

POINTERS TO TRUTH¹

Haim Gaifman
Columbia University

1 Preview

Consider the following:

Jack: What I am saying at this very moment is nonsense.

Jill : Yes, what you have just said is nonsense.

Apparently Jack spoke nonsense and Jill spoke to the point. But Jack and Jill seem to have asserted the same thing: that Jack spoke nonsense. Wherefore the difference?

To avoid the vagueness and the conflicting intuitions that go with 'nonsense', let us replace 'nonsense' by 'not true' and recast the puzzle as follows:

line 1 The sentence on line 1 is not true.

line 2 The sentence on line 1 is not true.

If we try to evaluate the sentence on line 1 we find ourselves going in an unending cycle. For this reason alone we may conclude that the sentence is not true. Moreover we are driven to this conclusion by an elementary argument: If the sentence is true then what it asserts is true, but what it asserts is that the sentence on line 1 is not true. Consequently the sentence on line 1 is not true. But when we write this true conclusion on line 2 we find ourselves repeating the very same sentence. It seems that we are unable to deny the truth of the sentence on line 1 without asserting it at the same time.

¹Basic ideas underlying this paper were first presented in a conference to the memory of Bar-Hillel held in October 1985 at the University of Boston. Since then the framework has undergone considerable development. Various stages have been presented in lectures given during 1986 at UCLA, Harvard, Princeton, Stanford and UC Irvine, and in invited talks to a CSLI conference on self-reference (February 1987), to the Pacific Division of the APA meeting (March 1987), to a conference at the University of Texas, Austin and to the 1991 ASL meeting. The technical core of these ideas was presented in the second TARK conference (Theoretical Aspects of Reasoning About Knowledge) held in Monterey, March 1988, and was published in the TARK proceeding (Gaifman [1988]). I have benefited from many reactions, observations and discussions of the issues. Among the many to whom thanks are due I would like to mention, in particular, David Kaplan, Brian Skyrms, Hillary Putnam and Charles Parsons. I am also indebted to Rohit Parikh for useful comments and for finding a bug in one of the earlier proofs.

The mathematical part of a previous draft has been replaced here by a more informal description and an illustrative example, which, I trust, are sufficient to give a good idea of the method. I did this for reasons of length and in order to preserve a non-technical profile in what is intended as a philosophical paper. I plan to present the mathematics, in full, in a separate article.

Essentially, this puzzle is a reformulation of what van Frassen [1968] has named "The Strengthened Liar"; "The Strong Liar" is what I shall call it here. We might recall here that the usual version of the Liar paradox consists in the fact that the sentence on line 1 is neither true nor false, because either assumption leads to contradiction. That version can be resolved by admitting gaps, i.e., by giving up the rule that every sentence must be either true or false. The present paradox is an altogether different kettle of fish. It consists in our seeming inability to express a true conclusion — namely that a certain sentence is not true — without repeating the very same sentence, or one equivalent to it.

It is known that the semantic paradoxes can be reconstructed in various modal frameworks (cf. Montague [1963]). Indeed, using a knowledge predicate we get an analogous puzzle:

line 1 What is written on line 1 is not known by Jill to be true.

Regarding the sentence, Jill concludes that she does not know it to be true. (Her knowing it to be true would imply that it is indeed true, which, given what the sentence says, would imply that she does not know it to be true; thus — a contradiction.) Jill writes her conclusion on line 2:

line 2 What is written on line 1 is not known by Jill to be true.

So Jill knows the truth of what is written on line 2, because she has just deduced it, but not of what is written on line 1.

A similar puzzle can be made with 'known' replaced by 'believed'.

With 'necessary' we get the following version:

line 1 The sentence on line 1 is not necessarily true.

line 2 The sentence on line 1 is not necessarily true.

Again, the line 1 sentence is not necessarily true; because if it were, it would not be true and, a fortiori, not necessarily true. This conclusion is necessarily true, because we have just proved it. Writing it on line 2, we get a necessarily true sentence, which merely repeats that not necessarily true sentence on line 1.

The moral of all these puzzles is simple: In situations of this nature we should assign truth-values not to sentence-types but to their tokens. Having concluded that the line 1 sentence is not true, we express this conclusion by displaying another token of the very same sentence. This second token, on line 2, expresses something altogether different from what is expressed, if anything, by the token on line 1. The people who agreed with Jill in the Jack and Jill exchange had no difficulty of appreciating this point.

In this respect, modal predicates like 'know', 'believe' and 'necessary' are in the same boat with 'true' and the same remedy is required; namely: to allow different tokens of the same sentence to have different meanings, or — if you want — to express different statements, or different propositions. One should expect the formalisms to differ according to the predicates

in question, but the same general framework will underlie them. In the present work we set up the formalism for truth, thereby providing also the basic framework for the various modalities.

Following the intuition just given, we can see that in general what a token expresses depends on (1) what it says, i.e., on the sentence type and (2) on the whole network: on the tokens that the sentence refers to and on the tokens that they in their turn refer to, etc. This is what distinguishes the self-referential sentence-token on line 1 from its non-self-referential brother on line 2.

I shall propose a formal setup for specifying networks of tokens and an algorithm for assigning truth-values. It assigns the sentence-token on line 1 the value GAP, that on line 2 the value T (True) and it yields the same kind of intuitive results in general.

I base the formalism on three truth-values T, F and GAP. The first two of which are referred to as the standard truth-values

The third is not merely an "undefined". It signifies not only an absence of a standard value, but recognized failure. The idea is that we assign a tokens, or | as we shall later see | pointers, the value GAP when, in the course of the evaluation, we conclude that they must fail, i.e., that the rules will not enable us to assign them T or F. We can then go on and use other, "uninfected" tokens to assert of a previously failed token that it is not true, or not false, or that it is neither true nor false. By making GAP an active value we can construct on top of the gap instead falling into it.

Now, the concept of a token is too narrow for the purpose of a general framework. For we might want to refer to sentences without having them displayed somewhere as tokens. Suppose that there is just one sentence on line n, then we can use "the negation of the sentence on line n" in order to refer to the negation of that sentence, independently of there being a token of the negation. And, in general, we might speak about the sentence obtained from that on line n by any syntactic manipulation, e.g., the substitution of some term by another. Therefore I use a more general concept, that of a pointer. A pointer is any object that is used to point to some object.

A token constitutes a special kind of pointer, it points to the type of which it is a token.

In general, pointing is determined by a variety of conventions. Being a pointer is not something intrinsic in an object, but amounts solely to its functioning in a certain role. Any object can serve. If desired, we can regard descriptions as pointers to the described objects. And one can set up a formalism whereby every sentence-type is a pointer to itself. However, the solution to the semantic paradoxes requires that we have many pointers to the same sentence-type, so that when a pointer fails another is available.

As noted, any object can be pointed to. But since in this paper only pointers to sentence-types are needed, I shall reserve "pointer" for this case, unless the context indicates otherwise.

It is useful not to enter, at this stage, into the nature of pointers, but to take them as primitives. Thus, I assume a language, a set of objects called "pointers", which have names in that language, and some fixed arbitrary function that correlates with each pointer

a sentence (of that very same language) to which it points. A pointer whose name occurs in some sentence can point to any sentence whatsoever; all possibilities of direct or indirect self-reference are thus covered.

Only the following additional structural elements are needed for the pointer system. A correlation that correlates with each pointer to a compound sentence pointers to its sentential components. Thus any p pointing to $A \alpha B$, where α is a sentential connective, has correlated pointers p_1 and p_2 , which point, respectively, to A and to B . And if p points to the negation $\neg A$, then p_1 points to A (and we can, in this case, stipulate that $p_2 = p_1$). We call p_1 and p_2 the derived pointers of p . If the language has quantifiers, we correlate also with every p , which points to $\forall v A(v)$ or to $\exists v A(v)$, and with every constant term t , a derived pointer p_{jt} , which points to $A(t)$ (the result of substituting t for v in $A(v)$).

Note that if p is a token, then p_1 and p_2 can be simply identified with the corresponding subtokens. But with quantifiers we must go beyond tokens, for in general we cannot identify p_{jt} with a token.

For our present purposes, as long as quantifiers are not included, we can take our pointers to be just tokens. The reader may, for the sake of convenience, assume this to be the case. Extending it to the more general case is straightforward.

The upshot of this approach is a new kind of semantics in which truth-values are assigned to pointers and the usual recursive definition of truth is replaced by a set of rules for evaluating networks.

Here is an informal description of the algorithm, illustrated by a simple example. This, I think, should give an adequate idea of the method, providing thereby a basis for the philosophical discussion that follows. (A full mathematical presentation is planned for a separate companion paper. A concise version containing the formal definitions and the statements, without proofs, of the main theorems can be found in [Gaifman 1988].)

Consider the above mentioned system of language and pointers. Assume that the language (based on predicates and, possibly, function symbols) is interpreted, except for two, so-called truth predicates (or semantic predicates): $Tr()$ for 'true' and $Fa()$ for 'false'. These two predicates take pointer-names as arguments.

Assuming that p, q, r , etc. are pointers, and that the pointer names used in this discussion are also their names in the formal language, we get sentences of the form $Tr(p), Fa(q)$, etc., where p, q , etc., are pointers to sentences in this very same language.

The goal is an assignment of truth-values to all the pointers. It is achieved by the algorithm| also called the evaluation procedure| which is based on a collection of rules for assigning truth-values. The rules are naturally divided into various groups. The so-called standard rules, which make up the first group, correspond to the traditional rules for assigning standard values (i.e., T or F), including the classical rules for the truth predicates.

Standard Rules: Pointers to sentences of the form $P(t_1; \dots; t_n)$, where P is a non-semantic predicate, get their truth-values in the usual way, via the interpretation of the predicates and the denoting terms. Such sentences are called basic. Evidently, different

pointers to the same basic sentence get the same truth-value. It is a standard value determined by sentence-type alone. And this holds, as long as only standard rules are employed.

Now, if p points to a non-basic sentence, then, in the usual order of things, its value is determined by the values of other pointers, which, so to speak, are called directly by p . Thus, if p points to the disjunction $A _ B$, then the pointers it calls directly are defined to be its derived pointers p_1 and p_2 (which, recall, point to A and to B). If one of them gets the value T , so does p | no matter whether the other has been assigned a value, or what that value is. If both get F , p gets F as well. The standard rule for negation, i.e., the "toggling" of the standard truth-value, should be obvious. Other connectives are treated by expressing them in terms of $_$ and $:$.

A pointer, p , to a quantified sentence, calls directly each of the pointers p_j to the substitution instances. The rules for quantified sentences are obtained by treating them as (possibly infinite) disjunctions, or conjunctions. For the sake simplicity, all objects in the universe of interpretation are assumed to be named by constant term, and quantifiers are interpreted substitutionally. This assumption is not essential. Quantifiers can be interpreted referentially at the cost of some additional structural items².

Finally, a pointer to $Tr(q)$, or to $Fa(q)$, calls directly q ; if q gets a standard value, then p gets, in the first case | the same value, in the second case | the opposite value. These are the standard rules for the truth predicates.

