meant something like an unsound method of reasoning in general, given that it is not always right to take something as true because we cannot prove it false. I cannot prove it false, on the basis of what I know right now, that it is raining in Sydney. But I feel no urge at all to assert that therefore it is raining in Sydney. I simply lack evidence: both the facts or plausible metereological generalizations and statistics.

But alas, it seems that Lifschitz means nothing so reasonable in his dismissal of NAF. NAF he says has no classical semantics, its semantics are in the form of a procedure, and we all know that procedural semantics is wrong. Let us go slowly here: this point of Lifschitz's, even if true, has nothing whatever to do with
epistemology in any standard sense nor, interestingly, has it anything to do with EA as he defined it, as it would if a solution to a problem would not follow in general from the facts of a model and a logical program using NAF, simply because NAF is a perfectly effective procedure within logic programming. It just doesn’t (always) fit commonsense intuitions.

The mask is off: the only argument Lifschitz has against NAF is its lack of a classical semantics, and having that feature is all that EA means for him. EA models cover simply and only what can be known by (semantically justified) deduction from assumptions whose knowledge status one is not concerned with, and here we note again that even the very deductively-oriented metaphysicians, like Descartes, did also worry a great deal about the status of the assumptions or first principles.

Lifschitz also concedes that a semantics has in fact been given for NAF (at least for “stratified” systems, though one can be provided for virtually any effective procedure, given the nature of human ingenuity) so now he has a real problem, in that he wants to reject NAF, presumably on the common-sense grounds given above, but cannot because now it meets the only criterion he has: having a classical semantics. For the moment I rest my case that EA, even if it has content, has nothing to do with epistemology, cognition, problem solving, or knowledge representation.

3. Formal vs. common-sense semantics

Let us now turn from a logic-based approach to AI in general, so as to contrast formal semantics, as a tool for natural language processing in particular, with a different approach: commonsense semantics.

There need be no real dispute here about what is meant, in the broadest terms, by formal semantics (FS) when opposed to common-sense semantics (CSS); commentators on the distinction as different as Israel (in press) and Sparck Jones (in press) broadly agree on where the line is to be drawn.

Formal semantics (henceforth FS), at least as it relates to computational language understanding, is in one way rather like connectionism, though without the crucial prop Sejnowski and Rosenberg’s work (1986) is widely believed to give to the latter: both are old doctrines returned, like the Bourbons, having learned nothing and forgotten nothing. But FS has nothing to show as a showcase success after all the intellectual groaning and effort. Here, I must register a small historical protest at Israel’s claim that “Traditionally (indeed, until Montague, almost undeviatingly) the techniques of pure mathematical semantics were not deployed for formal or artificial languages” (ibid.). It all depends what you mean by techniques, but Carnap in his Meaning and Necessity (1947) certainly thought he was applying Tarskian insights to natural language analysis. And the arguments surrounding that work, and others, were very like those we are having now. I need that point if the Bourbon analogy is to stick: FS applied to natural languages is anything but new.
But there have been recent changes in style and presentation in computation as a result of the return: many of those working in the computational semantics of natural language now choose to express their notations in ways more acceptable to FS than they would have bothered to do, say, ten years ago. That may be a gain for perspicuity or may be a waste of time in individual cases, but there are no clear examples, I suggest, of computational systems where an FS theory offers anything integral or fundamental to the success of the program that could not have been achieved by those same processes described at a more common-sense level (what I am calling common-sense semantics, or CSS). I would suggest that recent discussions in AI concerned with inheritance systems, in particular, tend to confirm my hunch on this issue, in that the useful ones at the moment, such as Touretsky’s, are CSS systems, and attempts to formalise systems fully has made them intractable. This FS-as-mental-hygiene defense will return later in the paper when discussing McDermott.

I do not at all intend, at this point, to define CSS by any particular type, or level, of notational description: I believe it is best exemplified by the role of semantics in natural language processing within AI, and by a primary commitment to the solution of the major problems of language processing: those problems that have obstructed progress in the field for thirty years. These I take to include: large scale lexical ambiguity in text (i.e., over realistically large ranges of sense ambiguity for lexical items of English), the problems of phrase and other constituent attachment, where those require considerations of meaning to determine, as well as the mass of problems that collect around the notions of expertise, plans, intentions, goals, common knowledge, reference and its relationship to topic etc.

On these descriptions of FS and CSS, they are not necessarily exclusive: it would be quite conceivable for an FS system to aid with the solution of a problem important to CSS. And it will always be possible for a successful CSS theory to be subsequently axiomatised. But that does not equate CSS and FS an more than axiomatising physics does away with experiments: theories come first, axiomatisations follow. In fact that axiomatization has not been done for natural language processing and there is no reason to believe it will, because the origins and ultimate preoccupations of FS are always elsewhere. The examples Israel chooses to discuss in detail (Henkin and Kripke) make my point exactly. He (ibid) notes that he could have taken others when defending FS and that "the nature of the languages studied makes no essential difference". But, as he seems to concede elsewhere in the paper, those are the very areas where the techniques can be shown to work and that is why they are always chosen ("the choice of tools.....should be grounded somehow in the nature of the phenomena to be analysed"). My case, to be set out in a little more detail below, is that the kind of the language chosen for analysis (natural versus artificial) makes as much difference as any choice could make, and that in the last quotation Israel is dead right, though not in the way he intended.

