COMPUTATION OF A SUBCLASS OF INFERENCES: PRESUPPOSITION AND ENTAILMENT*

Aravind K. Joshi+
Ralph M. Weischedel++

Technical Report #74
February 1976

---

* This work was partially supported by NSF Grant SOC 72-0546A01. This is being issued simultaneously as a technical report at the Moore School of Electrical Engineering, University of Pennsylvania.

+ Department of Computer and Information Science, The Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, PA. 19174.

++ Department of Information and Computer Science, University of California at Irvine, Irvine, California 92717.

ABSTRACT

The term "inference" has been used recently in computational linguistics and artificial intelligence to refer to any conjecture or conclusion drawn from a text. Presupposition and entailment are a subclass of inferences that appears to be tied to the structure of language, for they arise from the semantics of particular words and from syntactic constructs of the language. As a subclass of inferences, presupposition and entailment exhibit several properties that do not hold for the general class of inferences; we examine some of those properties here. In particular, we show how to compute the presuppositions and entailments of a sentence while parsing. The computation is by structural means (e.g., uses tree transformations) using an augmented transition network.

0. Introduction

The term "inference" has been used in many ways. In recent artificial intelligence literature dealing with computational linguistics, it has been used to refer to any conjecture given a context (for instance, the context developed from previous text). The conjecture may be true or false. In this sense, "inference" includes more than formally deduced statements. Further, alternatives to formal deduction procedures are sought for computing inferences because formal deductive procedures tend to undergo combinatorial explosion.

A subclass of inferences that we have studied are presupposition and entailment (defined in Section 1). As one would hope in studying a restricted class of a more general phenomenon, this subclass of inferences exhibits several computational and linguistic aspects not exhibited by the general class of inferences.

One aspect is that presupposition and entailment seem to be tied to the definitional (semantic) structure and syntactic structure of language. As a consequence, we demonstrate how they may be computed by structural means (e.g. tree transformations) using an augmented transition network.

A second aspect is that presupposition and entailment exhibit complex interaction of semantics and syntax. They exhibit necessary, but not sufficient, semantics of individual words and of syntactic constructs.

Another aspect relates to the problem of knowing when to stop drawing inferences. There is a natural solution to this problem for the case of presupposition and entailment.

The definitions of presupposition and entailment appear in Section 1, with examples in Sections 2 and 3. A description of the system that

computes the presuppositions and entailments of an input sentence appears in Section 4. (The details of the computation are in Weischedel (1976).) Detailed comparison of this subclass of inferences with the general class of inferences is presented in Section 5. Conclusions are stated in Section 6. An appendix contains sample input-output sessions.

1. Definitions

In this section, we define the inferences we are interested in (presupposition and entailment), and comment on our use of the terms "pragmatics" and "context".

In order to specify the sub-classes of inferences we are studying, we need some preliminary assumptions and definitions. Inferences, in general, must be made given a particular body of pragmatic information and with respect to texts. Since sentences are the simplest cases of texts, we are concentrating on them. Presuppositions and entailments are particularly useful inferences for studying texts having sentences containing embedded sentences, and they may be studied to a limited extent independent of pragmatic information.

1.1 Subformula-derived

We assume that the primary goal of the syntactic component of a natural language system is to translate from natural language sentences to meaning representations selected in an artificial language. Assume further, that the meaning representations selected for English sentences have a syntax which may be approximated by a context-free grammar. By "approximated", we mean that there is a context-free grammar of the semantic representations, though the language given by the grammar may include some strings which have no interpretation. (For instance, the syntax of ALGOL is often approximated by a Backus-Naur form specification.)

Since we have assumed a context-free syntax for the semantic representations, we may speak of the semantic representations as well-formed formulas and as having well-formed subformulas and tree representations.

As long as the assumption of context-free syntax for semantic representations is satisfied, the same algorithms and data structures of our system can be used regardless of choice of semantic primitives or type of semantic representation.

Let S and S' be sentences with meaning representations L and L' respectively. If there is a well-formed subformula P of L and some tree transformation F such that

$$L' = F(P),$$

then we say S' may be subformula-derived from S. The type of tree transformations that are acceptable for F have been formalized and studied extensively in computational linguistics as finite-state tree transformations.

The main point of this work is that the presuppositions and entailments of a sentence may be subformula-derived. We have built a system by which we may specify subformulas P and tree transformations F. The system then automatically generates presuppositions and entailments from an input sentence S.

1.2 Pragmatics and Context

We use context to refer to the situation in which a sentence may occur. Thus, it would include all discourse prior to the sentence under consideration, beliefs of the interpreter, i.e., in short the state of the interpreter. We use pragmatics to describe all phenomena (and computations modelling them) that reflect the effect of context.

1.3 Entailment

A sentence S entails a sentence S' if and only if in every context in which S is true, S' is also true. We may say then that S' is an entailment of S. This definition is used within linguistics as a

more as a test rather than as a rule in a formal system. One discovers empirically whether S' is an entailment of S by trying to construct a context in which S is true, but in which S' is false.

Entailment is not the same as material implication. For instance, let S by "John managed to kiss Mary," which entails sentence S', "John kissed Mary." Givon (1973) argues that even if S' is true, we would not want to say that "John did not manage to kiss Mary." The reason is that "manage" seems to presume an attempt. Hence, if John did not kiss Mary, we cannot conclude that John did not manage to kiss Mary, for he may not have attempted to kiss Mary. Though S entails S', it is not the case that S $\supset$ S', since that would require $\sim S' \supset \sim S$.

We have shown that entailments may be subformula-derived, that is, that they may be computed by structural means. As an example, consider the sentence S below; one could represent its meaning representation as L. S entails S', with meaning representation L'.

S. John forced us to leave.

L. (IN-THE-PAST (force John
(EVENT (IN-THE-PAST (leave we)))))

S'. We left.

L'. (IN-THE-PAST (leave we))

From the meaning representation selected it is easy to see the appropriate subformula and the identity tree transformation which demonstrate that this is a subformula-derived entailment. (This is, of course, a trivial tree transformation. A nontrivial example appears in Section 1.4, for presupposition.) Many examples of entailment are given in Section 2.

Notice that it is questionable whether one understands sentence S or the word "force" if he does not know that S' is true whenever S is. In this sense, entailment is certainly necessary knowledge (though not sufficient) for understanding natural language. We will see this again for presupposition.

1.3 Presupposition

A second, related concept is the notion of presupposition. A sentence S (semantically) presupposes a sentence S' if and only if S entails S' and ~S entails S'.

From the definition one can easily see that all semantic presuppositions S' of S are also entailments of S. However, the converse is not true, as the sentence S and S' above show.

Again, this definition is primarily meant as a linguistic test for empirically determining the presuppositions of a sentence and not as a rule in a formal system.

Note that the truth of a presupposition of a sentence is a necessary condition for the sentence to have a truth value at all. If any of the presuppositions are not true, the sentence is anomolous. For instance, the sentence

"The greatest prime number is 23."

presupposes that there is a greatest prime number. The fact that there is none explains why the sentence is anomolous.

As an example of a subformula derived presupposition consider sentences S1 and S1' below. It is easy to see that whether S1 is true or false, S1' is assumed to be true.

S1: John stopped beating Mary.

L1: (IN-THE-PAST (stop (EVENT (beat John Mary))))

S1': John had been beating Mary.

L1': (IN-THE-PAST (HAVE-EN (BE-ING (beat John Mary))))

L1 and L1' are semantic representations for S1 and S1' respectively. The well- formed subformula in this case is all of L1. The tree transformation from L1 to L1' offers a nontrivial example of a subformula-derived presupposition.

Notice that one might wonder whether sentence S1 and the meaning of "stop" were understood if one did not know that S1' must be true whether John stopped or not. In this sense, presupposition is necessary (but not sufficient) knowledge for understanding natural language.