The essential part of the algorithm, by which it enhances classical logic, is constituted by two additional groups: The gap rules determine the cases of failure, calling for the assignment of where GAP. The jump rules determine assignments of standard-values, which are based on previous failures. They enable us to use new pointers in order to make semantic assertions about previously failed ones (e.g., that they are not true, or that they are not false, or neither true nor false). Before I go into these two groups, here is a rather simple illustrative example.

Let John, Joan, Jack and Jill make the following statements:

John: What Jill says is not true and what Joan says is not true.

Jill: What Jack says is true.

Joan: What Jill says is not true.

Jack: Either McX's conjecture is true or what Jill says is false.

For simplicity, we can use here each speaker as a pointer to his uttered sentence. The resulting network is represented in Fig. 1. The arrows between pointers (the nodes) do not represent the pointing, which is a relation between pointers and sentence-types, but the above mentioned direct calls between pointers. McX's conjecture, represented as a box, is not spelled out. It might involve a whole network of its own; but I assume, for the purpose of illustration, that it does not refer, directly or indirectly, to any of the four utterances.

²Namely, pointers would point to $w^@s$; truth-values would be assigned to pairs consisting of pointers and assignments of objects to variables.

Actually, the diagram has been simplified by merging certain arrows and hiding certain derived pointers; e.g., the arrow from John to Jill should be split into an arrow from John to John1 (which points to the the first conjunct of what John says), an arrow from John1 to John11 (which points to "What Jill says is true") and an arrow from John11 to Jill. Note that the sentence-types to which the pointers point can be read from the diagram though they are not explicitly displayed.

Fig. 1

The algorithm is applied recursively. Assume that it results in assigning F to McX. This leaves Jack and Jill in a closed unresolved loop. Both get at this stage the value GAP. Then Joan who makes an assertion about Jill and does not belong to the loop gets a standard (T or F) value (here a jump rule is employed). Since her assertion is true, she gets T. In the same way, the first conjunct in John's assertion gets T. But the second conjunct (which asserts the non-truth of Joan) gets F, hence John gets F.

Had McX gotten T, Jack and Jill would have gotten T, while John and Joan would have gotten F.

In the case of the Two Line puzzle the network is:

Fig. 2

For 'i' read: 'the sentence-token on line i'. Right at the beginning we get a closed loop consisting of the pointer 1. It gets the value GAP. Then 2 gets the value T.

In the Four Speaker example, the standard rules for Tr and for negation are used for determining the value of John2 (which points to the second conjunct of what John says). Also the standard rule for conjunctions is used for determining the value of John. The additional rule groups are as follows.

Gap Rules: The most important of these is the closed loop rule, which states the following: If, in the course of applying the evaluation procedure, a closed unevaluated loop forms and none of its members can be assigned a standard value by any of the rules, then all of its members are assigned GAP in a single evaluation step.

[A closed unevaluated loop is a set, say L, of unevaluated pointers (i.e., which have not been assigned truth-values), such that: (1) Unevaluated pointers called directly by any member of L also belong to L and (2) From each member of L one can reach any other member of L by one, or more, \bar{n} itely many direct calls; i.e, by a sequence $p_1; ::; p_n$, where $n > 1$, the p_i s are in L and each calls directly its successor.]

In the Four Speaker example, Jack and Jill get their value via the closed loop rule. And in the Two Line example this rule applies to the token on line 1, which forms by itself a closed unevaluated loop. In both examples the loops are simple cycles. But, in general, the closed unevaluated loop can be as complex as you can imagine and might consist of any number, \bar{n} ite or in \bar{n} ite, of pointers.

There is also a simple gap rule that states: A pointer is assigned GAP if all pointers called directly by it have been assigned values, but the pointer cannot be assigned a standard value by any of the rules^{3 4}.

Jump Rules: Assume that q points to Tr(p), or to Fa(p), and that p, but not q, has already been assigned GAP; then the jump rules assign q the value F. Consequently, if an unevaluated r points to : Tr(p) or to : Fa(p)), where p has been assigned GAP, then, by a jump rule, r1 (which points to Tr(p) or to Fa(p)) gets F and, by the standard negation rule, r gets T. Thus, the jump rules provide for the possibility of successfully asserting that

³ Given the standard rules, the effect of this rule is to enforce Kleene's strong three-valued truth tables: A pointer p to a disjunction A_B is assigned GAP if p1 and p2 have been assigned values that do not enable us to assign, by some other rule, a standard value to p; this is the case where one of p1; p2 is assigned GAP and the other is assigned either GAP or F. However, in other variants, based on changing the standard rules (e.g., using supervaluations) the simple gap rule does not imply Kleene's strong truth-tables.

⁴ The gap rules determine the assignments of GAP that signify recognized failures. However, GAP can be assigned also by an additional so called give-up rule: If at some stage we are left with unevaluated pointers to which none of the other rules apply, then all of them are assigned GAP. Gaps assigned by this rule do not constitute recognized failures, but are rather like Kripke's 'unde \bar{n} ed" u. It can be shown that in all networks of \bar{n} itary type, i.e., where every pointer calls (directly, or indirectly) \bar{n} itely many pointers, the give-up rule is never employed. The aim of more sophisticated versions of the algorithm is to avoid, as far as possible, the employment of such a rule in non- \bar{n} itary networks.

a given GAP-pointer is not true, or that it is not false. \Jump" signifies here true ascent in the metalinguistic hierarchy.

In our example, a jump rule (for Tr) is needed, in order to assign Joan and John1 their values. It is also invoked for assigning a value to the line 2 token in the Two Line puzzle.

To sum up, the assignment of GAP signifies recognised failure: We decide that the pointers in question fail to evaluate to a standard value. The grounds for such a decision are given in the gap rules. The assignment of GAP can serve as a basis for further assignments of standard values via the jump rules. What is inexpressible in the usual denotational semantics is thus expressible through network evaluation.

Various mathematical results, which I shall not go into here, establish the desirable properties of the evaluation procedure. I shall only describe informally two of the basic results, whose significance can, I hope, be seen:

Order Independence: The rules can be applied in any order, to whatever pointers; the (possibly transfinite) sequence of applications results in a unique total valuation which does not depend on the order of applications⁵.

Limited Pointer Dissent (abbreviated as LPD): Two pointers to the same sentence are assigned different values only if one of the pointers is assigned GAP. This is also true for pointers pointing to logically equivalent sentences.

The evaluation procedure, called also 'algorithm', has indeed the form of an algorithm: an exact prescription for applying a well-defined set of rules. But this 'algorithmic' form should not disguise its true potential. It applies to all networks, including infinite ones of arbitrary cardinality, and it extends both Tarski's and Kripke's systems. Each of these is obtained by deleting from my algorithm certain rules: Tarski's truth definition is what we get if we employ only the standard rules for basic sentences and for connectives and quantifiers (and assign GAP to all pointers that are left at the end unevaluated). Kripke's system is obtained if we restrict ourselves to the standard rules, including the standard rules for the truth predicates.

It can be shown that if the jump rule is deleted, then a pointer's value depends only on the sentence it points to; truth-values would be thus determined by sentence-types alone.

Note that, in Kripke's construction, both tokens in the Two Line puzzle constitute gaps (i.e., remain unevaluated), as do the utterances of the four speakers. In the variants of Gupta and Herzberger they are assigned oscillating values.

One could say that the two major points of difference between the present proposal and all the previous ones are: first, the use of pointers and second, the employment of the gap and jump rules.

⁵A rule application extends a given partial valuation. If only the standard rules are used, then these extensions constitute monotone operators and the Order Independence claim follows easily by a fixed-point argument. However, with the gap and jump rules, the operators are not monotone. The claim is nonetheless true, but the proof is much more involved.

Recovering The Metalanguage Hierarchy

There is a precise definition, based on the evaluation procedure, which correlates with each pointer an ordinal number expressing an intuitive notion of "metalinguistic level". The pointers can be thus stratified so as to reflect a Tarski-like hierarchy.

Roughly speaking, a pointer's level is the number of nested applications of jump rules, needed in order to evaluate the pointer. In the Four Speaker example, if α is the level of McX, then also Jack and Jill are on level α , whereas John and Joan are on level $\alpha + 1$.

We can also refine each of our levels so as to reflect a minor, or local subhierarchy within it. Roughly speaking, the minor subhierarchy measures the height of nested applications of the standard rules for the truth predicates.

Unlike Tarski's hierarchy, we have a single language with a single truth-predicate and the stratification is effected after fact, i.e., after the assignment of truth-values. In this respect it is like the hierarchy obtained via Kripke's construction; as he aptly puts it: the pointers (in his case | sentences) seek their own levels. But his whole hierarchy of grounded sentences corresponds (exactly) to our local subhierarchy of level 0. Real climbing starts where Kripke leaves off. All our non-zero levels are in his model gaps ("undefined").

The construal of truth as a predicate over pointers rather than types is related to the metalinguistic hierarchy as follows. Call pointers dissenting if they point to the same sentence and have different truth-values. We know, by LPD, that of two dissenting pointers one is assigned GAP; it turns out that the non-GAP pointer is one level higher in the hierarchy:

Higher Level Dissent (abbreviated as HLD): Whenever p_1 and p_2 are dissenting pointers, if $l(p_1)$ is p_1 's level and p_1 is the GAP pointer, then $l(p_2) = l(p_1) + 1$.

The significance of LPD and HLD will be discussed in section ??.

Before proceeding let me clarify certain points, answering thereby possible questions and allaying, perhaps, certain misgivings that might have arisen. Some of these points will be reconsidered, at greater detail, in later sections.

What Pointers Express: Some have found the idea of assigning truth-value to pointers strange; the phrase "that pointer is true" may sound bizarre. Truth and falsity, so it is argued, have to do with what we say. Surely it is inappropriate to attribute them to some arbitrary objects, merely by virtue of their pointing function? To argue thus is to misunderstand the present proposal. The phrasing is not essential. It is not strange to speak of the truth or falsity of sentences, meaning thereby sentence-tokens; and if it comes to pointers, one could speak of a pointer pointing to a truth, or to a falsity, or indicating one, or failing to indicate a truth, or a falsity. We can regard pointers as means of expression.

As far as the ontology of statements, propositions, meanings, etc. is concerned, my proposal is neutral. A nominalist who bars intensional entities can use something like the evaluation procedure as part of the account of the functioning of language. But the proposal is compatible with seeing pointers (in the contexts of networks) as means of expressing propositions, or statements ⁶. On a pretheoretic level, it is not difficult to make sense of "the

⁶cf. footnote ?? concerning "propositions" and "statements".

proposition expressed", or "the statement made" by means of a pointer. For example, what is expressed by the sentence-token on line 2 is simply that the sentence-token on line 1 is not true, which is to say: is not assigned the value T by the evaluation procedure. Although the token on line 1 is of the same sentence-type, it fails, because of the loop, to express this. Similarly, in the Jack and Jill exchange, Jack says of himself that he does not express a proposition, but in this very saying no proposition is expressed, or | at least | no proposition that says this. What Jack seems | but has failed | to express is successfully expressed by Jill. If only sentence-types are considered, Jack and Jill appear to have stated the same thing. Actually they did not. The source of the illusion is that they employed sentence-types that are, in the given conversational context, logically equivalent.