A small concrete example may help: the choice between generating "a" and "the" is notoriously difficult in English, one that non-native speakers continually get wrong. Examples are sometimes hard to grasp in one's own language, and the
choice between "des" and "les" in French is similarly crucial and notorious, though it is not the same distinction as the English one, yet it rests on the same kind of semantic criteria. It is not a problem with an arbitrary solution: French grammar books claim to offer principles that underlie the choice.

It also seems, on the surface, to be a problem that FS, or any logical approach to language structure, ought to help with: it is certainly some form of idiosyncratic quantification. Those particles are exactly the kind that Montague grammar, say, offers large, complex structures for, just as, as Sparck-Jones notes (ibid), such systems offer such minimal, vacuous, codings for content words. This is probably the clearest quantitative distinction between FS and CSS. As far as I am aware no FS solutions have ever been offered for these problems, nor would seem remotely plausible if they were; at best they would simply be a recoding in an alternative language of criteria satisfactorily expressed in other ways. Yet this is exactly the sort of place where FS should offer help with a concrete problem if it is to be of assistance to the NLP task at all, for it is to such items that its representational ingenuity has been devoted.

The historical and intellectual sources of FS lie in an alternative approach to what constitutes a proof: to meta-logical methods for the establishment of properties of whole systems, such as complete, consistent, etc., and the employment of properties such as decidability to establish the validity of theorems independently of normal proof-theoretic methods, i.e., by "semantic" methods, in that special sense. The applicability of this methodology has been perfectly clear in the case of programming languages, and to proofs of correctness of programs (even if the scope of the applications is still depressingly small) but in the application to natural language understanding its original aims have simply got lost.

From time to time, an application within the original methodology surfaces, such as Heidrich's proof (1975) of the equivalence of the methods of generative semantics and Montague grammar, but the result proved is then seen to be vacuous, in the terms of CSS at least, in that nothing was established whose absence had constituted any pre-existing problem. The usefulness or (in)adequacy of generative semantics was not anything that could be established or questioned by the sort of guaranteed equivalences that the proof offered. The problems with generative semantics, whatever they were, lay elsewhere and were not alleviated by any such proof.

The heart of the issue is good old decidability, or whether or not the sentences of a language form a recursive set in any interesting sense. It is clear to me that they do not, although I contrasted various senses in which they might back in 1971 (Wilks 1971). Contributions like Israel's only makes sense on the assumption that sentences do form some such set, unless he is adopting only a "descriptive logical language position" (see below position #1), and his whole position paper suggests he is doing far more than that.

The alternative position is that natural language is not a phenomenon of the sort required and assumed by the various systems of logic under discussion; and that the interpretation of a sentence in a context is an approximative matter (including whether or not it has a plausible interpretation at all, and hence whether
or not it is a sentence at all), one computed by taking a best-fit interpretation from among a number of competing candidates. That is not a process reducible to a decidable one in any non-trivial manner. Indeed, I recall going further in (Wilks, 1971) and arguing that that process of assigning an interpretation to a string, from among competing candidates, could be taken as a criterion of having a meaning: namely, having one from among a set of possible meanings. I mention this point for purely self-serving reasons: I do not want Israel to get away with identifying CSS with Schank (1975) and Chomsky as he does in his paper. I do not object to him joining them as bedfellows: that has been done before, and all serious opponents are said to share premises. In the case of those two, the similarities become clearer as time goes on; and include a certain commitment to genetic claims, but above all a commitment to representations rather than procedures. My self-servingness is to point out that my own approach, preference semantics, was not about commitment to representations of a particular kind at all, but only to certain procedures (based on coherence and connectedness of representations) as the right way to select interpretations from among competitors, for competitors there will always be. CSS can be about procedures as much as representations leave this. In his own commitment to representations, and dismissal of procedures ("semantics, even construed as a theory of language use, is not directly a theory of processing") Israel is actually in the same bed with that distinguished company. But not to worry, it is a big bed.

Preference semantics was, in a clear sense, procedural and had the advantage of declaring strings that did not admit the assignment of a single interpretation (i.e., remained ambiguous with respect to interpretation) as meaningless. Meaninglessness on that view was not having no meaning but having too many. I found, and still find that a satisfying position, one true to the process of language interpretation. Moreover, it also offers an opportunity, for computation, processing, artificial intelligence, or what you will, to have something to say about fundamental questions like meaning. It is clearly an assumption of Israel, and all who think like him, that that cannot be: "real semantics", as he puts it, is being done elsewhere.

It is one of the advantages of connectionist fun and games, from my point of view, that it has, against FS, brought implementation back to center stage from the wings; thus the new movement can be of enormous political interest and importance, whether or not one is a believer in it.