We have shown that presuppositions (as we have defined them above) may be subformula-derived. Henceforth, we will use "entailment" to mean an entailment which is not also a presupposition.

2. Elementary Examples

This section is divided into two subsections. Section 2.1 deals with presuppositions, section 2.2 with entailments. All example sentences are numbered. An (a) sentence has as presupposition or entailment the corresponding (b) sentence.

2.1 Presupposition

Presuppositions arise from two different structural sources: syntactic constructs (the syntactic or relational structure) and lexical items (semantic structure).

2.1.1 Syntactic constructs

Perhaps the most intriguing cases of presupposition are those that arise from syntactic constructs, for these demonstrate complex interaction between semantics and syntax.

A construction known as the cleft sentence gives rise to presuppositions for the corresponding surface sentences. Consider that if someone says (1) to you, you might respond with (2a).

1. I am sure one of the players won the game for us yesterday, but I do not know who did.

2. a. It is B who won the game.

   b. Someone won the game.

The form of the cleft sentence is the word "it" followed by a tensed form of the word "be", followed by a noun phrase or prepositional phrase, followed by a relative clause.

Note particularly that the presupposition (2b) did not arise from any of the individual words. Rather, the presupposition, which is clearly semantic since it is part of the truth conditions of the sentence, arose

from the syntactic construct. Thus, the syntactic (or relational) structure of the sentence can carry important semantic information.

Cleft sentences illustrate one important use of presuppositions: co-reference. Cleft sentences asert the identity of one individual with another individual referred to previously in the dialogue.

Further, the syntactic constructions associated with definite noun phrases have presuppositions that their referents exist in the shared information between the dialogue participants. By "definite noun phrases", we mean noun phrases which make definite (as opposed to indefinite) reference. Such constructions include proper names, possessives, adjectives, restrictive relative clauses, and nonrestrictive relative clauses. For example, consider the following (a) sentences and their associated presuppositions as (b) sentences.

3. a. John's brother plays for the Phillies.

   b. John has a brother.

4. a. The team that the Phillies play today has won three games in a row.

   b. The Phillies play a team today.

5. a. The Athletics, who won the World Series last year, play today.

   b. The Athletics won the World Series last year.

"Restrictive relative clauses" are relative clauses that are used to determine what the referent is. "Nonrestrictive relative clauses" are not used to determine reference, but rather add additional information as an aside to the main assertion of the sentence. (In written English, they are usually bounded by commas, in spoken English by pauses and change of intonation.)

Note particularly that the restrictive clauses as in (4) presuppose

merely that there is some referent which must have that quality. On the other hand, nonrestrictive relative clauses, such as (5) presuppose that the particular object named also has in addition the quality mentioned in the relative clause. Sentence (5a) might be taken as a paraphrase of "The Athletics play today, and the Athletics won the World Series last year." However, using the syntactic construct of the nonrestrictive relative clause adds the semantic information that not only is (5b) asserted true, but also that (5b) must be presupposed true. Thus, this distinction between the restrictive and nonrestrictive relative clauses demonstrates again that the syntactic construct selected can carry important semantic information.

It is well-known that one role of syntax is to expose (by reducing ambiguity) the relational structure of the meaning of the sentence. The examples of presuppositions of cleft sentences and restrictive and nonrestrictive relative clauses demonstrate that another function of syntax is to convey part of the meaning itself.

For other examples of syntactic constructs that have presuppositions, see Keenan (1971) and Lakoff (1971).

2.1.2 Lexical entry

Presuppositions play an important part in the meaning of many words; these presuppositions may therefore be associated with lexical entries. Only a few classes of semantically-related words have been analyzed so far; analyses of many words with respect to presupposition are reported in Fillmore (1971), Givon (1973), and Kiparsky and Kiparsky (1970). Examples and a summary of such analyses may be found in Keenan (1971) and Weischedel (1975).

All of the following examples of presuppositions arise from the lexical entries for particular words. Again, the (b) sentence in each example is presupposed by the (a) sentence.

The (very large) class of factive predicates provide clear examples of presuppositions, (see Kiparsky and Kiparsky (1970)). Factive predicates may be loosely defined as verbs which take embedded sentences as subject or object, and the embedded sentences can usually be replaced by paraphrasing them with "the fact that S."

6. a. I regret that the Phillies have made no trades.

   b. The Phillies have made no trades.

Example (6) above demonstrates that another function of presupposition in language is informing that the presupposition should be considered true. We can easily imagine (6a) being spoken at the beginning of a press conference to inform the news agency of the truth of (6b).

It should be pointed out that presuppositions arising from lexical items have been studied primarily for verbs and verb-like elements such as adverbs. For instance, presuppositions have not, in general, been associated with common nouns.

Fillmore (1971) has found presupposition to be a very useful concept in the semantics of a class of verbs that he labels the verbs of judging. For instance, (7a) presupposes (7b) and asserts (8b). On the other hand, (8a) presupposes (8b) and asserts (8b). Thus, "criticize" and "accuse" are in some sense the dual of each other.

7. a. The manager criticized B for playing poorly.

   b. B is responsible for his playing poorly.

8. a. The manager accused B of playing poorly.

   b. B's playing poorly is bad.

Keenan (1971) pointsout that some words, such as "return", "also", "too", "again", "other", and "another", carry the meaning of something being repeated. These words have presuppositions that the item occurred at least once before.

9. a. B did not play again today.

   b. B did not play at least once before.

Note that these words include various syntactic categories. "Also", "too", "again", are adverbial elements (adjuncts). "Other", and "another" have aspects of adjectives and of quantifiers. Again we see that the phenomenon of presupposition is a crucial part of the meaning of many diverse classes of words.

Given these introductory examples, let us turn our attention to examples of entailment.

2.2 Entailment

Entailments appear to have been studied less than presupposition. All of the examples identified as entailment thus far seem to be related to lexical entries of particular words. Two comprehensive papers that analyze words having entailments are Karttunen (1970) and Givon (1973).

2.2.1 Classification of words having entailments

At least five distinct semantic classes of words having entailments have been identified by Karttunen (1970). In the following examples, the (b) sentence is entailed by the (a) sentence.

Predicates such as "be in a position", "have the opportunity", and "be

able", are called "only-if" verbs because the embedded sentence is entailed only if the predicate is in the negative. For instance, (10a) entails (10b), but (11) has no entailment.

10. a. The Phillies were not in a position to win the pennant.

b. The Phillies did not win the pennant.

11. a. The Phillies were in a position to win the pennant.

Verbs such as "force", "cause", and "compel" are "if" verbs, for the embedded sentence is entailed if they are in the positive.

12. a. Johnny Bench forced the game to go into extra innings.

b. The game went into extra innings.

13. Johnny Bench did not force the game to go into extra innings.

Note that (12a) entails (12b), but (13) has no such entailment.

A "negative-if" verb entails the negative of the embedded sentence when the verb is positive. "Prevent" and "restrain from" are such verbs.

14. a. His superb catch prevented the runner from scoring.

b. The runner did not score.

15. His superb catch did not prevent the runner from scoring.

Thus, (14a) entails (14b), but (15) has no such entailment.

The three classes of verbs above may be called one-way implicative verbs; there are also two-way implicative verbs. Such verbs have an entailment whether positive or negative.

If the entailment is positive, we may call these "positive two-way implicative" verbs. Examples (16) and (17) illustrate "manage" as such a verb.

16. a. B managed to win.

b. B won.

17. a. B did not manage to win.

    b. B did not win.

There are also "negative two-way implicative" verbs. Consider (18) and (19).

18. a. B failed to make the catch.

    b. B did not make the catch.

19. a. B did not fail to make the catch.

    b. B made the catch.

For this class of verbs, the entailed proposition is positive if and only if the implicative verb is negated.