The following can perhaps add clarification. Two intuitive theses about truth lead to the Strong Liar:

(T1) If the sentence 'x is not true' is true, then x is not true.

(T2) If x is not true, then the sentence 'x is not true' is true.

Here 'x' ranges over sentence-types. We get the Strong Liar when x is, or is equivalent to, 'x is not true'. But if we construe 'true' as a predicate of pointers, then 'x' should range over pointers and (T1) and (T2) should be modified by replacing 'the sentence' by: 'any pointer to the sentence'. We can still maintain (T1) in its new form, but we have to qualify (T2) by rewriting it as:

(T2^a) If x is not true then any unfailed pointer to 'x is not true' is true.

The paradox is thus blocked by allowing pointers (and, in particular, tokens) to fail, and by giving up (T2) for failed tokens of 'x is not true'. If we apply the analogous strategy to types, when x is a Liar sentence, it is the type 'x is not true' which fails; therefore one cannot use it to say that x is not true.

Further observations concerning pointers and propositions are made in section ??.

An Essentially Non-Tarskian Semantics: In Tarskian semantics, truth is a property of sentence-types; tokens as such are not considered. That tokens play a central role in natural language, due to the abundance of indexicals and demonstratives (words like 'I', 'you', 'now', 'this' 'that') is, of course, very old news. But it would seem that the Tarskian framework can be adjusted, or rather supplemented, so as to handle phenomena like indexicality. The dependence of truth-value on tokens, can | in such cases | be traced to the shifting denotations of indexical phrases. Having replaced (presumably, in some systematic way) these items by others, whose denotations, within the considered discourse, are not context-sensitive, we can then apply Tarskian semantics. To use an admittedly simple example, transform 'He is tall' to 'X is tall', where 'X' is an appropriate context-insensitive name, or description, of the indicated person, and evaluate the result as a sentence-type. If needed, a further reduction can be used to eliminate the temporal indexicality of 'is'. The truth

predicate which is defined over tokens is thus reduced to a predicate defined *à la* Tarski over types.

The present situation is altogether different. The predicate defined by the evaluation procedure is irreducibly over tokens, or | more generally | over pointers ⁷. As I will later show, differences in truth-value that derive from differences of token-place in referential the network, cannot be traced in any way to indexical-like, or other ambiguously denoting terms that occur inside these tokens.

Attempts like Burge's [1979] to resolve the semantic paradoxes by construing 'true' as an indexical cannot but fail to achieve that goal. Indeed, as the analysis will show, Burge's account puts all the burden of work on a rather obscure mechanism of implicatures, leaving indexicality idle. The general argument concerning indexicality is laid out in section ??; the more specific discussion of Burge's work is in section ??.

The departure from Tarskian paradigm is significant in that it shows how modeling, radically different from standard compositional semantics, can be formally handled in a way as precise as Tarski's truth definition for formal languages. An argument propounded by Davidson has it that the ability of finite beings | such as humans | to use a potentially infinite number of sentences is to be explained by assuming a finite number of rules⁸, whereby meanings of compound expressions are determined by the meanings of their components. A setup of this nature resembles Tarski's truth-definition. That something of the kind underlies a great deal of the functioning of language cannot be doubted. But it is equally true that a great deal lies beyond it. If human ability for controlling a potential infinity is to be modeled by a formal device | a plausible working assumption, though far from an axiom | then the appropriate concept is that of an algorithm. A finitely axiomatized compositional semantics amounts to a rather special kind of algorithm. Appealing as the model is, it is more than plausible that the processing of language, on the level of meaning, involves also algorithms of quite a different nature. I shall say more on this score in section ??.

Truth, Falsity and GAP: In the pointer formalism the predicates representing truth and falsity figure as two primitives. However, the evaluation rules relate truth and falsity, through negation, as follows: A pointer to the negation : A gets the value T, or the value F, iff the derived pointer to A gets the opposite value. Thus, our concept of falsity accords with the well known thesis, according to which a sentence is false iff its negation is true. It is possible to set up an equivalent formalism which uses only Tr, or only Fa, as a primitive⁹.

⁷Formally, one can of course redefine "type" by stipulating that tokens that get different truth-values (e.g., those on lines 1 and 2) are to be considered of different type. Isaac Levi observed that such a stipulation might not be bizarre, given that the token-type distinction is not always clear cut and that homonyms might be considered as different types, because of their different meanings. But if we do this, the very definition of "type" presupposes the assignment of truth-values to tokens; in this sense, the truth predicate is still irreducibly over tokens.

⁸On the syntactical level, Chomsky had argued in this way for an innate language-processing mechanism.

⁹ We can define the evaluation procedure for a subsystem based on a single truth predicate. But to have the expressive power of our present system, we need a way to express both Tr(p) and Fa(p); e.g., both are needed in order to say that p is a gap ($Tr(p) \wedge Fa(p)$). This requires an additional structural item

Let 'false₁' denote falsity in the sense employed here. Another way of relating falsity to truth is given by 'false₂': A sentence is false₂ just when it is not true.¹⁰ Consequently, every sentence is either true or false₂. The truth-value gaps disappear miraculously, when one reads 'false' as 'false₂'. In fact they have only been camouflaged, and they can be easily unmasked: A sentence is a gap if and only if both it and its negation are false₂. As far as expressive power is concerned, the systems based on the two interpretation of falsity are equivalent; there are obvious translations from one terminology to the other. A formalism based on 'false₂', which is equivalent to the present one, can be set up at some cost of additional structure¹¹.

Cases in which a sentence and an apparent negation of it are both false are provided by Russell's theory of descriptions. E.g., (i) 'the present king of France is bald' and (ii) 'the present king of France is not bald'¹². This however, is due to the restricted scope of 'not' in the reconstruction of (ii). There is no gap here. For true negation | one that applies to the whole sentence | does convert falsity to truth: 'It is not the case that the the present king of France is bald' is true. Much of the appeal of Russell's theory is due to its ability to distinguish between local negation, as exemplified in (ii), and true negation, expressible by 'It is not the case that ...'. Whether accepted or not, Russell's theory rests squarely within the framework of classical two-valued logic.

The present situation is altogether different. One cannot explain the 'falsity' (i.e., non-truth) of the line 1 sentence (in the Two Line puzzle) by appealing to the scope of 'not'. For we can write on line 1:

It is not the case that the sentence on line 1 is true.

And this sentence (type or token, the distinction does not matter here) should be a gap. For let x be this sentence; then if it is false, so is the sentence 'x is true'. Hence both 'x is true' and 'it is not the case that x is true' are taken to be false; as counterintuitive as the analogous case of (i) and (iii). (And, of course, taking that sentence to be true fares even worse.) Formally, one can construe 'false' as 'false₂', without incurring a contradiction. But this is no more than a lexical gap-masking device; it is certainly a far cry from classical two-valued logic.

There is a moral in this concerning propositions. Either we say that Liar type sentences express no propositions, or we make place for propositions that constitute truth-value gaps.

for pointers: A negation operant that associates with every pointer p a pointer, say neg(p), to the negation of the sentence pointed to by p. We can then translate Fa(p) into Tr(neg(p)). (We should stipulate that neg(p)1 = p and that if p points to a negation then neg(p1) = p). Note that, in general, neg(p) will not be a token, even if p is. The present formalism avoids the need for a negation operant; hence, unless we consider quantifiers, we can assume that all our pointers are just tokens.

¹⁰This reading of 'false' is adopted by Barwise and Ethchemendy in [BE 87]. The same reading, for the case of atomic sentences 'ϕ is ©', where ϕ is a (possibly non-denoting) description, is suggested in in [Kaplan 1970]).

¹¹We shall need a negation operant for pointers, cf. footnote ??.

¹²I am ignoring here the more radical Russellian position according to which definite descriptions are incomplete symbols to be eliminated, in the final account, altogether.

The Liar and the Truth Teller: The algorithm described so far treats the Truth Teller (a sentence asserting its own truth, e.g., 'What I am saying now is true') in the same way as the Liar, both get GAP. Our framework does, however, provide means for distinguishing between the two. The distinction is made by considering consistent assignment of truth-values, or what we call in the technical paper self-supporting valuations. There is a total self-supporting valuation which assigns the Truth Teller (token) the value T; and there is also one which assigns it F. But every self-supporting total valuation must assign the Liar (token) the value GAP. (Roughly speaking, a self-supporting total valuation is one that is closed under rule applications: applying to it the rules of the algorithm yields the same valuation. For reasons of space, I cannot elaborate on this here.)

Different Variants of the Algorithm: The present variant of the algorithm leads to an assignment that conforms to Kleene's three-valued truth-tables (cf. footnote ??). We might, however, prefer to assign all (pointers to) tautologies the value T, e.g., any pointer to $A \rightarrow A$, or to $A \vee \neg A$. This can be achieved by using a variant in which the set of standard rules is replaced by another set, based on so-called supervaluations. In the usual supervaluation method which applies to sentence-types a sentence is given a standard value when it gets this value in all possible complete standard extensions of the partial valuation defined so far. This idea can be adopted to assignments of values to pointers; I defer details to the technical paper.

It is noteworthy that only the standard rules have to be changed in order to obtain the supervaluation variant; the gap and jump rules, remain the same. (Changing the standard rules can affect the applicability of other rules, because certain rules have preconditions requiring the inapplicability of other rules. The rules themselves, however, need no change.) This is also true in the case of other variants and it illustrates the modular character of the evaluation procedure. We can augment or change one of the rule groups without touching the remaining ones.

More sophisticated variants, not to be presented here, incorporate additional gap rules; the idea being that sometimes we can assign GAP by recognizing ahead the eventual failure of certain pointers. This makes it possible to apply the jump rules on a wider scope and, thus, to reduce the total assignments of GAP. Some of these versions call for distinctions between various kinds of pointers, to be expressed formally by appropriate enrichments of the pointer formalism. These variants are aimed at achieving higher degrees of self-contained systems for some highly expressive languages a goal that will become clearer in the next section.

The rest of the paper is organized as follows. In the next section I try to put the present proposal in wider perspective. I define the concept of a black hole in order to bring out the essential problem posed by the Strong Liar and its kin, and to clarify the goals of the proposal and the motives for setting up more sophisticated versions of the algorithm. Section ?? is devoted mainly to showing that 'true' is a non-indexical predicate over tokens (or, generally, over pointers) and to the import of a non-Tarskian semantics. Section ?? contains some

observations concerning pointers and propositions, and comments on some aspects of the Barwise-Etchemendy work. In section ?? I discuss three previous accounts of the Strong Liar those of Parsons (or Russel-Parsons) Burge and Skyrms.

2 Black Holes

Let us extend the original two line text as follows:

line 1 The sentence on line 1 is not true.

line 2 The sentence on line 1 is not true.

line 3 The sentence on line 2 is not true.

line 4 The sentence on line 3 is not true.