Let me try to separate levels (or perhaps just a continuum of positions, or aspects) that the claims of FS make about natural language processing:

1) There must be a certain style of formal description as the basis of any system of natural language description or processing. That this claim can be pretty weak can be seen clearly if we remember McCarthy's insistence on first order forms of expression combined with his advocacy of heuristics and indeed, his claim that heuristics constitute the essence of AI. One can acquiesce to this demand without giving any support to the central part of the FS claim about decidability.

2) The assumption of compositionality has been central to FS since Frege. From a computational point of view that doctrine is almost certainly either trivial...
or false. I am sure this has been said many times; but I discovered it in (Wilks 1984b), and it has been argued strongly by Schiffer (1987).

3) FS puts an emphasis on the particular role of quantifiers and the need of a field of distinguishable entities to quantify over (this is quite independent of #1, and more central to FS). The set of distinguishable entities is easily provided with labels like Lisp 'gensyms' and doing that in no way concedes the FS claim. It must be admitted that it was a notable weakness in some early CSS systems (e.g., Conceptual dependency or Preference Semantics) that they did not offer a clear identification of individual entities, independent of intensional codings of concepts. But that lack was easily remedied. The first demand can always be met by special quantifier procedures (e.g., Woods in his papers on procedural semantics (Woods, 1968)). Nothing more is needed and demonstrations like Montague's stock example of radical quantifier ambiguity ("Every nice girl likes a sailor") are effectively quantificational garden paths and no plausible natural language processing system is under any obligation to treat them. How could any system for the representation of natural language depend upon such cases, for they have no relation at all to crucial experiments in the sciences?

4) The relation of reference to the world: The claim of FS's capturing this relation is its weakest one, yet practitioners return to it repeatedly. If you do not adopt our methods, the claim is, you are trading in mere symbols, unrelated to the real world we, as plain men, know we share. I never cease to be amazed at the barefacedness of this: the classic statement of the position is David Lewis' attack on Fodor and Katz as peddlers of markerese (1972). But what else did he, or anyone else in FS, offer but symbol-to-symbol transformations? What else could they, in principle, ever offer by any conceivable formal methods? For symbols, and only symbols, are as much the trade and language of the denotational semanticist as of any computer modeller. Whatever the mysterious nature of the relation of symbols to things, it is not one on which FS could possibly throw light. Their solipsism is CSS's solipsism, and their position is metaphysically identical to ours.

Of course, arbitrarily named nodes identifying individuals are a handy, not to say essential, device, but no monopoly of FS. Worse yet, the proof procedures of FS demand such sets of entities to quantify over, but there is no formal way of guaranteeing that the entities established (so as to provide the guarantees that the proved theorems are true within the model that such entities form) are appropriate, in the sense of corresponding to any plausible real entities in the world. Any model set whatever that allows proved theorems to be true would suffice for the purposes of FS, a point that Potts (1976) among others has pointed out repeatedly. Proofs of program correctness have faced this problem, but FS applied to natural languages and common sense reasoning has not and cannot. Any claims to give access by such means to a plausible and appropriate world are not only false but utterly misleading as to the nature of FS.

What we reach by any formal or computational methods is always other symbols, and a "theory of meaning" for computational processes over natural language should recognise this fact. A suggestion along these lines was made in Wilks 1974 under a potentially misleading slogan "meaning is always, in fact,
other words", but sophisticated forms of that view are to be found in the work of Wittgenstein and Quine as well as, on a different cultural planet, Habermas. It was work of the last author that let Winograd to a version of this view in e.g. (Winograd 1985).

An important additional claim of FS, certainly made in Israel’s paper, is that semantics, whatever it is, must be separated off from knowledge of the world: "Lexical semantics does not yield an encyclopedia" and "Any plausible semantic account, then, will have to distinguish between analytic truths and world-knowledge". It is interesting to see that baldly and forthrightly stated without qualification, as if Quine were not still alive and well, but had never been. Practical experience with natural language processing always suggests the opposite: Sparck-Jones (ibid) shows in her paper how this borderline has to be fudged by peculiar means in certain FS approach to practical problems. Real lexicons are such that information about meanings and the world are inextricably mixed, or are simply alternative formulations (at least Camap, in opposing his "formal" and "material" modes of expression, got that right in 1947).

We should notice the repeated offer, to our sloppy and heuristic discipline, of a Real Serious Theory for the field (remember Chomsky’s similar offer in 1957 to machine translation and language well-formedness; and more recently FS’s for the semantics of selected AI systems). The chances always are that this prescription is unrelated to the disease; has Chomsky really helped computer language processing? We do indeed need a good theory but these are quack cures trading chiefly on the fears and inadequacies of practitioners and patients in the field.

One other consistent position is possible (I suspect it may be Dijkstra’s 1986): One can point out frequently, as he likes to do, the gap between the the interpretations normally given to logical entities (e.g., propositional implication or conjunction) and the interpretations given to apparently corresponding language items. One can also, at the same time, advocate the most stringent formal methods in computational applications to areas whose underlying structure or properties will support such methods. By such standards, natural language is not such an area and therefore one should not attempt this form of computational activity. That is a consistent position, but not one that those committed to the computer understanding of natural language can take. It poses no problem for CSSers, but does, I believe, put a serious choice before FSers who want to remain in some way relevant to language processing.

