INFORMATION TO USERS

This was produced from a copy of a document sent to us for microfilming. While the most advanced technological means to photograph and reproduce this document have been used, the quality is heavily dependent upon the quality of the material submitted.

The following explanation of techniques is provided to help you understand markings or notations which may appear on this reproduction.

1. The sign or “target” for pages apparently lacking from the document photographed is “Missing Page(s)”. If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to assure you of complete continuity.

2. When an image on the film is obliterated with a round black mark it is an indication that the film inspector noticed either blurred copy because of movement during exposure, or duplicate copy. Unless we meant to delete copyrighted materials that should not have been filmed, you will find a good image of the page in the adjacent frame. If copyrighted materials were deleted you will find a target note listing the pages in the adjacent frame.

3. When a map, drawing or chart, etc., is part of the material being photographed the photographer has followed a definite method in “sectioning” the material. It is customary to begin filming at the upper left hand corner of a large sheet and to continue from left to right in equal sections with small overlaps. If necessary, sectioning is continued again—beginning below the first row and continuing on until complete.

4. For any illustrations that cannot be reproduced satisfactorily by xerography, photographic prints can be purchased at additional cost and tipped into your xerographic copy. Requests can be made to our Dissertations Customer Services Department.

5. Some pages in any document may have indistinct print. In all cases we have filmed the best available copy.
TOWARD A COMPUTATIONAL THEORY OF PRAGMATICS

- DISCOURSE, PRESUPPOSITION, AND IMPLICATURE

DISSERTATION

Presented in Partial fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By
Takao Gunji, B.S., M.S., M.A.

* * * *

The Ohio State University

1981

Reading Committee:
H. William Buttelman
Balakrishnan Chandrasekaran
David R. Dowty

Approved By

[Signature]

Adviser
Department of Computer and Information Science
ACKNOWLEDGMENTS

I would like to express my gratitude to my adviser, Bill Buttelmann, for his patience and understanding during the preparation of the dissertation on a topic which involves diverse disciplines.

Because of the interdisciplinary nature of the work, I have greatly benefited from the comments given by the other members of the reading committee, who I consider to be among the best representatives of the two fields closely related to computational linguistics – B Chandra from artificial intelligence and David Dowty from linguistics.

I would also like to thank Norm Sondheimer for his encouragement to tackle model theory in the environment of computation at the early stage of my thinking. Thanks are also due to Mark Steedman, who shares the interest in studying language in the computational environment for his thorough comments on an earlier version of the dissertation.
VITA

September 29, 1951 .................................... Born - Tokyo, Japan
March, 1974 ............................................. B.S., University of Tokyo, Tokyo, Japan
March, 1976 ............................................. M.S., Department of Physics, University of Tokyo, Tokyo, Japan
September, 1978 - June, 1981 ..................... Research Associate, Department of Computer and Information Science, The Ohio State University, Columbus, Ohio

July, 1981 - August, 1981 ............................ Research Associate, Department of Linguistics, The Ohio State University, Columbus, Ohio
September, 1981 ..................................... M.A., Department of Linguistics, The Ohio State University, Columbus, Ohio
September, 1981 - December, 1981 ................. Research Associate, Department of Computer and Information Science, The Ohio State University, Columbus, Ohio

PUBLICATIONS

1. Analysis of Hash Addressing Methods, Technical Report 76-03, Department of Information Science, University of Tokyo, Tokyo, 1976.


**FIELDS OF STUDY**

Major Field: Computer and Information Science

  Professor H. William Buttelmann

- Studies in Artificial Intelligence.  
  Professor Balakrishnan Chandrasekaran

- Studies in Linguistics.  
  Professor David R. Dowty
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>11</td>
</tr>
<tr>
<td>VITA</td>
<td>11</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. THE SYNTAX AND THE SEMANTICS OF EIL</td>
<td>5</td>
</tr>
<tr>
<td>2.1. Extended Intensional Lisp</td>
<td>5</td>
</tr>
<tr>
<td>2.2. Syntax of EIL</td>
<td>6</td>
</tr>
<tr>
<td>2.3. Semantics of EIL</td>
<td>9</td>
</tr>
<tr>
<td>3. COPMUTATION AND COMPOSITIONAL SEMANTICS</td>
<td>18</td>
</tr>
<tr>
<td>4. SUPER-INTERPRETATION AND THE DISCOURSE STRUCTURE</td>
<td></td>
</tr>
<tr>
<td>4.1. Language-Game</td>
<td>24</td>
</tr>
<tr>
<td>4.2. Hierarchical Universes of Discourse</td>
<td>27</td>
</tr>
<tr>
<td>4.3. UDs and Discourse Anaphora</td>
<td>28</td>
</tr>
<tr>
<td>4.4. Tense and Other Operators</td>
<td>34</td>
</tr>
<tr>
<td>5. SAID, IMPLICATED AND PRESUPPOSED</td>
<td>38</td>
</tr>
<tr>
<td>5.1. Grice's Program</td>
<td>38</td>
</tr>
<tr>
<td>5.2. Taxonomy of Implicatures</td>
<td>42</td>
</tr>
<tr>
<td>5.3. Aspects of Presupposition</td>
<td>49</td>
</tr>
<tr>
<td>5.4. Psychological Reality of Cancelability</td>
<td>54</td>
</tr>
<tr>
<td>6. PRECONDITIONS AND POSTCONDITIONS</td>
<td>56</td>
</tr>
<tr>
<td>6.1. Representation and Interpretation</td>
<td>56</td>
</tr>
<tr>
<td>6.2. Pragmatic Procedures</td>
<td>58</td>
</tr>
<tr>
<td>6.3. Implicatures of Factive Verbs</td>
<td>61</td>
</tr>
<tr>
<td>6.4. Degree of Cancelability</td>
<td>66</td>
</tr>
<tr>
<td>6.5. Gricean Implicatures</td>
<td>66</td>
</tr>
<tr>
<td>7. PROJECTION PROBLEM</td>
<td>70</td>
</tr>
<tr>
<td>7.1. Cumulative Hypothesis</td>
<td>70</td>
</tr>
<tr>
<td>7.2. &quot;Plugs,&quot; &quot;Holes,&quot; and &quot;Filters&quot;</td>
<td>73</td>
</tr>
<tr>
<td>7.3. &quot;Extension,&quot; &quot;Implicature,&quot; and &quot;Heritage&quot;</td>
<td>80</td>
</tr>
<tr>
<td>7.4. &quot;Im-Plicature&quot; and &quot;Pre-Supposition&quot;</td>
<td>89</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