.

.

.

line n+1 The sentence on line n is not true.

Suppose that we assign truth-values to sentence-types. Then the sentences on line 1 and 2 are equivalent, being the same. Moreover, if the sentences on lines $k_j - 1$ and k are equivalent then we can derive the equivalence of the sentences on lines k and $k + 1$ from the following mild assumption: If '...' and '___' refer to equivalent sentences, then '... is not true' is equivalent to '___ is not true'¹³. Hence all the sentences are equivalent. Since the sentence on line 1 is not true, the sentence on line 3 is also not true, but what I have just stated is not true, because it is the sentence on line 4 (or an obviously equivalent reformulation of it), and also this last statement of mine is not true, because it is the sentence on line 5, etc. None of these sentences can be successfully asserted, because none of them is true; but again I find myself slipping into non-truth: What I have just said is not true for it obviously includes the conjunction of these very same sentences; and also this last assertion is not true, and so on ad infinitum.

The innocent looking gap has developed into a black hole that sucks into itself every sentence asserting that some sentence in the hole is not true¹⁴.

¹³This law is operative inasmuch as truth-values are determined by sentence-types. The giving up of the law signifies that something else is involved in determining truth-values.

¹⁴So far we assumed the above mentioned law of equivalence and the following rules that imply that the line 1 sentence is a gap: (1) If 'S is true' is true then 'S' is true. (2) If 'S is not true' is true then 'S' is not true. (3) If the negation of a sentence is false then the sentence is true. We shall also assume, in the rest of the discussion, that if the negation of a sentence is true then the sentence is false and the equivalence of 'S is false' and 'not-S is true'.

Formally, a hole-hierarchy is definable as follows: 'S' is a 0-hole if it is a gap and, by induction: 'S' is an $n + 1$ -hole if it is an n -hole and 'S is true' and 'S is false' are gaps. Then 'S' is a black hole if it is an n -hole for all n .

We shall use 'hole', unqualified to denote a 1-hole. Thus, we cannot assert of a hole that it is not true, or not false, or a gap (neither true nor false), without this second assertion being itself a gap.

Of course, syntactical properties can be asserted of any sentence. But no information about the truth-value of a hole can be conveyed directly. And as for black holes, they are semantic untouchables. No semantic information about them can be conveyed in any way, be it as indirect and across as many layers as one may wish.

Actually, the arguments produced so far in the case of our example imply only that the sentence on line 1 is a "semi black hole"; we have only iterated '...is not true' but not '...is not false'.¹⁵ This is however bad enough. And it becomes a fully "edged black hole, under some additional very natural assumptions. In particular, it is easily seen that every gap is a black hole if we assume that 'S' and 'S is true' are equivalent, under an equivalence relation satisfying Leibniz's substitutivity law¹⁶ (Note that this is not the Tarski biconditional: 'S' is true \leftrightarrow S, but only the requirement that 'S is true' and 'S' should get the same truth-value, including GAP.) It is difficult to see how this equivalence can be avoided if truth-values depend only on sentence-types. Indeed, the equivalence holds in the systems of Kripke et al, and those of Gupta and Herzberger. Every truth value gap is a black hole in these systems.

When truth-values depend only on types, black holes are thus bound to occur on the most elementary levels of language. (Think how little is needed to produce sentences like the sentence on line 1.) Within such a system the little game I have just played, of repeatedly refuting myself by trying to assert the "unassertable", signifies a genuine metaphysical mystery. All this, in spite of the fact that the trick is quite transparent and that we, Jack and Jill included, have no trouble at all of asserting this sort of "unassertable" by employing another token of the very same paradoxical sentence. Linguistic usage includes a mechanism for overcoming the black-hole dilemma. And if our goal is to model natural language, or a basic aspect of it, by a formal language containing so to speak its own truth predicate, then the elimination of black holes| at least those of the simpler varieties| should be an overriding concern. Failure in this respect means that we have failed to capture something very essential to the functioning of language.

¹⁵Our minimal assumptions (cf., the preceding footnote) do not imply that 'The sentence on line 1 is false' is a gap. This can be shown by constructing a model, to be sure, a highly pathological one, satisfying the minimal assumptions, in which this last sentence is false.

¹⁶That is to say, the equivalence of 'S₁' and 'S₂' entails that of 'S₁ is true' and 'S₂ is true' and similarly for 'false'. Actually, for the sentence on line n to be a black hole a weaker assumption than the equivalence of 'S' and 'S is true' will suffice. E.g., for $n = 2$ the assumption is that 'S is true' is equivalent to 'S is true' is true'. For larger n one iterates 'is true' accordingly; the assumption becomes weaker with each iteration.

Though Kripke notices the failure, he plays it down, not giving the issue the central place it deserves¹⁷. \The sense (so he writes) in which we can say, in a natural language, that a Liar sentence is not true must be thought of as associated with a later stage in the development of natural language, one in which the speakers re°ect on the generation process leading to the minimal fixed point."

Indeed, the nontruth of the sentence on line 1 hinges on a sort of metareasoning. But this does not place the assertion on line 2 at a later stage in the development of natural language." Certainly not at a stage that comes after the construction of the minimal fixed point, an inductive process that moves through all recursive ordinals! The fact that, after carrying out this transfinite construction, we are still left with the elementary problem of making sense of what Jack and Jill said shows that, far from having completed a home run, we are still at base one.

The basic (and simplest) version of my proposal eliminates the holes in all situations of infinite type| those in which every pointer calls only finitely many pointers¹⁸. To this type of network, which I call locally finite, belong all the tangled loops that can arise when finitely many people make assertions about each other, provided that altogether finitely many assertions are involved. In particular, it contains the examples discussed in the literature. The absence of holes is formally expressed by the following result, which holds for locally finite networks under quite general conditions¹⁹:

If p has the value GAP then there is a pointer q to the sentence $: Tr(p) \wedge : Fa(p)$ (i.e., the sentence asserting that p is a gap), such that q has the value T.

This means that we can successfully assert, of any gap, that it is a gap. Another result, which applies under the same conditions, states that every sentence has a pointer to it that has a standard (T or F) value. (Recall that, by LPD, pointers to the same sentence must agree in value if their values are standard.)

When the language is sufficiently expressive, then in some situations of infinite type, holes, even black holes, appear. Of these, some can be eliminated by more sophisticated versions of the algorithm and it appears that further improvements are possible. It is far from clear how far one can go towards the elimination of holes. In the context of mathematical languages the elimination of holes will bring us near to what one might consider a "universal language" | a self-contained system, one that includes its own semantics. For this reason alone it is highly plausible that rich languages must have holes. Note, however, that we have no formal proof of this. The standard argument from the Liar is no longer valid for setups of the kind proposed here. But perhaps the question of proof should not even be raised at the present stage, since it is not clear what the present proposal might, in further developments, encompass.

¹⁷This point has been nicely put by Burge, [1979].

¹⁸A pointer p calls a pointer q if one can reach q from p by one or more, finitely many, direct calls; cf. the "Preview" where the algorithm is described.

¹⁹The required condition is that there are finitely many independent pointers to each sentence-type, where "independent pointers" are those that have no common subpointers, a subpointer being the natural generalization of "subtoken".

In the broader perspectives of natural language "hole like" phenomena are bound to occur. Some discourses take us to the edge of meaning, whereby we are tempted to express the inexpressible or to think the unthinkable. Holes are perhaps the inevitable price for a powerful language capable of evolving. Indeed, some situations can be described only by saying that we have changed the language, or that we have altered the very fabric of our conceptual framework. But I do not think that the Jack and Jill exchange calls for such a description. Here we can at least tidy up the more elementary levels of language.

A comparison with set theory comes naturally to mind. The analogy between the semantical and set theoretical paradoxes is obvious and well known. Russell considered them to be of a kind and proposed to solve them along the same lines²⁰. His solution is based on a rigid predetermined classification of sets, which makes room only for quantifying over sets of the same type, not over sets in general. The hierarchy is fixed. It is incorporated into language itself. Axiomatic set theory, say ZF, has disposed of the built-in hierarchy, replacing the aggregate of concepts set-of-type- i by a single concept of set.

Russell's hierarchy of sets parallels his hierarchy of propositions, of which Tarski's hierarchy of metalanguages is a close relative. The representation of Tarski's hierarchy in a formal system yields an aggregate of truth predicates true_i , where i ranges over some list fixed in advance, but no single true. Viewed from that perspective, adequate formal systems that contain their own truth predicate should do for semantics what ZF does for intuitive set theory.

Now, axiomatic set theory is a system in which mathematics can be rigorously formalized. And, as long as mathematical statements are interpreted in structures that are sets, their truth-values can be defined within set theory by the usual (Tarskian) truth definition. A closer look shows that the metalinguistic hierarchy is thus subsumed under the set-theoretical hierarchy, obtained by iterating the power-set operation. The problems resurge when the structures in question are no longer sets; in particular, when truth in "Cantor's universe" is considered. In this respect, axiomatic set theory has not solved the problems arising out of the set theoretical paradoxes. It has shifted them to the theory's fringes; where they are of no concern, except to set theoreticians who investigate the theory's borders and who try to express the "true picture of Cantor's universe" by adopting strong infinity axioms²¹.

Given this perspective, it is natural to try to construct highly expressive formal languages containing their own truth predicate, wherein the presence of black holes is minimal. Unlike set theory, pointer semantics draws on insights derived from natural language rather than mathematical thought. Nonetheless, it results in systems that are as formally precise as any piece of mathematics. The present work can be seen as a first step that eliminates black holes at the basic linguistic level. If we include in the language an arithmetical machinery that makes possible a syntactic representation of the language within itself, à la Gödel, and

²⁰More recently, the analogy has been emphasized by Parsons, [1974]. In [Gaifman 1983] I discussed both types of paradoxes as cases of what I called an infinity paradox: a contradiction that results from an attempt to construct self-containing systems.

²¹This point is elaborated in [Gaifman 1983].

if, treating pointers as numbers, we include also means of describing in the language the pointing relation, then black holes can be constructed in a variety of ways. Attempts to eliminate them lead to more sophisticated versions of the algorithm and to some intricate, interesting questions. Thus, pointer semantics, which formalises mechanisms inherent in linguistic practice, gives also rise to foundational questions of logic.

3 Tokens and Types

Let me start by a commonplace observation: when a child learns a language he encounters first of all tokens of linguistic entities. The displaying of tokens under certain circumstances may affect the environment in which the displaying takes place. The child acquires the language by learning the effects of token-display and by participating in the token-display game. This observation does not prove that tokens are prior (in some nontrivial sense) to types; for the ability to learn the language is preconditioned by the ability to recognize token similarity and, thereby, to classify tokens according to type. The observation is made in order to point out the obvious fact that there are many ways of passing information by means of an interactive game-like process. A useful description of the game must, inevitably, ignore through abstraction various aspects. Else we shall be submerged under sheer detail²². Of course, we may ignore only inasmuch as this does not restrict us essentially when it comes to making sense of what is going on. There are thus two conflicting requirements: That the account we give be sufficiently simple, and that it be adequate for handling a broad enough class of applications. Usually there is trade-off between the two.