4. An interesting intermediate case: McDermott

Let us turn now to examine McDermott’s various positions on the logic approach to AI. My aim will be to show that maintaining proposition p at one time (that formal semantics is the proper basis of AI) and NOT-p at another is not, in itself, a guarantee of being right at least once.

All well-brought up children know that there is more joy in heaven over one sinner that repenteth, etc...but this is not heaven so I shall push ahead, uncharitable though it may be to do so. The plan will be as follows: first a discussion of
McDermott (1976) which has been reprinted many times and contains the germ of his logicism. Secondly, we examine McDermott (1978) where the espousal of Tarski semantics became explicit and, finally, (McDermott 1987) where his recantation was announced.

4.1. AI meets Natural Stupidity (1976)

Any writer in this field who has ever used the phrase "natural language" in a paper must have felt acute pain while reading McDermott's inspired Artificial Intelligence meets where he targeted Natural Stupidity (AINS for short); the quick and easy use of "epistemology" was also savaged, as was a whole mass of pretentious usage in AI and linguistics: particularly wishfulfillment programming: the naming of flow-chart modules with terms like UNDERSTAND. However, we should, in honesty, concede that that is very much the same point that Dreyfus made about AI for years: in particular about Bobrow's use of "understand" (1972, p.46).

But McDermott overstated his case at one point: when he attempts to determine what he calls the "intrinsic description" of a link in a semantic net, and thereby commits what I shall call the Gensym Fallacy. McDermott takes the IS-A link between FIDO and DOG, in a semantic network of the sort that is standard in AI, and says that its intrinsic description is "indicator value pair inheritance link" (AINS, p.5). His argument is that it is begging the question to call it IS-A because one sense of "is a" is what it is supposed to explicate. In the same vein he asks those who would use a node labelled STATE-OF-MIND to rename it GI073 and see if they still admire their system as much.

McDermott is clearly right that such systems have never, as yet, done much in practice, and the inflated naming of nodes gives a spurious satisfaction to the researcher involved. But the essence of the error is the failure to deliver, and the ability to fool oneself that one has delivered: it is certainly not node naming as such. If the system is bad then naming the node GI073 doesn't make it better, but is irrelevant. McDermott is slipping into the Gensym Fallacy: that everything is logically all right if names are all (arbitrary) number names. Shakespeare might not have been so pleased with one of his sonnets if the words had been named, in order, GI to GI40, but that proves nothing, and nor does the disillusion of the researcher who is forced to write GI073 instead of the more fulsome STATE-OF-MIND. He could still produce just as silly a program and make just as silly claims.

I think McDermott's error is to believe that there really are "intrinsic descriptions" and that we could use these innocently, in a way that wishful descriptions are not innocent. This is an error because our high level program items are not, and cannot be, purpose-free if we are to understand and communicate what we are doing. Suppose someone said that to call the truth table mapping that we could write by the vector (1 0 1 1) "material implication" was begging the question, and it should always be called "truth table mapping type 4" until shown to explicate implication. Sorry, "truth" begs the question too, so let's be even purer and say it
should be called "TF table mapping type 4". Communication would be impeded, would it not, and no one would have any idea of what the purpose of such descriptions, and the papers containing them, was? One could make just such a remark about material implication itself: in Schapiro (1976, p.5) quotes the great phrase of Anderson and Belnap where they write that "(material implication) is no more a kind of implication than a blunderbuss is a kind of bus".

McDermott has not followed through the logic of his own arguments and, were he to so, he would, in my view, be forced to choose one of the following more radical positions (in AINS his own general position is never explicitly stated, for the soft target shooting is much more fun):

1) There should be no perspicuous representation (meaning: easily interpretable with respect to some natural language, such as English) in AI natural language processing.

That is consistent with his view that "It seems much smarter to put knowledge about translation from natural language to internal representation in the natural language processor and not in the internal representation" (1976, p.6). On that view, the representation could be largely or wholly arbitrary. If he means largely, I agree with him for the reasons stated earlier; but if he means wholly arbitrary (and his use of Gensyms suggests that) then either he commits the Gensym Fallacy, or he is a closet connectionist, where implementation procedures are everything, and representations are totally unperspicuous, as their critics have claimed (Charniak in press). My intuition is that McDermott is no connectionist and that he believes symbolic but arbitrary representation can still be defended.

2) The interpretation of a representation, however arbitrary its symbols, can be provided by scientific procedures in some direct relation to the arbitrary items. No natural language need creep into this process.

I am sure that McDermott would be strongly tempted by that possibility, and it is, of course, no more or less than the grand design of the Vienna Circle: the Unified Science grounded in Protokölzaetze (Neurath 1932). That project, incorporating Carnap’s Logical Syntax of Language, was a clear precursor of both AI and Chomskyan linguistics.

But alas, it was all a magnificent mistake: there are no protocol, or basic, sentences and no scientist now believes such things, any more than he or anyone else should believe McDermott when he writes that "Eventually, though, we all trick ourselves into thinking that the statement of a problem in natural language is natural". Yes, we do and it is not a trick, and the collapse of Unified Science showed just that.