The five classes of words having entailments, then, are: <u>if</u>, <u>only if</u>, <u>negative if</u>, <u>positive two-way implicative</u>, and <u>negative two-way implicative</u>. All of the words cited in the literature as having entailments are predicates. In the examples here, many were verbs; some were adjectives such as "able". However, some are nouns such as "proof"; example (20) demonstrates this.

20. a. The fact that he came is proof that he cares.

    b. He cares.

We now turn our attention to various factors that must be accounted for in computing presuppositions and entailments of compound sentences.

3. Complex Examples: Embedded Entailments and Presuppositions

In this section, the following question is considered: Suppose that a sentence S has a set of entailments and a set of presuppositions. Suppose further, that S is embedded in another sentence S'. Are the entailments and presuppositions of S also entailments and presuppositions of S' as a whole?

This has been referred to as the projection problem for entailments and presuppositions. A solution to the problem involves rules for combining semantic entities of embedded (projected) sentences in order to compute the semantic entities of the whole sentence.

A solution to the projection problem evolved in Karttunen (1973, 1974), Karttunen and Peters (197), Joshi and Weischedel (1974), Smaby (1975) and Weischedel (1975). The results are briefly reported here. A summary of the solutions may be found in Weischedel (1975).

Karttunen (1973, 1974) divided all predicates into four classes: the speech acts, predicates of propositional attitude, connectives, and all other predicates. The classes were defined according to the effect of the predicate on presuppositions of embedded sentences. We found that the same classification was appropriate for entailments, and extended the solution to include entailments, as well as presuppositions.

3.1 Presupposition

As an example sentence, consider (1), which presupposes (2).

1. Jack regretted that John left.
2. John left.

In the following sections, we will consider the effect on presupposition (2) of embedding (1) under various predicates taking embedded sentences.

3.1.1 Holes

Many predicates taking embedded sentences could be called holes because they let presuppositions of embedded sentences through to become presuppositions of the compound sentence. "Aware" is such a predicate; (3) presupposes (2).

3. Mary is aware that Jack regretted that John left.

All predicates taking embedded sentences, except for the verbs of saying, the predicates of propositional attitude, and the connectives appear to be holes.

3.1.2 Speech acts

The verbs of saying, or "speech act" verbs, permit the presuppositions to rise to be presuppositions of the compound sentence, but those presuppositions are embedded in the world of the claims of the actor performing the speech act. Smaby (1975) first pointed out this important fact.

For instance, (4) presupposes (5), not (2).

4. Mary asked whether Jack regretted that John left.

5. Mary claimed John left.

3.1.3 Predicates of propositional attitude

Analysis of predicates of propositional attitude is very similar to that of speech acts. Some predicates of propositional attitude are "believe", "think", and "hope". In general, presuppositions of sentences embedded under such a predicate must be embedded under the predicate "believe" to reflect that they are presuppositions in the world of the actor's beliefs. This was first pointed out by Karttunen (1974).

For example, (6) presupposes (7), not (2).

6. Mary thinks Jack regretted that John left.

7. Mary believes John left.

3.1.4 Connectives

The effect of connectives is rather complex, as (8) and (9) demonstrate. Sentence (8) presupposes (2), but (9) clearly does not.

8. If Jack was there, then Jack regretted that John left.

9. If John left, then Jack regretted that John left.

Let A and B be the antecedent and consequent respectively of the compound sentence "if A then B".

The examples of (8) and (9) are complex, for they seem to demonstrate that the context set up by the antecedent A must be part of the computation. This would in general require complex theorem provers in order to determine whether the presuppositions of B are implied by A, and therefore are not presuppositions of the compound sentence. However, Peters suggested (a footnote in Karttunen (1974)) that the presuppositions of "if A then B," (where material implication is the interpretation of "if - then"), arising from the presuppositions of B are of the form "if A then C", where C is a presupposition of B. Further, all presuppositions of A are presuppositions of "if A then B." This suggestion eliminates the need for theorem proving and offers instead a simple computation similar to that for the verbs of saying and the verbs of propositional attitude.

For the examples given then, (8) presupposes (10), and (9) presupposes (11) which is a tautology.

10. If Jack was there, then John left.

11. If John left, then John left.

One may easily verify that (8) presupposes (10) by a truth table computation.

Karttunen (1973) argues that the solution of "A and B" reduces to the solution of "if A then B" and that the solution to "A or B" reduces to the solution of "if not(A) then B".

This completes the description of the four classes of embedding predicates and their effect on embedded presuppositions. However, there is another phenomenon, that of embedded entailments becoming presuppositions of compound sentences.

3.1.5 Entailments promoted to presuppositions

Clearly, any entailment of a presupposition must be a presupposition also; this is evident from the definitions. For instance, (12) presupposes (13). Since (13) entails (14), (14) must also be a presupposition of (12).

12. Jack regretted that John's children forced Mary to leave.

13. John's children forced Mary to leave.

14. Mary left.

The five cases discussed above outline a solution to the projection problem for presuppositions.

3.2 Entailments

In the examples, we will embed (15) under various predicates, to see how the entailment (16) of (15) is affected.

15. Fred prevented Mary from leaving.

16. Mary did not leave.

3.2.1 Chain of entailments

Corresponding to the class of holes for presuppositions, two cases arise for entailments. One case was covered in 3.1.5; entailments of an embedded sentence which is a presupposition are presuppositions of the compound sentence.

A second, disjoint case involves setting up a chain of entailments. For instance, (17) entails (15) which entails (16).

17. John forced Fred to prevent Mary from leaving.

This is truly a chain of entailments, since breaking a link in the chain causes embedded entailments to be blocked. For instance, the presence (absence) of negation is crucial; if (17) were negative, it would not entail (15) nor (16), through (17) did.

Thus, for the case involving a chain of entailments, the entailments of an embedded sentence are entailed by the compound sentence only if such a chain of entailments can be set up.

3.2.2 Speech Acts

Smaby has pointed out that there are at least two subclasses of speech act verbs according to behavior of embedded entailments. Further, the syntactic shape of the embedded sentence affects entailments.

For instance, if the syntactic shape of an embedded sentence S is "whether S or not", "for NP to VP", or "if S", all embedded entailments are blocked. For instance, (18) entails nothing about Mary's leaving.

18. John asked whether or not Fred prevented Mary from leaving.

However, a "wh-some" embedded sentence (beginning with "who", "what", "when", "which", etc.) have all entailments of the embedded sentence promoted to presuppositions, since the embedded sentence is presupposed. For instance, (19) presupposes (20), and therefore presupposes that "Mary did not leave".

19. John asked who prevented Mary from leaving.

20. Someone prevented Mary from leaving.

For embedded sentences of the form "that S", we notice two subclasses of speech acts. Verbs such as "say", "declare", and "affirm" are like "if predicates", for embedded entailments are not blocked if the verb is not in the negative. However, in the positive, the embedded entailments become entailments of the compound sentence, but under the speaker's claims. For instance, (21) entails (22).

21. John said that Fred prevented Mary from leaving.

22. John claimed that Mary did not leave.

A second subclass of verbs includes "deny". They are analogous to "negative if verbs". When "deny" is in the negative, embedded entailments are blocked. However, when "deny" is positive the entailments of the negative form of the embedded sentence are entailed by the compound sentence, but under the speaker's claims. For instance, (24) is entailed by (23).

23. John denied that Mary was able to leave.

24. John claimed that Mary did not leave.

3.2.3 Predicates of propositional attitude

Smaby (1975) analyses these predicates in the same way as the speech acts. "Believe", "think", and "suspect" are examples of a subclass analogous to "if predicates" or to "say", "declare", and "affirm". "Doubt" is an example of a second subclass analogous to "negative two-way implicative predicates" such as "fail".