Of the trichotomy of the study of language (signs) by Morris (1938) — syntax (or "syntactics" in Morris' original terminology), semantics, and pragmatics — pragmatics seems to be the least formalized field, partly because it inevitably involves the users of language in addition to the system of signs, which syntax takes care of, and the real world to which the signs refer, which is part of the task of semantics.

For the treatment of the real world, we have at least one formal solution, namely, model-theoretic semantics, where a formal, set-theoretic, 'model' of the real world is defined. Although the 'model' employed here may seem to be too simplified and too abstract at first sight, I believe it is an approximation in the right direction in the sense that we can further elaborate the approximation in any way we desire to get an infinitesimally closer approximation of the real world.

On the other hand, there doesn't seem to be any predominant formal theory which applies to pragmatics equally well as model theory applies to semantics. Based on each person's discipline, taste or prejudice, etc. different approaches are taken almost chaotically to the problems of pragmatics. Some will try to reduce pragmatics to "garden variety semantics," or perhaps even to syntax, as generative semanticists once tried (Lakoff 1970). Some will try to make additions to model theory to cope with such phenomena as indexical (or deictic) expressions — I, you, this, etc. — (Lewis 1970, Montague 1970a). A less formal approach is represented by the theory of speech acts (Austin 1962, Searle 1969), and by the conversational maxims by Grice (1975). Psychological, or "cognitive" explanations are also tried in psychology and artificial intelligence (Schank 1975).

The approach taken here is an augmentation, or reformulation, of model-theoretic semantics by computational components. As is mentioned above, model theory gives a reasonable approximation to deal with the real world. It is a formal theory compatible with the computational environment. The "model" has a natural counterpart in the present-day digital computer by the name of "data base." (I will use these two terms interchangeably in the following.) Moreover, computational components can contribute to pragmatics by exploiting the concept of procedure, which, I hope, provides a natural model of
the processing of utterances performed by the language user. Since there is no counterpart of the language user in the conventional model theory, except perhaps the treatment of indexical expressions for the first and the second personal pronouns, computational reformulation of model theory offers a new perspective in semantics and pragmatics. In a way, model-theoretic semantics is naturally expanded to a "computational model-theoretic pragmatics."

In the system I am going to describe in this paper, I will try to incorporate some of the promising approaches to pragmatics, such as the Gricean maxims as long as they are compatible with the computational environment. Although they are less formalized than the main body of model theory, some of them can have natural computational realizations.

One of the things which I emphasize as computational features in the system is the mandatory requirement of sequential execution. In this respect, a multi-argument function, for example, is no longer a set-theoretical function, i.e., a set of ordered n-tuples. Take a two-place predicate P. In Montague (1973, henceforth PTQ), P(a,b) is simply a notational variant of P(b)(a), i.e., the result of applying P(b) to a, where P(b) is the result of applying P to b. However, when there is a possibility of side-effects during the processing of functional application, which is typically the case in the present-day von Neumann type computers, where the data and the program are not distinct internally, the order of application is significant in giving a desired result. In P(a,b), P is applied to the pair <a,b> in one step, while in P(b)(a), P is applied to b first, then the result is applied to a. It is possible that during the processing of P(b) the environment is changed so that the value (denotation) of a, b, or P is changed. Moreover, it is also possible that the execution of P(b) will be trapped in an error situation and an unusual termination occurs. Furthermore, since the halting problem for Turing machines is undecidable, P(b) may never terminate to give a result. Thus, the order of evaluation may produce significant difference to the final outcome.

Our evaluation mechanisms of logical formulas are designed to take advantage of these characteristics of computation — sequential evaluation and side-effects. In fact, the concept of constantly changing environment by the addition or the deletion of information to or from the model by the side-effects, which are performed during the sequential evaluation, provides a powerful mechanism to deal with such pragmatic aspects as "presuppositions" and "implicatures." The model, or the data base, is used as the "context" on which to base the evaluation.

In our system, semantics and pragmatics are handled separately by different components. Each lexical item is assigned both a procedure to handle compositional semantics and procedures to handle pragmatics.
These two kinds of procedures, however, do interact via the model. The model, which behaves as the read-only memory for semantic procedures, can be modified by pragmatic procedures. Thus, the results of pragmatic procedures will cause different consequences for semantic procedures because the model, i.e., the "context" changes. This kind of interaction is, I believe, what underlies such phenomena as "presuppositions" and "implicatures."

Another closely related property of computational model is its incompleteness. Unlike the set-theoretical model, what is realized as the "model" (data base) in a computer can only represent part of the real world. This, however, corresponds more naturally to what is called "the universe of discourse." Missing information or undefined situation is not unusual. The model contains inherently insufficient information, and the conversational contribution of an utterance is to supply new information, or to eliminate old contradictory information. That is, the partial model is constantly updated during the course of conversation. What follows is an attempt to explain some of the pragmatic problems, particularly "presuppositions" and "implicatures" from this information-processing point of view.