3) There is a representation with some degree of (non-arbitrary) interpretation and that can be provided by logic, properly conceived.

McDermott never states this view explicitly, but it is the ground of his later views and implicit in "Clearly, there must be some other notation, different in principle from natural language". This is an essential plank in the logicist case, and one for which there is no evidence. There is of course mathematics, but we are here discussing commonsense subject matter (love, life, chairs, tables, smart
weapons, cars and intentions) and not the domains of which mathematics treats.

My argument is that this choice is not really available, at least if that means an interpretation wholly unconnected to, and independent of, natural languages. A language of Gensyms without interpretation is as vacuous as any other calculus, logical or otherwise, without interpretation. Nor can "Unified Science" provide interpretations, and so bypass natural language considerations. Weighted networks may do so, but no one is very clear about that issue yet.

I am sure that (3) would have been McDermott's choice at the time he wrote AINS. He was led into confusion by his own assertion that "Many researchers tend to talk as if an internal knowledge representation ought to be closely tied to the structure of native language." He gave no references at that point to AI researchers, and it would be hard to do so. To the best of my knowledge I am the only person who has argued thus within AI and then only very surreptitiously, because it is so unacceptable (even if true).

McDermott probably believed when he wrote those words that Schank and the Yale School did believe that proposition, but he was simply confusing two things if he did. Schank always strongly denied that his representations were related, in their representational structure and primitive alphabet, to natural language, specifically to English. Unkind critics like myself repeatedly pointed out (e.g., Wilks in Charniak & Wilks 1976) that Yale representations did in fact have many residues of surface English, both in form and content.

But none of that is what McDermott claimed: that the researchers wanted that similarity. This is not mere nit-picking, but evidence of radical confusion in McDermott. He fails to distinguish:

   i) claiming one's representation is based on natural language,

   from

   ii) having one's representation interpretable only in terms of natural language.

If I am right, this last charge (ii) tells as much against McDermott as anyone, because there are no viable alternatives: a language of uninterpreted Gensyms or any logic of uninterpreted predicates is just gibberish, unless and until connectionism saves the day for us all. McDermott is as guilty as those he criticises: if there is a "natural language fallacy", he too must be committing it, unless he shows us a clear way out, and AINS certainly does not.

4.2. McDermott on Tarskian Semantics (1978)

In a later paper (1978 abbreviated as NSWD: No Semantics Without Denotation) McDermott more forcefully brings out the assumptions of the earlier AINS and ties them to the denotational semantic theories associated with Tarski. And yet, the doubts that overcome him later are already present, as when he writes: "the application of SD (systematic denotational semantics) in an informal way can still be valuable" (ibid, p. 278), a remark that only has sense on the assumption that the full rigorous application of the method will not be possible, as is indeed the case in AI and programming in general.
The bulk of NSWD is an informal examination of a small number of well-known AI systems such as Schank's, and the observation that some of the rules proposed for it are not generally true. Two most important points must be made here, one polemical and one substantial. First, even if it is of value to show that a rule in an AI system is not generally true, or is inconsistent with other rules, then that can always be seen simply by argument and careful observation, and with no call for the tools McDermott uses in NSWD. There is no sense in which the application of denotational semantics, formally or informally, helps one to see that: indeed, in the end it is known that there can be no such proof of inconsistency of any system of interesting richness, so the defect is more than practical. There is no clear relation between the disease (false rules, if that be a disease) and the remedy proposed.

Although NSWD is intended as a robust defense of denotational semantics in Artificial Intelligence it does contain an argument against its use:

"It would perhaps be surprising for an outsider to learn that computer scientists, in spite of the fact that they study purely formal objects like programs and data structures, have a pronounced 'anti-formalist' streak. This arose initially from the painful discovery that even the most formal objects have to be debugged." (ibid, p.8)

He is right about many of his colleagues, but, as ever, the reason given is the wrong one. The anti-formal "streak", insofar as there is one, is not about formalisms as such at all: programs must be totally formal or they do not run. The opposition, from papers like the present one, is to the application of a particular methodology—model theoretic semantics—and not at all on any ground to do with debugging, which is a pure red herring. It has to do with the application of computational methods to areas of AI like natural language processing where the formal structure of the area to be explicated does not support such methods.

This fact was touched on earlier, and this argument appeared in fuller form in Wilks (1971), but it can be set out again here in simple form. The argument is that the meaningful sentences of a natural language like English do not form a recursive or decidable set, and that fact makes any strong application of model theoretic semantics to, say, a natural language understanding system inappropriate.

The argument that natural language sentences are not such a set goes as follows: given any string of English words it can be rendered meaningful, and given an established use, by successive explanations in the way Wittgenstein (1953) constantly illustrated. It is merely a question of ingenuity and determination.