The fixing on sentence-types as on the basic truth carrier units seems an obvious natural move. Besides making for the enormous simplification of abstracting away tokens, it clears the path for a compositional semantics. Linguistic entities are analysed thereby as complexes whose meanings derive in a systematic way from that of their components, via certain, relatively few recursive rules. The truth-value of a sentence is thus determined recursively; the context in which the sentence occurs can be completely ignored. A description of this kind fits accurately formal languages constructed by logicians; which is no wonder, since it is the very principle on which such languages are constructed. The picture does not fit natural languages, where context dependencies abound, due to indexicals, demonstratives, and ambiguous terms. Nonetheless, it reflects a very essential aspect of natural language. For if the dependency of truth-value on context can be traced to context-sensitive particles (noun-phrases and predicates) then, once these particles have been disambiguated | how, is another matter | we can define the truth values by using the above mentioned recursive rules²³.

I will now show that such a picture cannot account for those aspects of language that

²² On a more fundamental level: the very concept of description means that many details have been ignored.

²³ Thus, Davidson's program for setting up a meaning theory for natural language adopts a Tarski-like view.

revolve on its containing its own truth predicate. To draw it in sharper detail, assume p and q to be dissenting sentence-tokens, i.e., tokens of the same sentence-type whose truth-values differ. Then, according to the picture, the dissent is due to the presence of certain terms in the sentence, say $t_1; t_2; \dots; t_n$, whose references differ on the two occasions. If we are to replace each t_i by a suitable t_i^0 , which denotes on both occasions what t_i denotes in the context of p , the resulting tokens, say p^0 and q^0 , should have the same truth-value, namely that of p .

Let us call such an elimination of context-dependency local disambiguation.

The terms t_i can be either noun-phrases or predicates. I assume here, for the sake of simplicity, extensional contexts. But this is not essential; local disambiguation makes sense in intensional settings as well²⁴. The "context-insensitivity" of the terms t_i^0 means that they have the same denotations throughout some wider context, which includes both that of p and of q . It does not mean that they are insensitive to context altogether

Well known common examples illustrate how local disambiguation works:

When 'I am tall' is uttered by two speakers, only one of whom is tall, the conflict in truth-value is resolved by using proper names. If needed, a name can be introduced by stipulation: 'Let S1 be the first speaker'. Now, the counterfactual situation, where the two speakers use 'S1' instead of 'I', the two utterances have the same truth-value: that of the first speaker's original utterance²⁵.

Local disambiguation applies to all cases of indexicality (including demonstratives) and to cases of ambiguous reference, in general: for example, two tokens of: 'Aristotle was married to a woman called 'Jacqueline'', the first in an answer to a quiz question about shipping magnates, the other in a paper about ancient philosophers. Here, replacing 'Aristotle' by 'Aristotle Onasis' would do the redressing^{26 27}.

Let us now go back and see whether such an account can fit the situation arising from the use of 'true'.

I take it for granted that our Two Line puzzle is a case of token dissent, that we do want to say that the sentence on line 1 is not true but the one on line 2 is and that, in the Jack and Jill exchange, we side with Jill against Jack. Or, to use Buridan's example²⁸, if Plato

²⁴The difference being that some stronger notion of equivalence is called for: The "context-insensitive" t_i^0 should be intensionally equivalent to t_i -in-the-context-of- p .

²⁵Instead of changing, counterfactually, the original utterances we can let the two speakers re-utter the sentence, with 'I' replaced by 'S1'; the natural assumption being that, in the given circumstance, an utterance and its repetition by the same speaker are equivalent. This strategy does not work if the indexical is time-sensitive, unless the repetition follows sufficiently closely, so as to make the time difference insignificant.

²⁶In this case the tokens can be actually changed.

²⁷In lumping together indexicals with cases of ambiguity I am not glossing over what distinguishes the first. A systematic account of how indexicals work can be given (e.g., [Kaplan 89]); the ambiguity of 'I' means that its denotation shifts according to context, but because the shift is systematic, 'I' can be regarded, in one sense of "meaning", as having a fixed meaning: it always denotes the speaker. No such account exists in general. Perhaps in the general case we can do no better than what informal pragmatics will allow us.

²⁸Quoted in Burge [1979].

says, at time t : "What Aristotle says at time t is not true" and Aristotle says at time t the same about Plato, then both sayings are not true; but if a third party were to assert at time t that what Plato (or what Aristotle) says at time t is not true, his saying would be true.

Evidently, 'The sentence on line 1 in the Two Line puzzle refers in both tokens to the same object, be this sentence-type or sentence-token. The only possible source of discord is the predicate 'true'. And if this is to be resolved by local disambiguation, then 'true' in the context of line 2, denotes something different than 'true' in the context of line 1. But note that in our argument for the non-truth of the line 1 sentence we employed 'true' in the very same sense of the 'true' occurring on line 1.²⁹ The argument simply shows that the sentence on line 1 does not fall under what 'true' in this very same sentence denotes. And if this is what we write on line 2, then the 'true' on the second line has the same denotation as that on the first. (Indeed, if the 'true' on line 2 is construed as being on a higher level, then the sentence on line 2 is false!) This shows that the difference in truth-value cannot be accounted by indexicality.

Interestingly, the analysis just offered accords with that given by Burge [1979] in his discussion of a similar puzzle. Phrased in terms of the present puzzle, his conclusion is that the tokens of 'true' on the two lines should be construed as standing for the same truth-concept. The difference in truth-values is traced by him to different implicatures involved in evaluating the sentences. When we evaluate the line 1 sentence on the same truth-level as the 'true' that occurs in it, say level i , we find it non-true, i.e., non-true $_i$; but when we evaluate the line 2 sentence on a higher level, k , we find it true, i.e., true $_k$. But, Burge remarks, the difference between the tokens is illusory. Both are not true $_i$ and both are true $_k$.

This, by the way, shows how the construal of 'true' as a predicate over types leads naturally to reading the Liar as ambiguous. On my proposal, there is no ambiguity when tokens, or pointers, are the truth-value carriers; but this does not concern us right now. What is crucial for the present argument is that, whatever the alleged ambiguity and whatever the token-dependence of truth-values, it does not derive from the 'true' that occurs inside the sentence, but is a product of shifts that we from the outside effect when we say that the sentence is, or is not, true.

Some further elaboration will be given in the discussion of Burge's work, but I think that the point about indexicality is already clear as it is.

The foregoing argument rests on the way by which the non-truth of the line 1 sentence is derived and on what we intend to state in the sentence on line 2. But one can also argue, in general, as follows.

Let the 'true' on line 1 be changed to a non-indexical 'true^a' that denotes once and for all what 'true' in the original context denotes³⁰. We thus freeze 'true' in its the line-1 context.

²⁹ The argument rests on (T1) of the 'Preview': If 'x is not true' is true, then x is not true. (And in this phrasing the same truth concept, whatever it may be, is denoted by the various occurrence of 'true'.) The familiar argument now runs thus: If the line 1 sentence falls under what 'true' in the line-1-context denotes, then by (T1) that sentence does not fall under what 'true' in the line-1-context denotes. Hence this sentence is not true, where 'true' means what it means in the line-1-context.

³⁰ Either the change is effected by erasing and rewriting, or we imagine a counterfactual situation in which

Such a change, one feels, should not affect anything that has to do with truth. Saying 'you are tall', while addressing Jack, and saying 'Jack is tall' come to the same.

There is yet a subtle complication: By changing 'true' to 'true^a' we have altered the reference of 'the sentence on line 1'. There is thus the additional difference in what that noun-phrase denotes. Nonetheless, as far as truth or non-truth is concerned, this added difference should not matter. It might have, had the sentence said something about lexical properties. Changing

'You are tall and this last utterance of mine is short'
made by Jill while addressing Jack, to

'Jack Horatio MacWilmington the Third is tall and this last utterance of mine is short'' makes a difference. But with 'true' instead of 'short', the two come to the same. True, the analogy is not altogether complete. Having the changes in two separate conjuncts ('you are tall' and 'this last utterance of mine is true') we can argue formally by considering first one, then the other. Such a procedure is not at hand when the changes are fused in the same atomic sentence. But this, I think, cannot provide any ground for taking the change of 'true' to 'true^a' as a cause for truth-value change. Whoever has misgivings on this score may consider a variant of the Liar in which the sentence is, so to speak, split:

Line 0: The sentence on line 1 is true.

Line 1: The sentence on line 0 is not true.

Can the freezing of 'true' on line 1 make a difference, for either the line 0 or the line 1 sentence, as far as truth is concerned? I think the answer is clearly no. Neither should an additional freezing of 'true' on line 0.

Thus, in the Two Line puzzle, our ruling on the non-truth of the line 1 sentence carries over to its disambiguated ^a-version: the ^a-version is not true. Moreover, it is also not true^a; for we can derive this in the familiar way from T1 of the 'Preview' (cf. footnote ??); T1 being a minimal requirement that truth^a, as any truth-concept, should satisfy. When we write this true conclusion on line 2 we have repeated the ^a-version of the line 1 sentence. The elimination of indexicality gives us back two dissenting tokens. Local disambiguation fails.

That truth is indexical is motivated by the idea of truth-levels; in particular | by the view that there are many truth concepts. That view may suggest that paradoxical phenomena are failures due to ambiguity. Which might further suggest the indexical model | this being the classical case of systematic shiftings of reference and the only one for which a precise treatment is at present available.

While my proposal breaks ω with the Tarskian model, it provides, as we saw in the 'Preview', for a precise formal reconstitution of the notion of levels. We might thus say that judging the line 2 token takes place at a higher level than the judging of its line 1 twin; for the jump rule is needed for evaluating that token but not for evaluating the one on line 1. A level-assignment to 'true' can be obtained as a correlate of the general assignment of the second version is written instead of the first.

ordinal numbers to pointers. The prescription is simple: Assign to the occurrence of 'true' in a token of 'true(p)' the ordinal number assigned to p; for this is the level that the evaluation of p reaches. In particular, the 'true' on both lines is assigned level 0 | the number assigned to the line 1 token. But were we to write on a third line 'the sentence on line 2 is true' (or 'not true'), that 'true' would get level 1. Thus, the shift to higher level is not in what the 'true' inside the tokens denotes, but in the way that we from the outside judge them.

Occasionally, truth-levels are indicated by attaching subscripts to 'true'. This would be harmless, provided that we are not misled to see in these subscripts, or indices | as they are also called, some form of indexicality. The indices play no role in determining truth-value, but come after the fact, as a kind of annotation to a finished process, an underlining of certain aspects of the evaluation. While the reference of 'I' is picked, through the indexicality mechanism, from the multitude of humans, nothing of the kind obtains in the case of 'true'. There is no preexisting multitude, or hierarchy, of truth concepts, out of which we pick the concept we invoke. The hierarchy is created by the evaluation process itself, by the very act of judging, by the linguistic exchange.