Though the subclasses for predicates of propositional attitude are analogous to those of the speech acts, the embedded entailments of propositional attitude predicates become entailments of the compound sentence under the actor's beliefs, rather than under the speaker's claims as in the

speech act case. For instance, (25) entails (26).

25. John thought that Fred prevented Mary from leaving.

26. John believed that Mary did not leave.

3.2.4 Connectives

For "if A then B", the entailments are of the form "if A then C", where C is an entailment of B. For "A and B", the entailments are the union of the entailments of A and of the entailments of B, since both A and B are entailed by "A and B". For "A or B", there do not seem to be any useful entailments.

This concludes the analysis of the projection problem for presuppositions and entailments.

4. Outline of the solutions in the system

The purpose of this section is to give an overall view of the system and an outline of the methods used to compute presupposition and entailment. Section 4.1 presents a block diagram of the system; 4.2 briefly outlines the computation for the various examples of sections 2 and 3; section 4.3 attempts to state some of the limitations of the system, including the memory and time requirements.

4.1 Block diagram

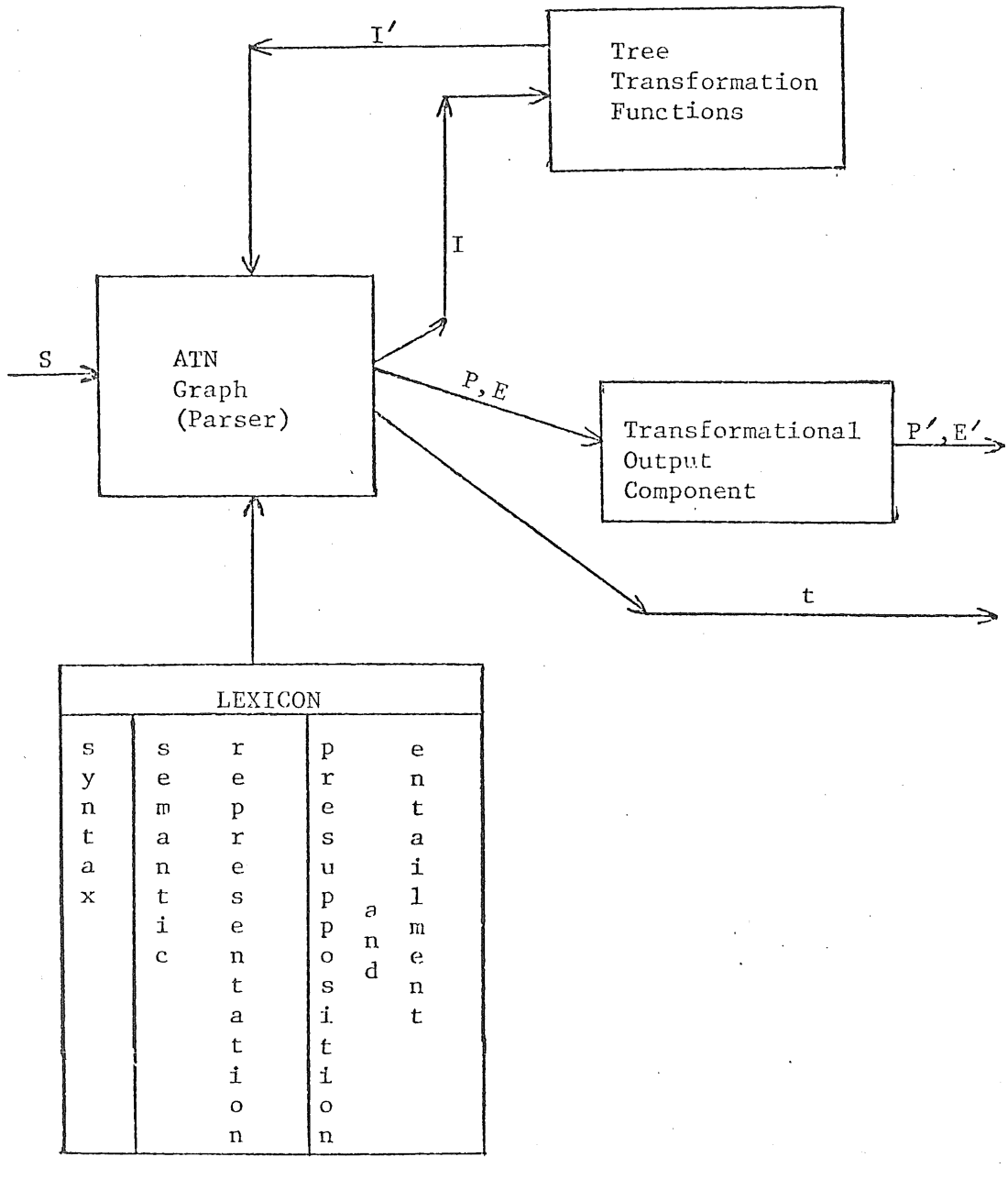
A block diagram of the system appears in Figure 4.1. All arrows represent data flow. A sentence S in English is input to the system. The parser is written as an augmented transition network graph (ATN). (Woods (1970) specifies the ATN as a formal model and as a programming language.) While parsing, the ATN refers to the lexicon for specific information for each word of the sentence S. Lexical information is of three types: syntactic information, information for generating the semantic representation or translation, and information for making lexical inferences -- presuppositions and entailments. The organization of the lexicon for computing lexical inferences (presuppositions and entailments) is a novel aspect of the system.

From the definition of presuppositions and entailments, it is clear that the system needs a set of functions for manipulating or transforming trees. These appear as a separate block in Figure 4.1. The parser calls them while parsing; this is represented in the diagram as input I and values I' of functions. These functions are written in LISP.

Figure 4.1

System Structure

(All arrows represent data flow)



Using the lexical information, the relational or syntactic structure of the sentence, and the tree transformation functions, the parser generates the semantic representation (translation) t of the sentence and a set of presuppositions P and entailments E of the sentence.

Since each presupposition P and entailment E is in the logical notation of the semantic representations of sentences, a small transformational output component has been included to give the presuppositions and entailments as output in English. These appear as P' and E'; in Figure 4.1. The transformational output component is also written in LISP. This output component is very small in scope and is not a major component of the work reported here.

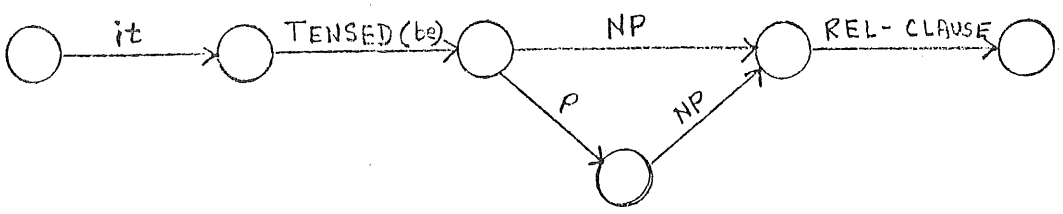
4.2 Outline of solution

A sketch of the computation of presupposition and entailment is presented here; details of computation are presented in Weischedel (1976).

There are four fundamental phenomena exhibited in sections 2 and 3: presuppositions from syntactic constructs, presuppositions from particular words (lexical entries), entailments from lexical entries, and the projection phenomena.

In order to compute presuppositions from syntactic constructs, two principles are important: detecting the syntactic construction and dealing with ambiguity. Syntactic constructs are syntactically marked in the sentence. Thus, the parser may be constructed such that there is a parse generated when those syntactic markings are present. In the ATN, one may construct the graphs representing the grammar such that there is a particular path which is traversed if and only if the syntactic construct is present. Then, we may associate with that particular path the tree transformation

yielding the presupposition of that syntactic construct. For instance, cleft sentences are syntactically marked as the word "it", followed by a tensed form of "be", followed by either a noun phrase or a prepositional phrase, followed by a relative clause. The path(s) in the graph might be as below.