If that is the case, then there is no prior survey of the "theorems of English" in the way that there is for, say, the Propositional Calculus. In that formalism we knew, in advance of the production of a decision procedure (the truth tables in 1919), what was and was not a theorem of the Propositional Calculus. There were firm intuitions as to the truth of certain well-formed formulas, and well-established deduction procedures to establish others. The semantics associated with the calculus—the truth tables—then underpinned that "syntactically based" deduction with a decision procedure.
None of that is available for a natural language, and it is an essential feature of it that it is not. If it were, we would not be dealing with a natural language. Any fragment of, say, English that could be so axiomatised, would not then be a natural language precisely because the freedom to extend it, as we all do every day, would be gone.

What does it mean to claim that the prior survey of English sentences is not available? It is simply that we cannot construct lists of definite sentences and definite non-sentences in the way that we can for formal languages, and which we must be able to do if the Tarskian techniques devised to underpin theoremhood and deduction are to grip. Tarski himself, as we all know, did not believe natural languages were susceptible to these techniques, though that fact is in no way decisive.

What this claim does not mean is anything in particular about the intuition of what a meaningful sentence in English is, or how that is related to the meanings of its parts. There is nothing compositional about the point I am making. It does follow from this claim that an axiomatisation of English, if it were conceivable, could not be a good way to enumerate English sentences, for there would always be a denumerable infinity of English sentences it could not produce.

The position above, if true, has consequences for the semantics of internal representations, too, in spite of McDermott when he writes:

"The objection has been made that denotational semantics cannot be the semantics of natural language in all its glory. This may or may not be true...but has nothing to do with its use as a semantics of internal knowledge structures" (NSWD p.281).

But that is simply not so, and the consequence is fatal to McDermott’s whole case, given the non-recursive property of sets of sentences of a natural language, for that property will also be true of the formulas of the internal representation if there is any kind of straightforward mapping between them. Human beings provide that mapping (on any kind of “internal representation” theory of a mental processing) as do parsers between computer representations and sentences. So McDermott’s point fails unless the natural language and the internal knowledge structure are independent. They are not and McDermott has never suggested for a moment that they are.

This point can even be strengthened in the following way: let us make the fairly sensible assumption (to all except adherents to “fuzzy logic”) that quantification is a discrete phenomenon, as regards any semantics to be given for its appearance in a natural language or in an internal representation language. Now, if the semantics given for quantified formulas of an internal representation language is a Scott–Strachey semantics for programs (and in AI that might seem a natural additional assumption), then that semantics requires phenomena to be continuous overall (Scott & Strachey 1971). This fact can be interpreted in a number of ways with respect to the discussion so far, but the most rational is to take it as evidence that a program semantics cannot be given for internal representations of at least one basic aspect of natural language.
That conclusion may be too facile, however, given the possible interactions of the assumptions required to reach it. But on any manipulations of those, the conclusion tends to tell against McDermott’s assertion in the last quoted passage, namely, that we might reasonably expect a more rigorous semantics for an internal representation than for a "decoupled" and less rigorous natural language, one that corresponded to internal expressions in some one-to-one or many-to-one manner (over sets of sentences or formulas).

4.3. The Critique of pure reason (1987)

In his most recent work (abbreviated CPR) McDermott has seized on a Kantian title to withdraw much of the content of AINS and NSWD, although he does it not by withdrawing earlier arguments, but by attacking other logicist positions and drawing new conclusions inconsistent with those of the two earlier papers. The burden of the paper is that

i) in understanding, including the understanding of natural language, many or most inferences are non-deductive.

ii) deductive techniques (and hence the formal semantics machinery that underlie them) have no role in assessing confirmation of belief, which is a quantitative matter.

As the author of a paper in 1973 called "Understanding without proofs" (one which drew attention to Hume’s point that the proofs of non-mathematical conclusions are short and non-deductive, a claim McDermott attributes to Pat Hayes!) I can only welcome such conversion, while trying to keep at bay the slightly sour response of the lifelong tee-totaller welcoming recently reformed drunks.

The second point carries a strong flavour of discussions of connectionist computation, prevalent everywhere in artificial intelligence at the moment, and might yet lead to a further radical shift in McDermott’s position, one hinted at also in:

"What we now conclude is that content theories are of limited usefulness in the case where the contemplated inferences are non-deductive"

(CPR p.14).

Connectionism is, of course, almost by definition, a non-content theory, as all its critics have pointed out in their different ways. What McDermott means, in this last quotation, is to be understood in opposition to a clear case of a content theory: logic programming, and the belief associated with it that the content of our knowledge would be simply written down in an appropriate formalism, a position of McDermott himself at one time, and close to what we earlier called the classic logicist position of McCarthy and Hayes.

There is a small but crucial difference here between the content-theory (in the sense of procedure-free logic programming) on the one hand, and the older logicist program in AI, on the other. The logicists, like McCarthy and Hayes, never thought that what they were advocating was procedure free: it naturally required some set of trusty machine logic procedures. So, too, of course, does logic programming, even though they are hidden tidily away in the Prolog interpreter.
McDermott himself draws on this very distinction between logic and logic programming for a different purpose, but fails to see that it undermines his discovery (appearing just before the last quotation) that "there is no way to develop a 'content theory' without a 'process model'" (ibid).