Our computational system, whose ultimate goal is a reasonable interactive question-answering system that is capable of handling human-machine communication by natural language, consists of several components. A parser analyzes an input English utterance according to a phrase structure grammar which is based on PTQ and augmented by the proposals by Cooper (1975), Gazdar (1981, to appear) and others. The output of the parser consists of two elements. The first shows the mood of the sentence and the second is the "sentence-radical" (Stenius 1967). The structure the second element shows is called an analysis tree, where quantifier scopes are disambiguated, all the categories of the lexical items are identified so that their functional relations are explicit, some of the pronoun references are determined, ellipses are recovered, etc. The analysis tree is fed to a translator, which translates it into a formula of EIL (Extended Intensional Lisp), which is a computationally enriched formal language based on Lisp and Montague's intensional logic used in PTQ. The formula is then evaluated by an interpreter, which recursively computes the denotation of the formula of EIL. At the same time, the interpreter causes necessary changes to the model based on the non-denotational properties of some of the lexical items of EIL. This manipulation of the model by the interpreter is one of the major mechanisms to cope with pragmatics in our system. Finally, a super-interpreter takes an appropriate action based on the denotation of the formula and the mood of the input sentence. Both the interpreter and the super-interpreter embody part of the non-truth-conditional aspects of the language use. I will concentrate on the interpreter and the super-interpreter in the following, leaving the descriptions of the parser and the translator only cursory[1].
Those successive stages are not intended to suggest a hypothesis about actual human processing of language. Unlike a cognitive approach, our system is not meant to be a psychological model to simulate human behaviors, although it may in the end reflect some of the aspects of human processing. After all, computers cannot be too far from human beings as long as they are made and programmed by human beings.

In the next chapter, the syntax and the semantics of the intermediate language EIL is fully described. This is the main device we are going to use in computationally treating semantics and pragmatics. After a brief discussion of computation in the context of compositional semantics in Chapter 3, the following chapters treat the main topic of the paper: pragmatics. In Chapter 4, we will see how the concept of super-interpretation is utilized in giving a higher-order interpretation of EIL formulas, as well as a dynamic organization of the data base as the context. Following Chapter 5, where some terminological remarks and the definitions of technical terms used in this dissertation are presented, Chapter 6 exhibits the capacity of the present system in coping with non-denotational aspects of natural language. Chapter 7 pays attention to a traditional problem concerning "presuppositions" and "implicatures." A solution possible in the current approach will be presented and compared with other formal proposals.

[1] The parser may have to be more closely knit with the interpreter and the super-interpreter than it is assumed to be here in order to resolve ambiguities, e.g., different scope relations based on the context. Since the parser itself can only list possible analysis trees, the determination or preference of a particular tree must be based on the information "fed back" from the interpreter and the super-interpreter. Such a mechanism is not included because it seems to be outside the scope of this dissertation. Rather unrealistically, the parser is assumed to produce one and only analysis tree as the input to the translator without the interaction with the context set up by the interpreter and the super-interpreter.
2. THE SYNTAX AND THE SEMANTICS OF EIL

2.1. EXTENDED INTENSIONAL LISP

Extended Intensional Lisp (EIL) is a logical programming language which is used in our system as an intermediate language. As Montague (1973) used his intensional logic as an intermediate language between the fragment of English and the set of denotations, the semantics of a natural language (English fragment) is given based on EIL in our system. As is indicated in Montague (1970b), the use of an intermediate language is not of logical necessity. Rather, it is a matter of convenience since the semantic rules which would be given directly to an English fragment would be too complicated to state and almost unintelligible. I adopt the same approach in our system to the computational semantics and pragmatics: a natural language sentence is analyzed and disambiguated by the parser first; then the analyzed tree structure is translated into a formula of EIL, which in turn is evaluated by the interpreter, and, along with the mood indicator, executed by the super-interpreter. To use a computer metaphor, EIL serves as the executable object code compiled from the source language (English fragment). I do not intend to claim that EIL corresponds to any linguistic or psychological entity, such as "deep structure," "semantic representation," "logical form," or "conceptual dependency," etc. The syntax and the semantics of EIL is systematic, if not simple, and more understandable, as compared with the syntax and the semantics of English itself. The parser and the translator are assumed to take care of many of the clarifications of the complication inherent in natural language (e.g., undoing morphophonemic changes, clarifying quantifier scopes, introducing extensional predicates by means of meaning postulates, determining some of the pronoun references, recovering ellipses, etc.). Thus, the EIL formulas are, though still close to the surface sentences, free from complications arising from these aspects. This level of representation will be extensively used when we talk about semantic and pragmatic operations in the following. See Section 6.1 for more discussion on representation, in particular in connection with non-denotational aspects.

The interpreter of EIL is the machine implementation of the formal semantics of EIL given below. Thus, I will sketch how the interpreter works by presenting the formal semantics of EIL, rather than presenting and talking about the actual code in Lisp, in which
the EIL system is written, though the metalanguage itself used to
describe the semantics contains many informal notations.