Associated with this path would be a trivial tree transformation which returns the semantic representation of the relative clause as a presupposition.

The second principle deals with ambiguity. Even though we have structured the graphs in the way above, the same surface form may arise from two different syntactic constructs, one having a presupposition and the other not. In such a case, our system (and in fact any parser) should be able to give semantic representations for both parses; with one parse our system yields a presupposition, with the other parse our system would not have the presupposition. It is the role of general semantic and pragmatic components to distinguish which semantic representation is intended in the context. In fact, the difference in the presuppositions with the differing parses is one criterion which general semantic and pragmatic components could use to resolve the ambiguity.

For generating presuppositions of words (lexical entries), the chief problems are how to encode the tree transformation in the lexicon (dictionary)

and when to apply it during parsing. In general a tree transformation would have a left hand side which is the pattern to be matched if the transformation is to apply and a right hand side giving the transformed structure.

The reason we can encode the left hand side in the grammar is simple. All of the examples in the literature dealing with presuppositions from lexical entries have in common the fact that the existence of the presupposition depends only upon the syntactic environment of the word and the word itself. Hence, we can structure the graph of the grammar in a way that the paths correspond to the necessary syntactic environments. Upon encountering a word of the appropriate syntactic category in such a syntactic environment, the system looks in the lexicon under that word for the (possibly empty) set of right hand sides of tree transformations.

The way of writing the right hand sides assumes that the parser in traversing a path undoes the syntactic construct encoded in that path, and assigns the components of the semantic representation according to their logical role in the sentence rather than their syntactic role. (This is not a new idea, but rather has been used in several systems pre-dating ours. As an example, the semantic representation of "Mary" in the following three sentences would be assigned to the same register while parsing, "John gave Mary a ball", "Mary was given a ball by John", and "A ball was given to Mary by John".) Thus, we can assume a convention for naming registers and assigning components of the semantic representation to them, independent of the syntactic environment. To encode the right hand side of the tree transformation, we use a list whose first element is the tree structure with constants as literal atoms and positions of variables as plus signs. The

remaining elements of the list specify the registers to fill in the variable positions.

This, then, is how we integrate the tree transformations for presuppositions into the parse. The lexical examples for entailment also must employ tree transformations but are complicated by the five different classes of predicates yielding entailments and their dependence on whether the sentence is negated or not. A further complication was illustrated in section 3, for a chain of entailments must be set up.

For entailments, we encode the left hand side and right hand side in the same way as the lexical examples of presuppositions. However, for entailments, for each right hand side we also encode three other pieces of information. They are the pre-condition of whether negation must be present (or absent), whether the entailed proposition is negative or not, and whether the entailed propositional corresponds to the left sub-tree or right sub-tree. At each sentential level, we verify that the left hand side of the tree transformation is present. If it is, we make the transformation indicated in the lexicon and save the resulting proposition along with the other three pieces of information mentioned above associated with it. We save this in a binary tree, one level of tree per sentential level. It is a binary tree since all predicates taking embedded sentences seem to permit only one or two of its arguments to be embedded sentences.

Upon hitting the period (or question mark), all of the negation information is present so that we may simply traverse the tree from the root, doing a comparison at each level to verify that the conditions for negation being present (absent) are met. This completes an outline or computation of entailments.

Next we outline a solution to the projection problem. The structural solution to the projection problem traced in section 3 has simple computational requirements. We have structured the ATN graphs such that recursion occurs for each embedded sentence. At each sentential level, the graph returns as a value a list of at least four elements: the semantic representation of the sentence at this level, a list of presuppositions of this sentence and any embedded in it, a tree as described above for computing entailment at this and lower levels, as well as a list of semantic representations of noun phrases encountered at this or lower levels.

Just before popping to a higher sentential level a projection function is applied, which is merely a CASE statement for the four cases described in section 3. For holes, nothing is changed. For speech act predicates and of propositional attitude, the presuppositions of embedded sentences and propositions in the tree for entailments are embedded under a special semantic primitive (CLAIM for speech acts, BELIEVE for verbs of propositional attitude). Embedding under these primitives places the presuppositions and entailments in the world of the actor's claims or beliefs.

For connectives, the computation is just as described in section 3. Again, an embedding is involved, this time under a semantic primitive IF-THEN to place the propositions in the world of the context created by the left sentence of the connective.

For details and solution to specific problems cited in section 2, consult Weischedel (1976).

4.3 What the System Does Not Do

The limitations of the system are of two kinds: those that could be handled within the framework of the system but are not because of limitations of

man-hours, and those that could not be handled within the present framework.

4.3.1 Limitations that could be removed

The system is currently limited in four ways, each of which could be removed, given time. One set of restrictions results from the fact that our program represents only a small part of a complete natural langauge processing system. Only the syntactic component is included (though these inferences, which are semantic, are computed while parsing). As a consequence, no ambiguity is resolved except that which is syntactically resolvable.

Second, though a transformational output component is included to facilitate reading the output, it has a very limited range of constructions. The principles used in designing the component are sound though.

A third aspect is computation time. Since our main interest was a new type of computation for a syntactic component, we have not stressed efficiency in time nor storage; rather, we have concentrated on writing the system fairly rapidly. Considering the number of conceptually simple, efficiency measures that we sacrificed for speed in implementing the system, we are quite pleased that the average CPU time to compute the presuppositions and entailments of a sentence is twenty seconds on the DEC PDP-10. The memory requirements were 90K words including the LISP interpreter and interpreter for augments transition networks. For further details and the simple economies that we have not used, see Weischedel (1976).

As a fourth class, we mention the syntactic constructions allowable as input to the system. We have not allowed several complex syntactic problems which are essentially independent of the problems of computing presuppositions and entailments, such as conjunction reduction, complex anaphoric reference,

or propositional phrases on noun phrases. The number of English quantifiers in the system is small. Also the dictionary is of very modest size (approximately 120 stem words). However, our lexicon is patterned after the lexicon of the linguistic string parser, which includes 10,000 words. Therefore, we have avoided the pitfall of grammatical ad hocness. (The linguistic string parser is described in Sager (1973).)

We have not included modal tenses or subjunctive mood. This is because the effect of modals and the subjunctive mood on presupposition and entailment has not been fully worked out yet. A limited solution for modals and subjunctives has been worked out for a micro-world of tic-tac-toe in Joshi and Weischedel (1975).

For a more detailed view of the syntactic and lexical restrictions in the system, as well as their extent, refer to Weischedel (1975).

4.3.2 Limitations difficult to remove

We have dealt with specific time elements for presupposition and entailment in a very limited way. Time has been explicitly dealt with only for the aspectual verbs; however, time is implicitly handled in detail for all presuppositions and entailments through tense (see Weischedel 1976)). We have not included time otherwise, because we feel that the same solution presented for assigning tenses to presupposition and entailment may be adapted for explicit time elements.

A more serious difficulty would arise if presuppositions or entailments were discovered which depended on different information than any considered up until time time. For instance, the occurrence of presuppositions thus far discovered has depended only on syntactic constructions, lexical entries, and the four classes of embedding predicates (holes, connectives,

speech acts, and verbs of propositional attitude). The existence of entailments thus far encountered has depended only on negation, syntactic constructions, lexical entries, and the four classes of embedding predicates.

It is conceivable that presuppositions and entailments will be discovered which depend on other entities; for instance, presuppositions or entailments of some predicate might be found to depend on the tense of the predicate. If such examples are found, different means of writing lexical entries would have to be devised in order to encode these dependencies.

5. Role of presupposition and entailment

In section 5.1, the role of presupposition and entailments as inferences is pinpointed. In section 5.2, the use of semantic primitives is considered.