But that is as true of classical machine logic as of any other part of AI, and the logicians knew it. In itself, it constitutes no reason at all to flirt with procedural (as opposed to denotational) semantics for knowledge representations. I write that as one who does firmly believe (Wilks 1981) in a procedural semantics, whatever it may turn out to be. My point here is that McDermott's reasons are bad ones for shifting from a wrong view to a right one.

That is equally true of his discovery of (i) above, construed as a claim about human psychology. It may be true, as Russell once said, that no one has ever performed a useful or practical deduction on any serious topic, but he did not allow that to interfere with his technical work. Even if true, it says nothing about how research in machine logic should proceed, nor about whether formal deductions can be produced to cover the inferences humans make. It has always been an assumption in traditional logic that any enthymeme (inference or truncated deduction) can be made deductive by the addition of suitable additional premises, and logicians could always provide these by, as it were, inspection.

It is a open question whether a machine logic can locate non-trivial assumptions, in general, so as to produce a consequence that is deductive. By "non-trivial" there, I mean: other than an assumption of the form p --> q, where p is a conjunction of the existing assumptions and q the consequent. The logicians continue with their program in the belief that it can be done but, whatever is the case there, McDermott's observations like the following do nothing to throw doubt on that program:

"But many inferences are not deductive. If I come upon an empty cup of soda pop that was full a while ago, I may infer that my wife drank it, and that's not a deduction...but an inference to the best explanation (The only way to mistake this for a deduction is to mistake logic programming for logic...)" (ibid, p.8).

But logicians know all this, and that the inferences can be made deductive, by trivial methods if necessary. One of these would be to have an assumption equivalent to: "if this cup, situated in my house, which was full at t0, is empty at t1 (where t0 precedes t1), and if my wife was in the house at some time between t0 and t1, then she drank it." It might well be argued at this point that the discovery and justification of such an assumption is what is traditionally called induction, rather than deduction. Logicians in the AI tradition normally make no distinction between those unless they also chose to call their work "machine learning", which is an area that seeks to show that perfectly formal accounts can be given of non-deductive inferences (e.g. Michalski 1976). If that is what McDermott is drawing attention to with the nice remark about logic versus logic programming, then so be it, but I suspect he is not, since induction is not a notion that features in his work. Unless he is prepared to offer some alternative account under such a heading, and I suspect he is not, then nothing follows from these observations that the logicians are not already fully aware of, and they cannot of themselves throw any general
doubt on the logicist program for AI (as opposed to say, one for psychology).

The bulk of McDermott's CPR consists of technical criticisms of devices in recent AI logic (non-monotonic, circumscription, etc.) that have been produced to support, and develop, the logicist case. The criticisms are admirable in themselves, interesting largely because of the past views of the author, and I have no wish here to dispute either their detail or general thrust. They are in fact unimportant for the purpose of this paper—other than their intense biographical interest, of course—since the criticisms do not bear directly on the general issues of principle raised in the paper. As I just noted above, any logicist can say of these criticisms that they are just technical details being fixed, just as he can say that diagnosing most human inferences as enthymemes, or incomplete deductions, in no way bears on the logicist program or on the general viability of machine deduction.

There are, I believe two casual remarks in CPR where McDermott moves close to the real problems for the logicist program: of principle not detail. They are:

(a) when he notes (p. 2): "The notation we use must be understandable to those using it and reading it".

(b) when writing of non-monotonic logic (p. 6): "Either theories like this don't have theorems, in which case they can't serve as the idealized inference engines we are seeking; or we are stuck with a weak notion of theorem,...".

The first remark is interesting in that, although implicit, it is one of the few direct withdrawls of a claim in one of the earlier papers: it is clearly a withdrawal of the whole line of argument in AINS that terms in the knowledge representation should be replaced by (inscrutable) Gensyms, so as to give moral health to a program description by removing the overtones imported from natural language.

It is obvious to anyone with experience in writing programs in high-level languages this that cannot be done or, if it can, it removes the point of using such languages in the first place. McDermott does not note the significance of such a retraction but, in the light of the previous discussion in this paper, it should be clear that insofar as such notations for knowledge representation are understandable, they are to that degree dependent on some natural language, and hence cannot have a semantics independent of it. From that point, ignored by McDermott, most of the arguments of this paper follow.

Point (b) simply notes a possibility and is passed over immediately, but it is of course a possibility that is argued in this paper as a fact concerning the status of natural languages for axiomatisation (as regards their meaningfulness, at least), and is the issue of principle on which, in my view, the logicist enterprise hangs.

In view of the nature of the arguments in CPR, and the virtual ignoring of issues of principle, while concentrating on ones that give the logicist no more than passing problems, it will not be surprising that I conclude that, although McDermott has apparently reversed his position, he remains wrong on the core issues. There need be no formal problem in saying that: one can perfectly well maintain P and Not-P at different moments, while retaining some false q throughout, where q is in the chain of inference to both P and Not-P. It is such q's that I have sought
to isolate here, particularly as regards the non-independence of the representation language and natural language(s), and the status of statements of meaningfulness for natural languages as a constituting a formal language whose meta-logical properties are to be investigated.