EIL is based on the programming language Lisp and Montague
(1973)'s intensional logic. In fact, his intensional logic is
included in EIL as a proper subset; EIL is "extended" in several
ways. First, denotations of well-formed expressions are considered
basically as executable procedures. Even truth-values are executed by
the super-interpreter to produce some kind of action. Other
procedures can directly interact with the model and this feature will
turn out to be one of the most important ingredients in our system in
coping with pragmatics. See Chapter 4 for one kind of example and
Chapter 6 for another. Second, to each primitive expression in EIL,
the precondition and the postcondition are assigned in addition to its
denotation. These two additional assignments are also procedures and
executed before and after computing the denotation, respectively.
These mechanisms will be utilized in dealing with non-denotational
aspects of language use including "presuppositions" and
"implicatures." These procedures are described in detail in Chapter 6.
Another extension is that EIL allows the "undefined" case, which will
occur due to the lack of sufficient information, since the data base
is only a partial model of the real world. Since each formula of EIL,
as a program, is actively interpreted, any suitable action can be
taken for the undefined case; we can embed such routines as
error-handling, user inquiry, or execution termination, etc., in the
interpreter or the super-interpreter as part of the system. This
feature allows us more flexible and reasonable treatment of undefined
cases, which often come up in pragmatics, than traditional approaches.
See Chapter 4 for some of the possibilities of such treatment. The
following two sections give formal definitions which set the basis for
the later descriptions of the system.

2.2. SYNTAX OF EIL

I will first briefly describe the syntax of EIL. As will be
seen, the semantic rules will correspond one-to-one to the syntactic
rules. Every well-formed expression in EIL is assigned a type, that
is, well-formed expressions are subcategorized according to their
types. The syntax, or the set of formation rules, refers to the types
of the constituent expressions. The set of types is recursively
defined as follows: starting with two basic types t and e, any
combination of the form <a1,a2, ... ,an,b> or <s,b> is a type if a1, a2,
..., an, and b are types. As can be seen from the following rules,
well-formed expressions of type t correspond to formulas (i.e.,
translations of sentences which will have truth values as
denotations), well-formed expressions of type e correspond to
individual constants and variables, well-formed expressions of type
<a1,a2, ... ,an,b> correspond to functions with arguments of types a1,
a2, ... , an, and the value of type b. <s,b> is a special type for
intensional expressions whose extensions are of type b. In the following, $La$ for any type $a$ denotes the set of well-formed expressions of type $a$. Capital letters and parentheses belong to the object language (EIL), and small letters are used as meta variables over types and well-formed expressions of EIL.

(2.1) a. Any variable $v$ of type $a$ is a member of $La$.

b. Any constant $c$ of type $a$ is a member of $La[1]$.

c. If $p$ is a member of $La$, then

$(\text{QUOTE } p)$ is a member of $La[2]$.

d. If $p_1$, $p_2$, ..., $p_m$ are members of $Lt$, and $q_1$, $q_2$, ..., $q_m$ are members of $La$, then

$(\text{COND } (p_1 q_1) (p_2 q_2) ... (p_m q_m))$ is a member of $La[3]$.

e. If $v_1$, $v_2$, ..., $v_m$ are variables of types $a_1$, $a_2$, ..., $a_m$, respectively and $p$ is a member of $Lb$, then

$(\text{LAMBDA } (v_1 v_2 ... v_m) p)$ is a member of $L_{a_1,a_2,...,a_m,b}[4]$.

f. If $p$ is a member of $L_{a_1,a_2,...,a_m,b}$, and $q_1$, $q_2$, ..., $q_m$ are members of $La_1$, $La_2$, ..., $La_m$, respectively, then

$(p q_1 q_2 ... q_m)$ is a member of $Lb[5]$.

g. If $p$ and $q$ are both members of $La$, then

$(\text{EQU } p q)$ is a member of $Lt[6]$.

h. If $p$ is a member of $La$, then

$(\text{INT } p)$ is a member of $L_{s,a}[7]$.

i. If $p$ is a member of $L_{s,a}$, then

$(\text{EXT } p)$ is a member of $La$.

[1] Many of the Lisp functions are included in EIL as EIL constants, e.g., $\text{EQ}$ for equality of atoms (primitive expressions). In addition, constants of type $t$, i.e., $T$ and $F$, are names for truth values, true and false, respectively. NIL is reserved for an empty list.

[2] As can be seen later in (2.4c), the effect of $\text{QUOTE}$ is to freeze the evaluation. One of the utility of this construction is to make explicit distinction between the "program" and the "data." Note that in a list $(A B C)$ $A$ is always treated as a function and $B$ and $C$ as arguments (cf. (2.1f)). $(\text{QUOTE } (A B C))$, on the other hand, is simply a list of three things: $A$, $B$ and $C$. $(\text{QUOTE } p)$ is often abbreviated as '$p$'.

[3] This is a multiple conditional. Each pair of $(p_i q_i)$ corresponds roughly to "if $p_i$ then $q_i$." See (2.4d) for the exact definition. This construction allows one to write complicated programs.

[4] This is Church (1941)'s lambda abstraction.

[5] This is a functional application of $p$ to arguments $q_1$, $q_2$, ..., $q_m$. As has been mentioned, multi-variable functions may have different results from multiple applications of higher-order functions of the form $\ldots ((p q_m) q_{m-1}) ... q_1$.

[6] $\text{EQU}$ is the equality operator, and is an extension of the standard $\text{EQ}$ of Lisp. Cf. (2.4g).