The failure of local disambiguation reflects that aspect of linguistic activity whereby sentence-tokens function as the units of the exchange. The meaning of a token is not reducible to what the sentence says, even when the denotations of its terms have been fixed. It derives also from the token's particular place in the network.

The principle, under other forms and on different scales, is quite common. An event, an assertion, a work of art might mean whatever it means by virtue of its place in a global system of similar events, acts or works. Duchamp's famous urinal refers to the whole previous tradition of art exhibitions. The reference, though not formal and not explicit, is there. For the place of that particular display within the historic network of art is what endows Duchamp's exhibit with artistic meaning. Moreover, it would be erroneous to derive this meaning from the urinal itself, as if it were an "indexical" that indicates different things in different contexts. Quite the contrary. That object means or connotes, within the displaying event, whatever a urinal means or connotes in everyday life. Only by retaining its ordinary, undignified status, can it cause the surprise, the outrage and the joke, which constitute the aesthetics of the display.

We can now take stock and assess the extent to which pointer semantics departs from a type-based one. The formal system underlying the present proposal excludes indexicals, or any kind of ambiguity. Hence we can take the extent of pointer dissent (i.e. of pointers to the same sentence-type that get different truth-values) as a sort of measure. Now, the results LPD and HLD mentioned in the "Preview" yield a very simple picture, perhaps surprisingly simple | given that it holds for full first-order logic and arbitrary pointer networks: Whenever pointers dissent, one of them has the value GAP and the other, which has a standard value, is one level higher in the linguistic hierarchy. Informally, we can say that departure from type-based semantics occurs just when a pointer fails to "express something", while another

at a higher level succeeds in expressing it³¹. Thus, departure takes place when a necessary ascent in the linguistic hierarchy cannot be accommodated within Tarskian semantics or any of its offshoots; and here I include Kripke's system and all its variants. In such systems the ascent can be represented only as a change of language. Its accommodation within a single language, which is what the present framework does, requires a price and the results show that we are paying the minimum³².

While the semantic paradoxes have focused a great amount of research, resulting in an extended and often sophisticated literature, they have not been at the centre of the research program of natural language semantics. This is understandable, given the tangle of problems encountered in the rather elementary stages of the investigation. To give the semantics of 'true' while we are far from clear on that of 'event', or that of mass terms, might seem a luxury³³. But the tendency to regard the problems raised by the paradoxes as something esoteric, to be handled later (if at all) when more basic aspects have been clarified, is, I think, an error. We are not concerned here with marginal brain-teasers, but with an essential feature of language. 'True' is the paradigm of a network-creating predicate. It can send us, via 'what she said', 'whatever he asserted', 'what is written on page 1' and their like, from John to Joan, from Joan to Jack, to Jill, to McX, and so on; the tour is unrestricted and sometimes endless. 'Know' (in the sense of 'know that') and 'believe' (in the sense of 'believe that') have similar potential: 'I know what she knows', 'what he believes I doubt'. But 'true' is the more fundamental³⁴. It is no wonder that problems arising out of this chain-forming capacity have surfaced in the theoretical research of Artificial Intelligence. Once it is realized that knowledge, or belief, are to be modeled as predicates rather than as sentential operants³⁵, the semantics of 'true' becomes relevant in the contexts of knowledge and belief representation³⁶.

³¹In this respect, all such cases are like the simple Two Line example. However, the sentence in question can be as complex as any first-order sentence and the failure can be caused by an extremely intricate loop. Moreover it can occur at any level, the ordinal of which can be of any cardinality not exceeding that of the total network.

³²I should, however, add that it is conceivable that in more sophisticated variants of the algorithm LPD will fail. I see no reason why LPD, pleasing as it is, should be imposed as a prerequisite on evaluation procedures in general. With indexicals present, different tokens of the same sentence may get different non-GAP truth-values. That a similar phenomenon may arise because of deeper reasons, deriving from the use of 'true', is not to be ruled out.

³³Using the concept of truth in order to give semantics in the form of truth conditions, as was proposed by Davidson, is of course quite different from giving the semantics of 'true' itself.

³⁴Note how often 'know', and to some extent 'believe', ride on the back of 'true'. We might say 'I know that what she says is true', but we cannot say 'I know that what she says', while 'I know what she says' means something different. Knowledge-modalities like $K_J(x)$, which are used in formal systems, are rendered into natural language by invoking 'true' or an equivalent expression: 'J knows that x is true' or 'J knows that x is the case'.

³⁵Sentential operants are too restrictive: 'John knows something that Bill does not know' is inexpressible if 'know' is treated as a sentential operant.

³⁶Thus, Morgestern [1986] and Kremmer [86] draw on Kripke's model, while Asher and Kamp, [1986], follow the constructions of Gupta and Herzberger. Among other works, motivated by problems of self-

4 Apropos Propositions

Some would have it that truth and falsity attach to propositions (or statements) | entities from which linguistic particulars have been abstracted away. There is, however, no agreed view on the nature of these objects and the difficulties surrounding the concept may incline us to consider, at least provisionally, less problematic creatures. In any case and however propositions are conceived, the crucial question is that of relating them to linguistic constructs; to sentences, for example. On the present view, the question takes the form: what proposition, or statement³⁷, is expressed by a pointer? I have suggested in the "Preview" that, prima facie, a pointer can be taken to express whatever is expressed by the sentence it points to, except for failed pointers | those marked by GAP. On this suggestion, non-failed pointers to the same sentence express the same thing. Since (by LPD) such pointers get, invariably, the same truth-value, we will not run into trouble, as far as truth-values are concerned. The plausibility of the suggestion rests, however, not on LPD, but rather on its proof, which shows the following: Whenever p and q are non-failed pointers to the same sentence, their evaluation requires the same sequence of necessary steps, ending at a stage where each can be assigned the same truth-value by applying the same rule³⁸.

All this is a far cry from determining when different pointers express the same thing; the criterion given is sufficient but certainly not necessary. But it seems that an account that correlates propositions with sentences could be extended to pointers. The heuristics would be as follows: Two non-GAP pointers express the same proposition, inasmuch as this can be said of the sentences they point to.

The propositions to be gotten along the suggested lines are "ontological", rather than "epistemic". A person stating what to him is a perfectly clear thought might fail to express a proposition, because unwittingly he has involved himself in some loop. A similar phenomenon

reference in epistemic contexts, are [Asher 88, 90], [Koons 88], [Perlis 85], [des Riviers and Levesque 86] and [Thomason 86].

³⁷ There is considerable overlap in the use of the two terms and I am not altogether certain as to the "right" one. There is a usage, which I do not follow, whereby propositions are taken as the meanings of declarative sentence-types, including indexical ones. By contrast, statements are attached to particular utterances. However, in the tradition deriving from Frege and Russell, the proposition expressed, say, by 'it is now raining' changes according to time. Read thus, propositions can be associated with sentence-types only after the sentences have been converted, by elimination of context-sensitive items, into "eternal" versions. What is then the difference between this notion of a proposition and the notion of a statement as advocated by Austin and Strawson? It appears that propositions are pre-linguistic entities, existing in their own right, whereas a statement is something one makes by producing a linguistic token. Put bluntly, if there were no people, there would have been no statements but there would have been propositions. Stripping them of their more metaphysical aspects, one can say that propositions are possible statements. And this is how I propose to view them here. I have opted for 'proposition', because usually this, rather than 'statement', is employed when a formal treatment is intended. Thus, in [BE 87], a formal model of so-called Austinian propositions is proposed for what Austin called statements. Indeed, if propositions are possible statements, then the general theory of statements is the theory of propositions.

³⁸ Were we to use another version of the evaluation procedure, a similar analysis of the procedure would be required in order to justify stipulations about "expressing the same thing".

obtains in some current theories of propositions, whereby certain terms (the so-called directly referring ones) are taken to introduce their very references as components of the proposition³⁹. Mistaken beliefs about the reference may lead to a person's expressing a proposition different from what he intended. And when a term of the directly referring kind fails to refer, no proposition is expressed.

But nothing prevents us, at least not in principle, from considering other kinds of propositions, aimed at reflecting finer shades of meaning. Just as, for various purposes, it is natural to hide the reference-picking machinery of certain terms, it might be desirable, for other purposes, to expose it in the proposition. What propositions are depends on what dimensions of meaning they are supposed to reflect. The question what are the right propositions is as meaningful as the question what sort of items a map should display, or what is the right granularity of a film.

In the present case, a move to more epistemic propositions might entail distinctions between various gap pointers. The gap constituted by the line 1 sentence derives from a contingent fact: the place where the sentence is written. Someone might have put it there because of a mistaken belief about line numbers⁴⁰. By contrast, the loop produced by 'What I am saying at this very moment is not true' cannot be amended by moving utterances to different locations. Whatever the speaker might have intended (assuming he knows the language) it is not expressed by the utterance taken at face value. In the present formalism the two cases are not distinguished; both are construed as a pointer, p , which points to $\text{Tr}(p)$. The distinction can be introduced at the cost of additional structure; for example, a set of possible pointing functions in addition to the actual one⁴¹.

Finally, it can be argued that some meaning is expressed even in the case of non-contingent gaps. For a pointer provides a prescription for computing truth-values. It amounts to a kind of algorithm. And as an algorithm it makes sense even when it leads inevitably to a non-terminating computation.

While not endorsing, at least initially, any particular position concerning propositions, the present proposal incorporates a certain methodology: Go into propositions after having clarified sufficiently the rules and mechanisms governing truth-value assignments. Only then can we tailor our propositions, if and when we introduce them, to fit the desired aspects of meaning. To do otherwise is to commit ourselves to a preconceived concept of propositions, deriving from some metaphysics or some given frame, prejudging thereby the issues at hand. An opposite approach underlies the work of Barwise and Etchemendy, [1987], on some aspects of which I shall now comment briefly⁴².

³⁹Cf. [Kaplan 89].

⁴⁰In an example of Prior [1961], used by Burge, a student who thinks he is in room 9 writes on the blackboard: 'Whatever is written on the blackboard in room 10 is false', expressing thereby his true belief. It turns out that he himself is in room 10.

⁴¹Distinctions of similar nature, between kinds of pointing, are used in certain more sophisticated variants of the evaluation procedure, aimed at reducing black holes.

⁴² An already condensed passage concerning their work, had to be further abridged for reasons of space. An evaluation of their work, which naturally should address the wider perspective of situation semantics, is,

Barwise and Etchemendy propose formal models of propositions, constructed according to situation semantics and classified into Russellian propositions and so-called Austinian ones (the second type is the one they prefer). One of their goals is to include in the model what they call cyclic propositions: those involving self-reference. And their ingenious idea is to employ, for that purpose, AFA (set theory with Aczell's Anti-Foundation Axiom) | a theory of non-well-founded sets which accomodates, naturally, a variety of \cyclic creatures".