5. Conclusion: are there empirical differences between these research programs?

If we stand back now from McDermott's intellectual struggles and return to the general fray, the obvious general question to ask is whether there are, or will be, genuine testable differences between the logicist program for AI and any discernible alternatives such as CSS. That is to say, will there be programs inspired by one of these classes of formalism capable of performing some indisputably AI task, where the other cannot? All parties should agree to such a test and probably would. But does this fact allow us to sit back comfortably, taking an optimistic view of scientific progress, and await outcomes? I fear the matter is not so simple.

Let us consider an empirical task in natural language processing, one often tackled by AI techniques and occasionally by those of formal semantics: machine translation (MT) from one natural language to another. This is not at all a task chosen at random; it is the original, founding, task of computational linguistics, a task rather like that of playing the piano sonatas of Mozart according to Rubenstein: too easy for the amateur, too hard for the master.

There is no doubt that MT is now possible with some degree of success,(see e.g. Lawson 1982), but that the very hard problems required for a proper solution (like the 'des'/‘les’ problem mentioned earlier) are nowhere near solution. Much of the recent success of MT, it must be said, gives no comfort to either kind of semantics discussed in this paper: it has often been a matter of the very crudest theories, whose performance has been improved, over anything thought possible twenty years ago, simply by the use of software engineering techniques. One might risk the following principle:

There is no theory of language structure so ill-founded that it cannot be the basis for some successful MT.

Those who doubt this should study the history of the SYSTRAN system (Hutchins 1979). The point of the principle, if true, is that it makes any prospect of an empirical test, or decision, as between formal semantics and any other type, applied to a concrete empirical task like MT, very improbable indeed, and that is exacerbated by another principle that lurks behind much of the discussion of this paper:

AI programs in general (including MT programs) do not always work by means of the formalisms that decorate them.

This is an important issue, and one which serves to separate the issue under discussion (that of finding some empirical programming task to settle the issue between types of AI representations and their associated semantics) from what might be an illuminating historical parallel, that of rival theoretical descriptions of
physical phenomena, between which a crucial experiment was sought: ether waves versus relativity, say, or particle versus wave accounts of sub-atomic phenomena.

Programming, alas, is not like that in the following sense: it is perfectly possible to write a program to perform some task (MT will still serve) using a descriptive theory or language, even though, in fact, and sometimes hardly perceived by the programmer, the results are achieved by part of a program that functions in such a way that it cannot be appropriately described by the upper-level theory at all, but requires some quite different form of description. Sometimes this happens for totally banal reasons, ones which involve an element of deceit or flattery: a student programs a system for his supervisor, describing what he is doing in terms consistent with the cherished theory of that supervisor, who does not himself write programs. In order to make the system function, the student is forced to re-conceive the task at another level, and it is there that he develops the "real" theory, one which never emerges in any published description.

Cases like this are not as unusual as one might hope; but a more common situation is what happened to the SYSTRAN Russian-to-English MT itself: it had and has a very elementary system of linguistic description in terms of grammar rules. After nearly thirty years of operation a very large number of lexical "patches" have been added to the system (of the order of half a million (see Wilks 1979)) which deal with particular input strings in Russian, rather as if one had a special dictionary of syntagsms, such that the grammar was only accessed where the syntags failed to "match" the input.

Now, consider the situation where, as I believe to be the case, SYSTRAN is described by the simple grammatical theory although in fact the system's success is largely attributable to the dictionary of "semi-sentences" that does not appear in the top-level description. It should be clear that, whatever the details of this particular system, the situation is not at all one of rival theoretical descriptions of phenomena, as in physics, but of determining what are the real, operational, principles by which a program works, as opposed to its apparent, sometimes decorative, ones.

This problem is well known in the field, but has no obvious solutions: in one sense it is precisely the problem of finding a proper semantics for programs, in the Scott & Strachey sense (1971), one that is close in methodology to formal semantics in the sense in which we have discussed it here. Unfortunately, and whatever the claims of its devotees, the semantics of programs cannot provide this service, even in principle, since that technique requires only the specification of objects and relations so that input is reliably transformed to output, as in a deduction. There is no requirement whatever that the objects chosen have any relationship to the "natural" objects of the theory with which the program works: they could be as remote as were the loyal student's principles from those of his advisor. They can be simply incomensurable in just the way that the semantics of a program at different levels of internal programming language translation are incomensurable (Scott & Strachey ibid).
This conclusion may seem pessimistic but, even in a situation of no reliable test or outcome, the social and psychological forces at work in an empirical and formal discipline like AI continue to function nonetheless. Machine understanders and translators will continue to appear and we shall be able to judge to some degree whether they benefit from formal semantics techniques or not. For those cases where there are reasonable doubts about that, there is coming into being a battery of techniques of reimplementing systems with somewhat changed and controlled principles and structures (see Ritchie & Hanna in press) and that can do much to help us decide what are and are not the formal principles underlying AI programs. So, as Leibniz would write at this point in an argument: come let us compute together!
6. References


Charniak, E. "Connectionism and Explanation". In Wilks (in press).


Sparck-Jones, K. (in press) “They it’s a new engine but it’s still the same SUMP”. In Wilks (in press).