[7] $\text{INT}$ stands for intension, $\text{EXT}$ in (2.1i) for extension.
If \( p \) is a member of \( L_t \), then
\[(\text{NOT } p) \text{ is a member of } L_t.\]

If \( p \) and \( q \) are members of \( L_t \), then
\[(\text{AND } p \ q) \text{ is a member of } L_t[8].\]

If \( p \) and \( q \) are members of \( L_t \), then
\[(\text{OR } p \ q) \text{ is a member of } L_t.\]

If \( p \) and \( q \) are members of \( L_t \), then
\[(\text{IMP } p \ q) \text{ is a member of } L_t.\]

If \( v_1, v_2, \ldots, v_m \) are variables of types \( a_1, a_2, \ldots, a_m \), respectively, and \( p \) is a member of \( L_t \), then
\[(\text{EVERY } (v_1 \ v_2 \ldots \ v_m) \ p) \text{ is a member of } L_t[9].\]

If \( v_1, v_2, \ldots, v_m \) are variables of types \( a_1, a_2, \ldots, a_m \), respectively, and \( p \) is a member of \( L_t \), then
\[(\text{SOME } (v_1 \ v_2 \ldots \ v_m) \ p) \text{ is a member of } L_t.\]

If \( p \) is a member of \( L_t \), then
\[(\text{NEC } p) \text{ is a member of } L_t[10].\]

If \( p \) is a member of \( L_t \), then
\[(\text{POS } p) \text{ is a member of } L_t.\]

If \( p \) is a member of \( L_t \), then
\[(\text{PAST } p) \text{ is a member of } L_t[11].\]

If \( p \) is a member of \( L_t \), then
\[(\text{PUT } p) \text{ is a member of } L_t.\]

Implicit in (2.1) is the convention that only those expressions that are constructed by a finite number of applications of rules in (2.1) are well-formed. The rules apply recursively to generate an infinite number of expressions.

As can be seen, some constructions come from Montague's intensional logic and some from ordinary Lisp. As a logic, EIL is an intensional, higher-order, tense, and typed logic with equality. As a programming language, it is a Lisp with many predefined logical primitives, which are necessary in order for EIL to be an effective intermediate language in the study of natural language semantics and pragmatics. Note that the syntax of EIL is not intended to be minimal; in fact, many of the primitives can be mutually defined (cf. next section).

[8] Logical connectives AND, OR, and IMP (material implication) appear as prefixes in EIL.
[9] EVERY is for universal quantification, SOME in (2.1o) for existential quantification. Multi-variable quantification is allowed in EIL.
[10] NEC is for the modal necessity operator (the "box"), POS in (2.1q) for the possibility operator (the "diamond").
[11] PAST is for the past tense operator, PUT in (2.1s) for the future tense operator.
2.3. SEMANTICS OF EIL

The semantic rules will be described corresponding to the above syntactic constructions. That is, each syntactic construction determines one semantic relation. Thus, the semantics is also recursive. This has an important consequence because the semantics of EIL is closely related to the semantics of natural language. As in the construction in Montague (1970c), the semantic mapping from natural language to the real world in our system is realized by a composition of the translation function from natural language to EIL and the semantic function from EIL to (the model of) the real world. As has been mentioned above, the intermediate translation process is rather a matter of convenience and theoretically there is no difference between the direct (without intermediate EIL) and indirect (via EIL) semantics. To impose recursiveness, or the Principle of Compositionality, as is commonly called, on the semantics of EIL is, then to comply with the Fregean principle concerning natural language semantics: the meaning (reference) of a sentence is computed from the meanings (references) of its constituents. This is desirable since it makes it possible for a finite number of semantic rules to compute meanings of an infinite number of sentences, just as a finite number of syntactic rules can generate and analyze an infinite number of sentences. Thus, the overall natural language understanding system based on EIL can be said to be an attempt to apply this principle to the extent that nontrivial aspects of language use are handled. One of my major theses is that, in so doing, computational machinery or at least computational interpretation of the process of language understanding, is essential in dealing with pragmatics which seems to necessitate some degree of noncompositionality. As can be seen later, the computational devices for pragmatics do allow a deviation from the strict compositionality, but they can still be embedded naturally in the compositional semantics, maintaining the overall compositionality of the semantics.

As has been mentioned, the overall framework on which the semantics of EIL is based is model theory[12]. Since EIL is an intensional and tense logic, the model includes the set of possible worlds and the set of moments of time. Formally a model for EIL is a quintuple $M = <D,W,T,V,f>$, where $D$ is called the set of domains, $W$ the set of possible worlds, $T$ the set of moments of time, $V$ the value assignment function, and $f$ the variable assignment function. Each member of $D$ corresponds to a type and related as follows: $D_e$ is the set of individual entities (which are denotations of EIL expressions of type $e$), $D_t$ is the set of truth values, $T$ and $F$, $D<k_1,a_2,\ldots,a_n,b>$ is the set of functions from $D_{a_1} \times D_{a_2} \times \ldots \times D_{a_n}$ to $D_b$, and $D<s,b>$ is the set of functions from $W \times T$ to $D_b$. In addition, $D$ contains $U$ ("undefined") as a typeless (or all-type) element[13]. The domains of $V$ and $f$ are well-formed expressions and variables in EIL, respectively. If $p$ is an expression of type $a$, then the value of $V$ for $p$ is a member of $D_a$. Similarly, if $v$ is a variable of type $a$, 


then \( f(v) \) is a member of \( Da \). The value assignment \( V \) plays the major role in the semantics. It recursively evaluates an expression of EIL. Actually, the function \( V \) written in Lisp constitutes the major component of the interpreter.