While the BE (Barwise-Etchemendy) framework assigns truth-values to propositions, it does not provide guidelines, let alone a procedure, for assigning truth-values to sentences | either types or tokens. For when it comes to determining what proposition is expressed by an utterance, a kind of free parameter enters⁴³. In cases of self-references there is considerable latitude; much is up to the interpreter, who can toggle truth-values by adjusting that parameter in various ways. Such utterances are thus construed as ambiguous. Presumably, Barwise and Etchemendy regard the question of disambiguation, which will lead to the determination of truth-values, as belonging to some branch of pragmatics.

It can be shown that the assignment of truth-values, according to intuitions formalized in the semantics of pointers, is outside the scope of the BE framework. For the needed distinctions cannot be made in AFA⁴⁴.

Can we correlate with sentence-tokens propositions, of the BE kind, whose truth-values match the values derived by pointer semantics? This is possible to some extent. In general, it requires the removal of certain restrictions underlying the BE modeling of propositions. Perhaps this is an artifact of the BE modeling; in any case it is a technical point to which I shall not enter here⁴⁵.

The major non-technical point on which the BE framework is at odds with pointer semantics is that it does not make place for the notion | central to the present proposal | of a failed utterance. The BE setup provides the Liar and its kin with circular propositions, which I do not find objectionable, as long as we mark such propositions by GAP. However, Barwise and Etchemendy, construing 'false' as false₂ (i.e. as not being true) regard these propositions simply as being false. I argued in the \Preview" that this cannot be considered more than a terminological blurring of truth-value gaps. It is also, I think, at odds with

of course, not intended here.

⁴³Namely, the situation (in the technical sense of situation semantics) associated with the utterance, whose choice is left to us.

⁴⁴For example, AFA does not provide for the distinction between the sentence-tokens on lines 1 and 2 in the Two Line puzzle; the two nodes in figure 2 of the \Preview" represent the same set, namely: the unique set containing 1 as its sole member. This is forced by the extensionality axiom.

⁴⁵In the BE framework propositions are about situations, the choice of which involves a considerable degree of freedom, especially in self-referential cases. However, in disjunctive (or conjunctive) propositions, all the disjuncts (or conjuncts) must be about the same situation | which is also the situation the whole disjunction (or conjunction) is about. This can imply counterintuitive consequences: In a disjunction $S_1 _ S_2$, where S_1 is a Liar sentence and S_2 is a sentence saying that S_1 is not true, both disjuncts must be about the same situation, which causes the second disjunct, and along with it the whole disjunction, to be non-true. In the evaluation procedure this disjunction (i.e., a token, or some pointer to it) gets the value T because the second disjunct does.

Austin, who devoted much effort to the taxonomy of failures and who distinguished failures caused by illegitimate self-reference from mere self-contradiction⁴⁶ Failure to distinguish failure can be a source of confusion. I think that it gives rise to some philosophical red herrings in the BE framework; but for reasons of length I cannot elaborate on this here (cf. footnote ??).

5 Three Previous Accounts

I shall discuss three accounts related to my proposal, those of Parsons, [1974] which can be seen as a development of a theme in Russell [1908], Burge [1979], and Skyrms [1982] and his earlier 1970 papers. These accounts do not assign truth-values to sentence-tokens, but they share with the present work the view that places the Strong Liar and its kin at the heart of the semantic paradoxes. Needless to say, I have no intention to review the vast literature devoted to the Liar. The choice of these works is dictated by their relevance to my proposal and reflects no attempt at ranking.

Ambiguity and Interpretation Schemes

Parsons proposes to view the Liar and its kin as cases in which different interpretation schemes, based on different truth concepts, apply to the same sentence (type). No formal explication is given of "interpretation scheme", but a certain direction is suggested by rewriting 'A is true' as: 'there exists a proposition P such that A expresses P and P is true'. The interpretation scheme is then determined by fixing the domain of propositions over which the quantified P ranges. This proposal goes back to Russell [1908], who suggested that the paradox arises from an attempt to quantify over "too many propositions" and is to be resolved by restricting the quantifier ranges. This solution underlies his theory of types and blocks also the set-theoretical paradoxes. Since there can be many natural domains of propositions, the sentence turns out to be ambiguous. Whatever the domain, it cannot include the proposition expressed by the Liar sentence itself.

Intuitively, the interpretation scheme reflects the comprehensiveness of the point of view from which the assignment of truth-value is made. In the case of our Two Line example, the sentence on line 1 fails to express a proposition, under the "first straightforward" interpretation; but under a second scheme, which takes into account the previous failure, the sentence is seen to be true. Parsons does not give an account of how the interpretation is determined, except for noting that it is determined by the circumstances of the sentence's utterance.

⁴⁶ "Those of the so-called 'logical paradoxes'... which concern 'true' and 'false' are not to be reduced to cases of self-contradiction, any more than 'S but I do not believe it' is. A statement to the effect that it is itself true is every bit as absurd as the one to the effect that it is itself false. There are other types of sentence which offend against the fundamental conditions of all communication in ways distinct from the way in which 'This is red and this is not red' offends..." , in [Austin 1950], footnote on page 129 in the Oxford University Paperback edition.

He proposes to regard this ambiguity in the same light as the ambiguity surrounding the interpretation of quantifiers or the use of expressions such as 'say' or 'mean'.

Given the analysis of section ??, that ambiguity is now seen to derive not from some ambiguity of 'true' that occurs in the Liar sentence, but from the different points of view that we| the interpreters| can adopt from the 'outside' with regard to it. Parsons suggested, in a 1982 postscript, Burge's indexical conception of 'true' as a way of explicating the interpretation-scheme picture. But it appears that this would not do justice to his ideas. Indexicality falls short of accounting for the phenomena at hand. Indeed, as the analysis of Burge's setup will show, the real work there is done by assuming some mechanism of implicatures.

Now, pointer semantics can provide a kind of backup for the concept of interpretation scheme. For one can regard different pointers to the same sentence as expressing various interpretations of it. There remains however the essential difference that my setup is based on construing 'true' as a predicate over tokens (or, more generally, pointers) and, doing so, it leaves no place for ambiguity. Truth-values are determined by the evaluation procedure. The token on line 1 is definitely not true (and not false either) the token on line 2 definitely is.

There are other systems that can provide formal backups for the ambiguity picture. The constructions of Gupta [1982] and Herzberger [1982], whereby the Liar changes its truth-value according to the evaluation stage, give rise to such a picture. And the more recent [BE 87], discussed above, provides another modeling of ambiguity. It is indeed directly related to the Russellian idea, for it models, among the rest, so-called Russellian propositions. (As can be expected, no indexicality of 'true' plays a role in any of these setups.)

While these models do not solve the Two Line puzzle, they solve or illuminate what we might call the One Line puzzle:

A philosopher contemplates the sentence on line 1, which says of itself that it is not true. At first he deduces that the sentence is not true; realizing on second reflection that the sentence itself 'says this', he deduces that the sentence is true after all. He may continue the process to the third round, whereby the sentence becomes non-true, then in the fourth round it is true again, and so on.

Burge's Proposal

I have already discussed in section ?? the indexical construal of 'true'. Here I shall confine myself to more specific details of Burge's work. His proposed setup comes in two parts. The first consists in a formal language containing a family of truth predicates, $truth_i$, $i = 0; 1; \dots$, arranged in a Tarski-like hierarchy, with subscripts denoting the levels. Each predicate is to serve as a truth predicate for the portion of the language 'below it', that is to say, for all sentence-types containing truth predicates with a strictly smaller index. Yet each predicate takes as arguments all sentence-names of the language. Because of self-reference there are, for each i , sentences that inevitably fail to get a $truth_i$ -value: neither the sentence nor its negation are $truth_i$. The idea is to regulate the assignment of $truth_i$ -values so as to approxi-

mate, as far as possible, the properties required of a truth predicate, while maintaining the predicate's position in the hierarchy. Burge stipulates various plausible axioms; for example, axioms to the effect that if A is true_i , then A , as well as $\text{true}_i('A')$ are true_k , for all $k \geq i$.⁴⁷ Apparently, the proposed axioms are not sufficient⁴⁸. But this is a technicality that can be fixed. There is, in fact, a neat generalization of Kripke's construction to languages containing a hierarchy of truth predicates, which achieves the desired result in a simple straightforward way. It is mentioned briefly in [Gaifman 83]⁴⁹. We can therefore assume that truth _{i} -values have been assigned in a satisfactory way to all sentences and all i . Call this language the stratified language. (Note that a Liar_i , i.e., a sentence saying of itself that it is not true_i , is not true_i and neither is its negation. But a Liar_i is true, when evaluated from the outside, via the usual truth definition. Since each true_k behaves like an ordinary truth predicate for the portion of the language below it, a Liar_i is true_k for all $k > i$.)

The second part of Burge's proposal is to rewrite the ordinary sentences within a given discourse as sentences of the stratified language; this is done by "indexing" | i.e., by attaching subscripts to the occurrences of 'true'. No precise rules are given; Burge sees it as a pragmatic question and provides certain informal guidelines⁵⁰. The recourse to informal pragmatics has brought the proposal under severe criticism by Gupta⁵¹.

However, the main trouble with Burge's proposal is not the imprecise, pragmatic principles for indexing, but that the indices do not do the desired work. An indexing can only lead to truth _{i} -values; it does not determine at which level a given sentence is to be evaluated: Which true_i are we employing when we say of a sentence A | written already in indexed form | that it is true, or not true, or false? Burge invokes here what he calls a pragmatic implicature: Consider the evaluating sentence | the sentence asserting the truth or non-truth of A ; index the 'true' in that sentence. That index is the level at which A is evaluated. This however is only the beginning of the story; for occasionally that implicature is disabled

⁴⁷Actually, Burge considers various sets of axioms, but this need not concern us here.

⁴⁸The following is, I think, not derivable: If : $\text{true}_i('A')$ is true_i then A is not true_i .

⁴⁹I found it when, independently of Burge's work, I was considering formal languages that incorporate Tarskian hierarchies. Here it is. Start by evaluating all sentences by the standard rules, treating each of the predicates true_i as a truth predicate. The standard rules consist of the usual (Tarskian) truth conditions (applied, as in Kripke's construction, via Kleene's strong three-valued logic) as well as the following rule: If A is true_i , then $\text{true}_i('A')$ is true_j for all j (including the case $j < i$!). Having done this, extend the resulting valuation as follows: For each A which is not true_0 , declare the sentence : $\text{true}_0('A')$ to be true_i , for all $i > 0$. Now repeat the same process of applying the standard rules, but only to the truth predicates on level > 0 . Thus, by the end of the first stage the extension of true_0 is determined. At the end of the second stage, for each A which is not true_1 , declare : $\text{true}_1('A')$ to be true_i for all $i > 1$. Again, repeat the process for all predicates at level > 1 . And so on, until we run out of truth predicates.

⁵⁰A principle called Verity instructs us to assign indices in a way that maximizes the applicability of the truth defining schema, that is to say | in a way that minimizes gaps. A less important principle called Justice says that the assignment should not prefer one statement to another, unless for some good reason. Thus, if each of two persons asserts of the other that he is not saying the truth, both should be given | for reasons of symmetry | the same subscript, resulting in both being gaps. Either principle is subject to a ceteris paribus clause.