5.1 Inferring

The term "inference" has been used recently to refer to any conjecture made, given a text in some natural language. Charniak (1973, 1972), Schank (1973), Schank and Rieger (1973), Schank, et. al. (1975), and Wilks (1975) concentrate on such inferences. All of the projects seek some computational means as an alternative to formal deductive procedures because those tend to combinatorial explosion.

That presupposition and entailment are inferences is obvious. However, the requirement in their definition that they be independent of the situation (all context not represented structurally) is strong. For instance, from sentence S below, one might feel that S' should be entailed; yet, it is not.

S: John saw Jim in the hall, and Mary saw Jim in his office.

S': John and Mary saw Jim in different places.

By appropriately chosen previous texts, S' need not be true whenever S is. For example, the previous text might indicate that Jim's office is in the hall. In general, common nouns do not seem to offer many examples of presupposition and entailment. From the example, it is clear that presupposition and entailment are strictly a subclass of inferences.

Presupposition and entailment are a subclass of inferences distinguished in several ways; First, presupposition and entailment are reliable inferences, rather than being merely conjectures. Presuppositions are true whether the sentence is true or false. Entailments must be true if the sentence is true.

Second, presupposition and entailment are inferences that seem to be tied to the structure of language, for they arise from syntactic structure and from definitional structure of individual words. The fact that they are tied to the structure of language enables them to be computed by structural means (i.e., tree transformations), a computational means not appropriate for all inferences.

Furthermore, since presupposition and entailment are tied to the syntactic and definitional structure of language, these inferences need to be made. For instance, upon encountering "John was not able to leave", one really does want to infer the entailment that "John did not leave". Whether or not it is wise to compute conjectural inferences, on the other hand, does not have a simple answer, by virtue of their conjectural nature.

A fourth distinction of presupposition and entailment is in the problem of knowing when to stop inferring. Inferences themselves can be used to make other inferences, which can be used to make still more inferences, etc. When to stop the inferences is an open question. Presupposition and entailment, as a subclass of inferences, do not exhibit such a chain reaction of inferences. The reason is that presupposition and entailment arise from either the individual words or the particular syntactic constructs of the sentence; presuppositions and entailments do not themselves give rise to more inferences.

We may summarize these distinguishing aspects of presupposition and entailment by the fact that presupposition and entailment are important semantically for understanding words and syntactic constructs. This does not deny the importance of other inferences; conjectural inferences are necessary to represent pragmatic aspects of natural language.

The role of presupposition and entailment in a complete natural language processing system, then, is that they are a subclass of the inferences which the system must compute. Inferences in general are made from an input sentence in conjunction with the system's model of the context of the situation. Presupposition and entailment are a subclass of inferences associated with the semantic structure of particular words and with the syntactic structure of the sentence. Thus, as we have shown, they may be computed while parsing using lexical information and grammatical information. The system's model of the context of the situation is not needed to compute the presuppositions and entailments for any reading or interpretation of a sentence; of course, to ascertain which reading or interpretation of a sentence is intended in a given context, the system's model of the context is essential.

5.2 Semantic primitives

Semantic primitives have been investigated as the element with which to associate inferences. (See Schank (1973), Schank, et.al. (1975), Yamanashi (1972)). This has the important advantage of capturing shared inferences of many similar words by a semantic primitive, rather than repeating the semantic information for those shared inferences for each word. Inferences would be made in the semantic component.

The assumptions of our computation do not preclude the use of primitives in semantic representations. On the contrary, the particular semantic representations our system uses do include primitives. However, we have not associated the computation of presupposition and entailment with semantic primitives.

The reason is that presuppositions arise from syntactic constructs, as well as from the semantics of particular words. Further, syntactic structure can interact with the entailments of words, as in the following example. Because S' is presupposed by S, S" becomes a presupposition of S, not merely an entailment.

S Who prevented John from leaving?

S' Someone prevented John from leaving.

S" John did not leave.

To compute such effects in the semantic component, sufficient syntactic structure of the surface sentence would have to be available to the semantic component. Whether that is possible or whether that would be wise is not clear. For that reason, we have not used semantic primitives to compute presupposition and entailment.

6. Conclusion

The main goal of this work is its demonstration of a method for writing the lexicon and parser for the computation of presupposition and entailment, and its exhibition of the procedures and data structures necessary to do this.

Presupposition and entailment comprise a special class of inferences, distinguished in three ways. First, they both may be computed structurally (by tree transformations), independent of context not inherent in the structure. Second, although inferences in general are conjectural, presupposition and entailment may be reliably asserted; entailments are true if the sentence entailing them is true; presuppositions are true whether the sentence presupposing them is true or false. Third, since presupposition and entailment are tied to the definitional and syntactic structure of the language, they do not spawn themselves nor lead to a chain reaction explosion, as other inferences may.

We suggest two areas of future research. One is to derive a means of accounting for presuppositions arising from syntactic constructs, in a way consistent with using semantic primitives to account for lexical examples of presupposition and entailment.

A second area is suggested by the interaction of syntax and semantics evident in presuppositions arising from syntactic constructs. A study of phenomena that cut across the boundaries of syntax, semantics, and pragmatics and a computational model incorporating them could prove very fruitful to our understanding of natural languages.

Included here is the output for several exemplary sentences. The semantic representations are a function and argument notation developed by Harris (1970) and modified by Keenan (1972). As in logic, variables are

bound outside of the formula in which they are used. Any semantic primitives may be used, as long as they employ the function - argument syntax. Details about the semantic represention may be found in Weischedel (1975).

APPENDIX

We now describe the format of the output. The first item is the sentence typed in. Note that /, means comma and /. means period, because of LISP delimiters.

The semantic representation of the input sentence itself is printed next, under the heading "SEMANTIC REPRESENTATION".

Presuppositions not related to the existence of referents of noun phrases are printed under the label "NON-NP PRESUPPOSITIONS". Presuppositions about existence of referents of noun phrases are printed under the label "NP-RELATED PRESUPPOSITIONS". The set of entailments follows the label "ENTAILMENTS". If for any of these sets, the set is empty, then only the label is printed. For the two sets of presuppositions and the set of entailments, the semantic representation of the set of entailments in Keenan's notation is printed first, then the English paraphrase generated by the output component.

In some cases the tense of a presupposition is not known. In such instances, the output component prints the stem verb followed by the symbol "-UNTENSED-".

Examples of presuppositions from syntactic constructs appear in examples 1 and 2; the cleft construction gives a presupposition in 1; the definite noun phrase in 2 gives a presupposition. Presuppositions from lexical entries appear in 3 and 4. "Only" in 3 has a presupposition; "fail" in 4 also has a presupposition. Comparing 4 and 5 demonstrates the computation of a chain of entailments.

Several examples of the projection problem have been included. Examples of predicates which are holes appear in 4 and 5. The effect of speech acts appears in 6. The effect of "if ... then" (interpreted as material implication) is evident in 7 and 8.

The terminal sessions follow.

IT IS DR SMITH WHO TEACHES CIS591 /.

SEMANTIC REPRESENTATION

((CIS591 /, X0006) ((DR$SMITH /, X0005) (ASSERT I (IN-THE-PRESENT (BE IT (IN-THE-PRESENT (TEACH X0005 NIL X0006)))))))

NON-NP PRESUPPOSITIONS

((CIS591 /, X0006) ((E INDIVIDUAL /, X0005) (IN-THE-PRESENT (TEACH X0005 NIL X0006))))

SOME INDIVIDUAL TEACHES CIS591 .

NP-RELATED PRESUPPOSITIONS

((DR$SMITH /, X0005) (*UNTENSED (IN-THE-SHARED-INFO X0005)))

DR SMITH EXIST -UNTENSED- IN THE SHARED INFORMATION .