In close association with the function \( V \) is a lexicon for constants in EIL[14]. The lexicon is an essential part of the model which contains all the definitions of the constants used in EIL, i.e., it is the most elaborate description of the world. However, not every

---

[12] Model theory is no stranger in computer science. For example, in the field of formal linguistics, Fregean compositional semantics has been used for context-free languages in Knuth (1968)'s "attribute" semantics. Moreover, in the study of programming languages, model theory has been utilized under the name of the "denotational semantics" (cf. Stoy 1977), which is one of the promising approaches to the semantics of programming languages. Strange enough, model-theoretic semantics, or any semantic approach to logical expressions for that matter, has not gained much recognition in the field of artificial intelligence. This is perhaps because the initial apparent "success" of syntactic approaches to logic, e.g., the resolution principle (Robinson 1965), were immediately followed by a major disappointment when people realized that the uniform mechanical method encountered a combinatorial explosion problem. As the result, many researchers had, incorrectly, an impression that the use of logic itself is not promising in artificial intelligence. Some embryonic arguments in favor of using logic with model-theoretic semantics in a computational natural language understanding system can be found in Sondheimer and Gunji (1978) and Gunji and Sondheimer (1980).

[13] Note that the domain includes \( U \) ("undefined"). \( U \) simply shows that the denotation, as a function, is a partial function due to the fact that the model is only a partial representation of the real world. Though it looks like a "third truth value" if the expression happens to be of type \( t \), its utility is far wider in the system with respect to the UDs. Cf. Section 4.1.

[14] The lexicon at the EIL level should not be confused with the lexicon at the surface English level. The latter is used by the parser and contains syntactic information such as the category, the gender, the case, etc. The former, which is used by the interpreter and the super-interpreter, contains semantic and pragmatic information such as the denotation, the precondition, the postcondition, etc. These two kinds of lexicons are kept separate because of the possibility of more than one surface lexical items corresponding to a single lexical item in EIL, and vice versa. For example, there is no single lexical item in EIL which corresponds to English articles; they are translated into quantified constructions using such EIL lexical items as SOME, EVERY, AND, and IMP. The distinction between two kinds of lexicons will become more important when we discuss implicatures (cf. Section 6.1). In the following, the word lexicon is used only in the sense of the EIL lexicon.
bit of information in the model is actively used in the process of conversation. In a plausible situation of discourse, only a limited portion of the model is used as an "active" universe of discourse. For example, an utterance of every boy need not have the domain of all the boys on earth; it may have only the boys in the classroom, or the neighbor's sons, as its domain if the previous discourse so dictates. This kind of consideration leads us to an additional device in model-theoretic semantics, which has several aliases and somewhat different definitions or specifications depending on the Christener. Among others are "assumed facts" (Karttunen 1973), "common background" (Stalnaker 1974), "(linguistic) context" (Karttunen 1974), "common (or mutual) knowledge" (Lewis 1975), "Pragmatic Universe of Discourse" (Kempson 1975), "common ground" (Karttunen and Peters 1979), and "contextual domain" (McCawley 1979). Adopting the suggestion of McCawley (1979), who was inspired by Karttunen (1969), and elaborating their points, the EIL system employs a set of hierarchical universes of discourse (UDs, henceforth): UDO, UD1, ..., UDn, where UDn is the most recently activated UD and contains only the information invoked by the most recent utterance, UDj (n>i>0) are older ones which contain the information invoked at the ith utterance and have accumulated any later information invoked by the following utterances. UDO is the maximal universe of discourse which has accumulated all the information invoked during the discourse since the beginning of time. Hence, UDO is assumed to contain permanent information, i.e., all the information independent of the context. As can be seen from the above informal specification, UDs are ordered according to the following relation with regard to the information defined in them.

\[(2.2) \quad i \leq j \iff \text{every information defined in UD}_j \text{ is defined in UD}_i.\]

Thus, UDn is the least defined UD. Note that if a piece of information is put in UD1, it is also put in every UDj such that j<i, i.e., the system updates lower (older) UDs after adding a new UD.

As McCawley (1979) suggests concerning the possibility of "branching time," it might be desirable to introduce a partial ordering among UDs, or even a lattice structure (Scott 1972, 1976). I will use a linear ordering defined in (2.2) to avoid too much complication, noting the fact that the concept of "branching time" could be taken care of by interweaving UDs with the set of moments of time, which, of course, should be partially ordered. Again, for simplicity, I will ignore such elaboration in this dissertation.

Another kind of elaboration would occur to one's mind. As has been extensively used in systems developed in artificial intelligence, a frame-like structure (cf. Minsky (1975) for the basic idea, and, e.g., Bobrow and Winograd (1977) for a proposal of an implementation) with abundant pointers to connect little chunks of the universe of discourse establishing a sophisticated information network in the universe, might be desired in a practical system. This aspect is also
UDs are manipulated by the super-interpreter and pragmatic procedures, which will be described in Chapters 4 and 6. See Chapter 4 for a more detailed description of UD. Briefly speaking, during the processing of a sentence, a new UD is created, which becomes UD_{n+1} if the latest one was UD_{n}. This newest UD will contain the following entities if they are newly invoked by the sentence just processed: explicitly mentioned individual constants, referents of existentially quantified variables (what Karttunen (1969) calls "discourse referents"), possible worlds which have made a formula of the form (POS p) true, moments of time which have made future and past formulas true, etc. I will use the term discourse referent rather broadly to refer to all of these entities in the following. By condition (2.2), any entity newly added to UD_{n+1} will be added also to all UD_i such that i<n. Using UD, the interpretation of quantification is performed first with respect to the intersection of the domain corresponding to the type of the variables and UD_{n}. Then, the interpreter goes down the hierarchy to more defined UD_{j} until the search succeeds. Thus, UD will reduce the domains to be searched to just the part which is pragmatically required. In the following, UD will appear in the interpretation of quantifiers, modal operators and tense operators. Pragmatic procedures are free to change the members of UD; moving an element from UD_i to UD_j, where i<j, will achieve the "focusing" of that element in UD, since it now has a higher priority of being used as a discourse referent.