⁵¹[Gupta 82], page 204 in Martin's collection.

and different one obtains. Assume, for example, that 'true' in the line 1 sentence in our old puzzle⁵² has been indexed by i . This sentence is an evaluating sentence for itself (recall, that here we are concerned with sentence-types). By the initial implicature, when we assert that the sentence is not true we are in fact asserting that it is not true $_i$; which, indeed, it is not. But when we go on and evaluate this very assertion, the same implicature rules again that we use 'true $_i$ '. We should therefore judge our assertion, written say as a sentence on line 2, as non-true. This unwelcome result is avoided by having the implicature cancelled and raising the level of the second evaluation to some k , greater than i . As Burge sees it, the move from line 1 to line 2 signifies a switch from one implicature to another. The change of truth-value is only apparent, both are true $_k$ and not true $_i$.

What Burge fails to observe is that the resort to implicatures voids the indexing of its significance. Not only the Liar, but all problematic cases of self-reference, require implicature switches. Thus, in the Plato-Aristotle example quoted from Buridan (cf. section ??), the 'true' in both sayings gets the same index (a conclusion endorsed by Burge on the grounds of symmetry) resulting in both not being true. If a third party were to assert that Plato's, or Aristotle's, saying is not true, his saying would be true. But this can be sustained only if the implicature employed in judging the first two assertions, is changed when judging the third. Or take the Nixon-Dean example, where each says of the other that all his utterances concerning Watergate are not true. If, except that assertion, all other Watergate assertions made by either are not true, a loop is created and we can say that these two assertions are not true. Again, the implicature must be changed when judging our assertion; because when the Nixon's and Dean's sayings are spelled out as conjunctions, our assertion is seen to repeat a conjunct from each.

In fact, whenever the jump rule is used in the evaluation procedure, an implicature must be canceled in favour of a different one that shifts the level of 'true' in the evaluating sentence. Since the shifts are cumulative, a count must be kept to determine the 'right implicature' for choosing the level; and the level can be any finite or infinite ordinal. Burge ignores all this. Having introduced implicature in the analysis of the Liar, he goes on in the second half of the paper to discuss the stratified language and indexing, without mentioning the matter again.

The process of implicature shifts is altogether obscure; no guideline, nor any indication, is given. But there lies the real problem; for knowing the mechanism that governs implicature choice is almost like knowing the evaluation procedure. It is this mechanism that does the real work. Indexicality rests idle.

The Intensional Conception

The analogy between an intensional case and a Strong Liar, on which Skyrms draws, is as follows⁵³: That the sum of 7 and 2 is odd is necessary, that the number of planets is odd is

⁵²The example used by Burge is somewhat different, but the difference does not matter here.

⁵³I use our Two Line puzzle; the difference between it and the example used in [Skyrms 82] does not matter here.

not| even though the sum of 7 and 2 is the number of planets; just so, the sentence on line 2 is true, the sentence on line 1 is not| even though the sentence on line 2 is the sentence on line 1. Skyrms proposes to solve the Strong Liar by construing 'true' as intensional. Truth-values are to be assigned to sentence-types under descriptions. Under the description 'the sentence on line 2' the sentence is true, under the description 'the sentence on line 1' it is not.

Formally, Skyrms proposes to assign truth values to sentence-description pairs, $(s; d)$, where d is a description of the sentence-type s , and to do so in a way analogous to Kripke's fixed point construction. He shows that, given a partial valuation of sentence-description pairs (that may distinguish between pairs that differ only in the description coordinate), a variant of Kripke's construction can be used to extend it. Skyrms' construction falls short of the desired goal since, being based on (the analogues of) Kripke's rules, it does not have the gap and jump rules, which are crucial. It works as long as we are moving through standard values; e.g., if we have already assigned $(s; d)$ the value T , we can go on and assign $(s^0; d^0)$ the value F , where s^0 is the negation of s and d^0 is some appropriate description obtained from d . But it does not provide a way for assigning T to the token on line 2 (i.e., to the pair $(s; d)$, where s is the sentence and d is the description 'the sentence on line 2'). In general, it falls short whenever the jump rule is invoked.

Now, if desired, one can regard the pair $(s; d)$ as a pointer to s . And, vice versa, a pointer to s can be changed to a pair $(s; d)$ where d is some description derived from the pointer. Formally, my algorithm can be viewed as providing a truth-value assignment to sentences under descriptions, achieving thereby Skyrms' goal⁵⁴. However, I do not see how the failure of substitutivity arising in the Strong Liar puzzle can be accounted by intensionality, without stretching 'intensionality' so as to render it uninteresting. For, by the same token, one could subsume under it all other context dependent phenomena, including for example indexicality. When each of Jack and Jill says : 'I am tall' then, assuming that only Jack is tall, the two utterances get different truth-values; we could explain that by the intensionality of 'true', namely: the sentence 'I am tall' is true, under the description 'the sentence uttered by Jack', but not true under the description 'the sentence uttered by Jill'. I doubt that we want to adopt this picture.

Intensionality has to do with the way in which referring terms pick their references. The sum of 7 and 2 must be the number 9, but it is only a cosmic accident, or so we think, that 9 is also the number of planets. The intensional effect disappears if the referential connection is made explicit: Given that the number of planets is the sum of 7 and 2, it is necessary that the number of planets is odd. Or, given that George IV knew that the author of Waverly was Scott, it is inconceivable that he wondered whether Scott wrote Waverly. But can we say that, given that the sentence-type on line 2 is the sentence-type on line 1, the sentence on line 1 is true? Certainly not. Nothing in the Liar situation hinges on some unclarity in the referential picture; bringing all the links into the open does not make a difference.

⁵⁴ Indeed, judging from discussions I had with him, Skyrms regards the algorithm as a way of constructing the desired intensional fixed point.

Viewed in retrospect, both the conception of truth: as indexical or as an intensional notion, are attempts to account for phenomena arising out of Liar-like paradoxes by using tools that fall essentially within the framework of traditional logic. Both come short of providing a solution, but for different reasons.

REFERENCES

- (TARK is an abbreviation for: Theoretical Aspects of Reasoning About Knowledge)
- N. Asher 1988 "Reasoning about Belief and Knowledge with Self-Reference and Time" TARK, ed. M. Vardi, Morgan Kaufmann Publishers, pp. 61 - 81.
- N. Asher 1990 "Intentional Paradoxes and an Inductive Theory of Propositional Quantification" TARK, ed. R. Parikh, Morgan Kaufmann Publishers, pp. 11 - 28.
- N. Asher and H. Kamp 1986 "The Knowers Paradox and Representational Theories of Attitudes" TARK pp. 131 - 148.
- J. Austin 1950 "Truth" Proceeding of the Aristotelian Society, Supplementary Volume xxiv, Reprinted in Philosophical Papers J. Austin, Clarendon Press 1961, reissued as an Oxford University Paperback, 1970.
- J. Barwise and J. Etchemendy 1987 *The Liar, an Essay in Truth and Circularity* Oxford University Press.
- T. Burge 1979 "Semantical Paradox" *The Journal of Philosophy* 76 pp. 169 - 198, reprinted in *Recent Essays on Truth and the Liar Paradox* ed. Martin 1984 Oxford University Press.
- J. des Rivieres and H. Levesque 1986 "The Consistency of Syntactical Treatment of Knowledge" TARK pp. 115 - 130.
- H. Gaifman 1983 "Paradoxes of Infinity and Self-Application" *Erkenntnis* vol. 20 pp. 131 - 155.
- H. Gaifman 1988 "Operational Pointer Semantics: Solution to Self-Referential Puzzles I" TARK pp. 43 - 59.
- A. Gupta 1982 "Truth and Paradox" *Journal of Philosophical Logic* 11 pp. 1 - 60, reprinted in *Recent Essays on Truth and the Liar Paradox* ed. Martin 1984 Oxford University Press.
- H. Herzberger 1982 "Notes on Naive Semantics" *Journal of Philosophical Logic* 11 pp. 61 - 102.
- D. Kaplan 1970 "What is Russell's Theory of Descriptions?" in *Physics, Logic, and History*, ed W. Yourgrau and A. D. Beck, New York Plenum Press. Reprinted in *The Logic of Grammar*, ed D. Davidson and G. Harman, Dickenson Publishing Company, California 1975.
- D. Kaplan 1989 "Demonstratives. An Essay on the Semantics, Logic, Metaphysics, and Epistemology of Demonstratives and Other Indexicals" in *Themes From Kaplan* ed J. Almog, J. Perry and H. Wettstein, Oxford University Press.

- Kindt 1979 "Introduction of the truth predicates into first-order languages" in Formal semantics and pragmatics for natural languages eds. Guentner and Schmidt, Reidel.
- R. Koons "Doxastic Paradoxes without Self-Reference" TARK, ed. M. Vardi, Morgan Kaufmann Publishers, pp. 29 - 42.
- M. Kremmer 1986 Logic and Truth Ph.D. Dissertation, University of Pittsburg.
- S. Kripke 1975 "Outline of a Theory of Truth" Journal of Philosophy 72 pp. 690 - 716.
- R. Martin and P. Woodruff 1975 "On Representing 'True-in-L' in L" Philosophia 5 pp. 213 - 217, reprinted in Recent Essays on Truth and the Liar Paradox ed. Martin 1984 Oxford University Press.
- R. Montague 1963 "Syntactical Treatments of Modality, with Corollaries on Reflection Principles and Finite Axiomatizability" Acta Philosophica Fennica 16 pp. 153 - 167.
- L. Morgenstern 1986 "A First Order Theory of Planning, Knowledge, and Action" TARK pp. 99 - 115.
- C. Parsons 1974 "The Liar Paradox" Journal of Philosophical Logic 3 pp. 381 - 412, reprinted in Recent Essays on Truth and the Liar Paradox ed. Martin 1984 Oxford University Press.
- D. Perlis 1985 "Languages with Self-Reference I: Foundations" Artificial Intelligence 25 pp. 301 - 322.
- A. N. Prior 1961 "On a Family of Paradoxes" Notre Dame Journal of Formal Logic, II, 1 pp. 16 - 32.
- B. Russell 1908 "Mathematical Logic as Based on the Theory of Types" American Journal of Mathematics, reprinted in Logic and Knowledge, Essays 1901-1950, R. Marsh ed. Macmillan Company 1956, pp. 59 - 102
- B. Skyrms 1970 "Return of the Liar; Three-Valued Logic and the Nature of Truth" American Philosophical Quarterly 7 pp. 153 - 161
- B. Skyrms 1982 "Intensional Aspects of Semantical self-reference" notes, reprinted in Recent Essays on Truth and the Liar Paradox ed. Martin 1984 Oxford University Press.
- R. Thomason 1986 "Paradoxes and Semantic Representation" TARK pp. 225 - 239
- B. C. van Frassen 1968 "Presumption, Implication, and Self-Reference" The Journal of Philosophy 65 pp. 136 - 152