((CIS591 /, X0006) (*UNTENSED (IN-THE-SHARED-INFO X0006)))

CIS591 EXIST -UNTENSED- IN THE SHARED INFORMATION .

((DR$SMITH /, X0005) (*UNTENSED (HUMAN X0005)))

DR SMITH BE -UNTENSED- HUMAN .

ENTAILMENTS

Example 1

THE PROFESSOR THAT I ADMIRE BEGAN TO ASSIGN THE PROJECTS /.

SEMANTIC REPRESENTATION

((((COLLECTIVE PROJECT /, X0010) (NUMBER X0010 TWO-OR-MORE)) /, X0017
) ((((THE PROFESSOR /, X0008) (IN-THE-PRESENT (ADMIRE I X0008))) /, X
0009) (ASSERT I (IN-THE-PAST (START (EVENT (ASSIGN X0009 NIL X0017))
NIL)))))

NON-NP PRESUPPOSITIONS

((((COLLECTIVE PROJECT /, X0010) (NUMBER X0010 TWO-OR-MORE)) /, X0017
) ((((THE PROFESSOR /, X0008) (IN-THE-PRESENT (ADMIRE I X0008))) /, X
0009) ((((E TIME /, X0018) (IMMEDIATELY-BEFORE X0018 NIL)) /, X0019)
(AT-TIME (NOT (IN-THE-PAST (HAVE-EN (BE-ING (ASSIGN X0009 NIL X0017))
))) X0019))))

IT IS NOT THE CASE THAT THE PROFESSOR THAT I ADMIRE HAD BEEN
ASSIGNING THE PROJECTS .

NP-RELATED PRESUPPOSITIONS

((((E PROFESSOR /, X0008) (IN-THE-PRESENT (ADMIRE I X0008))) /, X0009
) (*UNTENSED (IN-THE-SHARED-INFO X0009)))

SOME PROFESSOR THAT I ADMIRE EXIST -UNTENSED- IN THE SHARED
INFORMATION .

((((E PROJECT /, X0010) (NUMBER X0010 TWO-OR-MORE)) /, X0017) (*UNTEN
SED (IN-THE-SHARED-INFO X0017)))

SOME PROJECTS EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

((((COLLECTIVE PROJECT /, X0010) (NUMBER X0010 TWO-OR-MORE)) /, X0017
) ((((THE PROFESSOR /, X0008) (IN-THE-PRESENT (ADMIRE I X0008))) /, X
0009) ((((E TIME /, X0020) (IMMEDIATELY-AFTER X0020 NIL)) /, X0021) (
AT-TIME (IN-THE-PAST (BE-ING (ASSIGN X0009 NIL X0017))) X0021))))

THE PROFESSOR THAT I ADMIRE WAS ASSIGNING THE PROJECTS .

Example 2

ONLY JOHN WILL LEAVE /.

SEMANTIC REPRESENTATION

((((A INDIVIDUAL /, X0063) ((JOHN /, X0061) (NEQ X0063 X0061))) /, X0062) (ASSERT I (NOT (IN-THE-FUTURE (LEAVE X0062)))))

NON-NP PRESUPPOSITIONS

((JOHN /, X0061) (IN-THE-FUTURE (LEAVE X0061)))

JOHN WILL LEAVE .

NP-RELATED PRESUPPOSITIONS

((JOHN /, X0061) (*UNTENSED (IN-THE-SHARED-INFO X0061)))

JOHN EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

Example 3

THAT DR SMITH FAILED TO CHALLENGE JOHN IS TRUE /.

SEMANTIC REPRESENTATION

```
((JOHN /, X0045) ((DR$SMITH /, X0044) (ASSERT I (IN-THE-PRESENT (TRUE
 (IN-THE-PAST (NOT (COME-ABOUT (EVENT (CHALLENGE X0044 X0045)))))))))
)
```

NON-NP PRESUPPOSITIONS

```
((JOHN /, X0045) ((DR$SMITH /, X0044) (IN-THE-PAST (ATTEMPT (EVENT (C
HALLENGE X0044 X0045))))))
```

DR SMITH ATTEMPTED TO CHALLENGE JOHN .

NP-RELATED PRESUPPOSITIONS

```
((DR$SMITH /, X0044) (*UNTENSED (IN-THE-SHARED-INFO X0044)))
```

DR SMITH EXIST -UNTENSED- IN THE SHARED INFORMATION .

```
((JOHN /, X0045) (*UNTENSED (IN-THE-SHARED-INFO X0045)))
```

JOHN EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

```
((JOHN /, X0045) ((DR$SMITH /, X0044) (IN-THE-PAST (NOT (COME-ABOUT (
EVENT (CHALLENGE X0044 X0045)))))))
```

DR SMITH FAILED TO CHALLENGE JOHN .

```
((JOHN /, X0045) ((DR$SMITH /, X0044) (NOT (IN-THE-PAST (CHALLENGE X0
044 X0045)))))
```

IT IS NOT THE CASE THAT DR SMITH CHALLENGED JOHN .

Example 4

THAT DR SMITH FAILED TO CHALLENGE JOHN IS FALSE /.

SEMANTIC REPRESENTATION

((JOHN /, X0048) ((DR$SMITH /, X0047) (ASSERT I (IN-THE-PRESENT (NOT (TRUE (IN-THE-PAST (NOT (COME-ABOUT (EVENT (CHALLENGE X0047 X0048))))))))))

NON-NP PRESUPPOSITIONS

((JOHN /, X0048) ((DR$SMITH /, X0047) (IN-THE-PAST (ATTEMPT (EVENT (CHALLENGE X0047 X0048))))))

DR SMITH ATTEMPTED TO CHALLENGE JOHN .

NP-RELATED PRESUPPOSITIONS

((DR$SMITH /, X0047) (*UNTENSED (IN-THE-SHARED-INFO X0047)))

DR SMITH EXIST -UNTENSED- IN THE SHARED INFORMATION .

((JOHN /, X0048) (*UNTENSED (IN-THE-SHARED-INFO X0048)))

JOHN EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

((JOHN /, X0048) ((DR$SMITH /, X0047) (NOT (IN-THE-PAST (NOT (COME-ABOUT (EVENT (CHALLENGE X0047 X0048)))))))

IT IS NOT THE CASE THAT DR SMITH FAILED TO CHALLENGE JOHN .

((JOHN /, X0048) ((DR$SMITH /, X0047) (IN-THE-PAST (CHALLENGE X0047 X0048))))

DR SMITH CHALLENGED JOHN .

Example 5

DR SMITH SAYS THAT A STUDENT FAILED TO LEAVE /.

SEMANTIC REPRESENTATION

```
((E STUDENT /, X0052) ((DR$SMITH /, X0050) (ASSERT I (IN-THE-PRESENT
(CLAIM X0050 (IN-THE-PAST (NOT (COME-ABOUT (EVENT (LEAVE X0052)))))))
)))
```

NON-NP PRESUPPOSITIONS

```
((DR$SMITH /, X0050) (*UNTENSED (HUMAN X0050)))
```

DR SMITH BE -UNTENSED- HUMAN .

```
((E STUDENT /, X0052) ((DR$SMITH /, X0050) (IN-THE-PRESENT (CLAIM X00
50 (IN-THE-PAST (ATTEMPT (EVENT (LEAVE X0052)))))))
```

DR SMITH CLAIMS THAT SOME STUDENT ATTEMPTED TO LEAVE .

NP-RELATED PRESUPPOSITIONS

```
((DR$SMITH /, X0050) (*UNTENSED (IN-THE-SHARED-INFO X0050)))
```

DR SMITH EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

```
((E STUDENT /, X0052) ((DR$SMITH /, X0050) (IN-THE-PRESENT (CLAIM X00
50 (NOT (IN-THE-PAST (LEAVE X0052)))))))
```

DR SMITH CLAIMS THAT IT IS NOT THE CASE THAT SOME STUDENT LEFT .