In the following, in order to convey the computational process, I will use, as a meta-language, a hybrid language of English and a programming language called C (Kernighan and Ritchie 1978), which is convenient to express the procedural aspects concisely. Appendix A presents the complete definitions in C of some of the important portions of the system, which includes the pragmatic procedures which will be discussed in later chapters. In the following, some expressions are used with special technical meanings particular to C (with some customization specific to the purpose of this dissertation).

(2.3) a. ‘expl=exp2’ is true if both expl and exp2 are defined and are equal to each other. It is false otherwise (i.e., if expl or exp2 is undefined or if they are defined and are not equal to each other).

b. ‘if (expl) exp2 else exp3’ is the value of exp2 evaluated after the evaluation of expl if the value of expl turns out to be true. If the value of expl turns out to be false, it is the value of exp3 evaluated after the evaluation of expl. If the ‘else exp3’ part is missing, the next statement following ‘exp2;’ is executed.
The order of evaluation in (2.3b) is significant, since the evaluation of `exp1` may cause side-effects on the model (data base) in which to evaluate `exp2` or `exp3`. Note also that `exp3` is not evaluated if `exp1` evaluates to true, and `exp2` is not evaluated if `exp1` evaluates to false. This property will be utilized in dealing with non-truth-conditional aspects (cf. Chapter 7). In addition, a sequence of statements which behave as a single compound statement are enclosed by `{` and `}`. `++i` increases the value of `i` by 1 and makes it the new value of `i`, i.e., it is equivalent to `i=i+1`. Similarly for `--i`. The operator `!` is a logical negation. In this context, the meaning is more restricted than the standard definition in C `!a` is `F` if `a` is `T`, `T` if `a` is `F`, and `U` if `a` is `U` (the so-called "internal negation" as opposed to the "external negation" which gives the value `T` to `!a` if `a` is `U`, which is the standard definition in C). The statement 'break' causes the control to go outside of the smallest loop by 'for' etc.

Using these notations, the recursive definition of `V` is as follows, where the right-hand side of "def=" shows the value of `V` assigned by the rule expressed in English-C notation. Let `n` be the level of the newest UD. The entire definition is dependent on the parameter `i`, which lies between `n` and 0 inclusive and shows the level of the UD in question for quantification[15].

(2.4) a. If `v` is a variable of type `a`, then
\[
V[v,w,t,f,n,i] \\
\text{def=} f(v) \ n \ \text{UDi}
\]

[15] The values of `n` and `i` are set by the super-interpreter and passed to the interpreter. Thus, the super-interpreter, does at least the following operations for the utterance `s`:
```c
{
++n;
UDn={};
for (i=n; i>=0; --i)
{val=V[s,w,t,f,n,i];
  if (val) ... ;
  if (!val) ... ;
}
```
See Appendix A, as well as Chapter 4 for more details.
b. If \( c \) is a constant of type \( a \), then
\[
V[c,w,t,f,n,i] = \text{the value corresponding to } c \text{ with respect to } w \text{ and } t \text{ retrieved from UD}_1
\]
c. If \( p \) is a member of \( L_a \), then
\[
V[(QUOTE p),w,t,f,n,i] = p
\]
d. If \( p_1, p_2, \ldots, p_m \) are members of \( L_t \), and \( q_1, q_2, \ldots, q_m \) are members of \( L_a \), then
\[
V[(COND (p_1 q_1) (p_2 q_2) \ldots (p_m q_m)),w,t,f,n,i]
\]
\[
\text{def} = \{ \text{for } (j=1; j<=m; ++j) \}
\text{if } (V[p,j,w,t,f,n,i]==T)
\{V[q,j,w,t,f,n,i];
\text{break}\}
\}
\]
e. If \( v_1, v_2, \ldots, v_m \) are variables of types \( a_1, a_2, \ldots, a_m \), respectively, and \( p \) is a member of \( L_b \), then
\[
V[(LAMBDA (v_1 v_2 \ldots v_m) p),w,t,f,n,i]
\]
\[
\text{def} = \text{the function } h \text{ from } D_{a_1} \times D_{a_2} \times \ldots \times D_{a_m} \text{ to } D_b \text{ such that for any set of values } u_1, u_2, \ldots, u_m \text{ from } D_{a_1}, D_{a_2}, \ldots, D_{a_m}, \text{ respectively, } h(u_1, u_2, \ldots, u_m) \text{ is identical to } V[p,w,t,f',n,i], \text{ where } f' \text{ is the same as } f \text{ except possibly that } f'(v_1)==u_1, f'(v_2)==u_2, \ldots, f'(v_m)==u_m
\]
f. If \( p \) is a member of \( L_{<a_1,a_2,\ldots,a_m,b>} \) and \( q_1, q_2, \ldots, q_m \) are members of \( L_{a_1}, L_{a_2}, \ldots, L_{a_m} \), respectively, then
\[
V[(p q_1 q_2 \ldots q_m),w,t,f,n,i]
\]
\[
\text{def} = \{ \text{val}=V[p,w,t,f,n,i];
\text{if } (\text{val is defined})
\text{the result of applying val to arguments}
\langle V[q_1,w,t,f,n,i], V[q_2,w,t,f,n,i], \ldots,
V[q_m,w,t,f,n,i]\rangle;
\text{else } U\}
\]
g. If \( p \) and \( q \) are both members of \( L_a \), then
\[
V[(EQU p q),w,t,f,n,i]
\]
\[
\text{def} = \{ \text{val1}=V[p,w,t,f,n,i];
\text{val2}=V[q,w,t,f,n,i];
\text{if } (\text{val1 and val2 are both defined})
\{\text{if } (\text{val1 == val2}) \text{T};
\text{else } F}\}
\text{else } U\}
\]