Example 6

IF JOHN LEFT /, THEN MARY APPRECIATED THAT HE LEFT /.

SEMANTIC REPRESENTATION

```
((MARY /, X0056) ((JOHN /, X0054) (ASSERT I (IF-THEN (IN-THE-PAST (LE
AVE X0054)) (IN-THE-PAST (APPRECIATE X0056 (FACT (IN-THE-PAST (LEAVE
X0054)))))))))
```

NON-NP PRESUPPOSITIONS

```
((JOHN /, X0054) (IF-THEN (IN-THE-PAST (LEAVE X0054)) (IN-THE-PAST (L
EAVE X0054))))
```

IF JOHN LEFT THEN JOHN LEFT .

```
((MARY /, X0056) ((JOHN /, X0054) (IF-THEN (IN-THE-PAST (LEAVE X0054)
) (*UNTENSED (HUMAN X0056)))))
```

IF JOHN LEFT THEN MARY BE -UNTENSED- HUMAN .

NP-RELATED PRESUPPOSITIONS

```
((JOHN /, X0054) (*UNTENSED (IN-THE-SHARED-INFO X0054)))
```

JOHN EXIST -UNTENSED- IN THE SHARED INFORMATION .

```
((JOHN /, X0054) (IF-THEN (IN-THE-PAST (LEAVE X0054)) ((MARY /, X0056
) (*UNTENSED (IN-THE-SHARED-INFO X0056)))))
```

IF JOHN LEFT THEN MARY EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

Example 7

IF JOHN MANAGED TO LEAVE /, THEN MARY WILL ADMIRE HIM /.

SEMANTIC REPRESENTATION

```
((MARY /, X0060) ((JOHN /, X0058) (ASSERT I (IF-THEN (IN-THE-PAST (CO
ME-ABOUT (EVENT (LEAVE X0058)))) (IN-THE-FUTURE (ADMIRE X0060 X0058))
))))
```

NON-NP PRESUPPOSITIONS

```
((JOHN /, X0058) (IN-THE-PAST (ATTEMPT (EVENT (LEAVE X0058)))))
```

JOHN ATTEMPTED TO LEAVE .

NP-RELATED PRESUPPOSITIONS

```
((JOHN /, X0058) (*UNTENSED (IN-THE-SHARED-INFO X0058)))
```

JOHN EXIST -UNTENSED- IN THE SHARED INFORMATION .

```
((JOHN /, X0058) (IF-THEN (IN-THE-PAST (COME-ABOUT (EVENT (LEAVE X005
8)))) ((MARY /, X0060) (*UNTENSED (IN-THE-SHARED-INFO X0060)))))
```

IF JOHN MANAGED TO LEAVE THEN MARY EXIST -UNTENSED- IN THE SHARED INFORMATION .

ENTAILMENTS

Example 8

REFERENCES

Charniak, Eugene, "Toward a Model of Children's Story Comprehension". Artificial Intelligence Laboratory Report AI TR-266. Cambridge, MA: Massachusetts Institute of Technology, 1972.

Charniak, Eugene, "Jack and Janet in Search of a Theory of Knowledge". In Proceedings of the Third International Joint Conference on Artificial Intelligence. Palo Alto, CA: Stanford Research Institute, 1973.

Fillmore, C.J., "Verbs of Judging: an Exercise in Semantic Description". In Fillmore and Langendoen, Studies in Linguistic Semantics. New York: Holt, Rinehart, and Winston, 1971.

Givon, Talmy, "The Time-Axis Phenomenon". Language 49,4(1973): 890-925.

Harris, Z.S., "Two Systems of Grammar: Report and Paraphrase". In Harris, Structural and Transformational Linguistics. Dordrecht, Holland: D. Reidel Publishing Co., 1970.

Isard, S., "What Would You Have Done If...?" Unpublished, 1974.

Joshi, A.K., and Weischedel, R.M., "Some Frills for Modal Tic-tac-toe: Semantics of Predicate Complement Constructions", IEEE Transactions on Electronic Computers Special Issue on Artificial Intelligence, April 1976.

Karttunen, Lauri, "On the Semantics of Complement Sentences". In Papers from the Sixth Regional Meeting of the Chicago Linguistic Society. Chicago: University of Chicago, 1970.

Karttunen, Lauri, "Presuppositions of Compound Sentences". Linguistic Inquiry IV(1973): 169-193.

Karttunen, Lauri, "Presupposition and Linguistic Context". Theoretical Linguistics 1, 1/2(1974):182-194.

Karttunen, Lauri and Peters, Stanley, "Conventional Implicature in Montague Grammar". Presented at the First Annual Meeting of Berkeley Linguistic Society, February 15, 1975. Berkeley, CA.

Keenan, Edward, L., A Logical Base for English. Unpublished Doctoral Dissertation, University of Pennsylvania, 1969.

Keenan, Edward L., "Two Kinds of Presupposition in Natural Languages". In Fillmore and Langendoen, Studies in Linguistic Semantics. New York: Holt, Rinehart, and Winston, 1971.

Keenan, Edward L., "On Semantically Based Grammar". Linguistic Inquiry III(1972): 413-461.

Kiparsky, Paul and Kiparsky, Carol, "Fact". In Steinberg and Jakobovits, Semantics. New York: Cambridge University Press, 1971.

Lakoff, George, "Presupposition and Relative Well-formedness". In Steinberg and Jakobovits, Semantics. New York: Cambridge University Press, 1971.

Lehnert, Wendy, "What Makes Sam Run? Script Based Techniques for Question Answering". In Proceedings of the Workshop on Theoretical Issues in Natural Language Processing. Association for Computational Linguistics, 1975.

Rosenschein, Stanley J., Structuring a Pattern Space, with Applications to Lexical Information and Event Interpretation. Ph.D. Dissertation, University of Pennsylvania, 1975.

Sager, Naomi, "The String Parser for Scientific Literature". In Rustin, Natural Language Processing. New York: Algorithmics Press, 1973.

Schank, R., "The Fourteen Primitive Actions and Their Inferences". Stanford A.I. Memo-183. Palo Alto, CA: Computer Science Department, Stanford University, 1973.

Schank, Roger C., and Reiger, Charles J., III. "Inference and the Computer Understanding of Natural Language". Stanford A.I. Memo-197. Palo Alto, CA: Computer Science Department, Stanford University, 1973.

Schank, Roger C., et. al. "Inference and Paraphrase by Computer". Journal ACM 22, 3(July 1975): 309-328.

Schank, Roger C., "Using Knowledge to Understand". In Proceedings of the Workshop on Theoretical Issues in Natural Language Processing. Association for Computational Linguistics, 1975.

Smaby, Richard, "Consequence, Presuppositions and Coreference". To appear in Theoretical Linguistics, 1975.

Weischedel, Ralph M., "Computation of an Unique Class of Inferences: Presupposition and Entailment". Ph.D. Dissertation, University of Pennsylvania, 1975.

Weischedel, Ralph M. "A New Semantic Computation While Parsing: Presupposition and Entailment," Technical Report #76, Department of Information and Computer Science, University of California, Irvine, CA, 1976.

Wilks, Yorick, "A Preferential, Pattern-seeking, Semantics for Natural Language Inference". Artificial Intelligence 6(1975): 53-74.

Winograd, T., Understanding Natural Language. New York: Academic Press, 1972.

Woods, W.A., "Transition Network Grammars for Natural Language Analysis", Comm. ACM 13,10 (October 1970): 591-606.

Woods, W.A., "An Experimental Parsing System for Transition Network Grammars". In Rustin, Natural Language Processing. New York: Algorithmic Press, 1973.

Yamanaski, Masa-aki, "Lexical Decomposition and Implied Proposition". Unpublished paper. University of Michigan, 1972