[16] This definition will be revised when we consider preconditions and postconditions (cf. Chapter 6 and Appendix A). Note that some of the arguments \( q_i \) may evaluate to \( U \) depending on the function \( p \), e.g., predicate \textsc{EXIST} should have the truth-value \text{false} for an undefined argument.
h. If \( p \) is a member of \( L_a \), then
\[
V[(\text{INIT } p), w, t, f, n, i] = \text{the function } h \text{ from } W \to D_a \text{ such that for any } x \text{ in } W, h(x) = V[p, x, t, f, n, i]
\]
i. If \( p \) is a member of \( L_{<s,a>} \), then
\[
V[(\text{EXT } p), w, t, f, n, i] = \begin{cases} V[p, w, t, f, n, i] & \text{if } (\text{val is defined}) \\ \text{the result of applying val to } <w,t> & \text{else } U \end{cases}
\]
j. If \( p \) is a member of \( L_t \), then
\[
V[(\text{NOT } p), w, t, f, n, i] = \begin{cases} (\text{val} = V[p, w, t, f, n, i]) & F \\ \text{else } T \end{cases}
\]
k. If \( p \) and \( q \) are members of \( L_t \), then
\[
V[(\text{AND } p, q), w, t, f, n, i] = \begin{cases} \text{val} = V[p, w, t, f, n, i] & \text{if } (\text{val} = V[q, w, t, f, n, i]) \\ \text{else if } (\neg \text{val}) & F \\ \text{else if } (\neg \text{val}) & T \\ \text{else if } (\neg \text{val}) & T \\ \text{else } U \end{cases}
\]
l. If \( p \) and \( q \) are members of \( L_t \), then
\[
V[(\text{OR } p, q), w, t, f, n, i] = \begin{cases} \text{val} = V[p, w, t, f, n, i] & \text{if } (\text{val}) \\ \text{else if } (\neg \text{val}) & V[q, w, t, f, n, i] \\ \text{else if } (\neg \text{val}) & T \\ \text{else if } (\neg \text{val}) & T \\ \text{else } U \end{cases}
\]
m. If \( p \) and \( q \) are members of \( L_t \), then
\[
V[(\text{IMP } p, q), w, t, f, n, i] = \begin{cases} \text{val} = V[p, w, t, f, n, i] & \text{if } (\text{val}) \\ \text{else if } (\neg \text{val}) & V[q, w, t, f, n, i] \\ \text{else if } (\neg \text{val}) & T \\ \text{else if } (\neg \text{val}) & T \\ \text{else } U \end{cases}
\]

[17] \text{AND as defined in (2.4k) is not a truth-function in the usual sense, since each conjunct is sequentially evaluated one by one, and one of them may be undefined or not evaluated at all in order for } (\text{AND } p, q) \text{ to have a truth value. Similarly for other logical constants. One of the utilities of these definitions will be appreciated when we discuss the projection problem in Chapter 7. The assignment of truth values } T \text{ and } F, \text{ and the undefined case } U, \text{ coincides with Thomason (1979), though I am not advocating any kind of "presuppositional logic." The introduction of the undefined case } U \text{ itself plays a rather minor role in the treatment of presuppositions in the total system. } U \text{ simply shows a lack of information and has little to do with the situation where the "presupposition" doesn't hold.}
n. If \( v_1, v_2, \ldots, v_m \) are variables of types \( a_1, a_2, \ldots, a_m \), respectively, and \( p \) is a member of \( L_t \), then

\[
V[(\text{EVERY} (v_1 v_2 \ldots v_m) p), w, t, f, n, i]
\]

\[
def = \begin{cases} 
\text{T} & \text{if } (V[p, w, t, f', n, i] = \text{T} \text{ for all sets of values } u_1, u_2, \ldots, u_m \text{ in } D_{a_1} \cap U_D, D_{a_2} \cap U_D, \ldots, D_{a_m} \cap U_D, \text{respectively, where } f' \text{ is the same as } f \text{ except possibly that } f'(v_1) = u_1, f'(v_2) = u_2, \ldots, f'(v_m) = u_m) \text{ T}; \\
\text{F} & \text{else if } (V[p, w, t, f', n, i] = \text{F} \text{ for some set of values } u_1, u_2, \ldots, u_m \text{ in } D_{a_1} \cap U_D, D_{a_2} \cap U_D, \ldots, D_{a_m} \cap U_D, \text{respectively, where } f' \text{ is as above}) \text{ F}; \\
\text{U} & \text{else} 
\end{cases}
\]

\[
\text{else U}
\]

\[
p. \text{ If } p \text{ is a member of } L_t, \text{ then}
\]

\[
V[(\text{NEC} p), w, t, f, n, i]
\]

\[
def = \begin{cases} 
\text{T} & \text{if } (V[p, x, t, f, n, i] = \text{T} \text{ for all } x \text{ in } W \cap U_D) \text{ T}; \\
\text{F} & \text{else if } (V[p, x, t, f, n, i] = \text{F} \text{ for some } x \text{ in } W \cap U_D) \text{ F}; \\
\text{U} & \text{else U} 
\end{cases}
\]

\[
q. \text{ If } p \text{ is a member of } L_t, \text{ then}
\]

\[
V[(\text{POS} p), w, t, f, n, i]
\]

\[
def = \begin{cases} 
\text{T} & \text{if } (V[p, x, t, f, n, i] = \text{T} \text{ for some } x \text{ in } W \cap U_D) \text{ T}; \\
\text{F} & \text{else if } (V[p, x, t, f, n, i] = \text{F} \text{ for all } x \text{ in } W \cap U_D) \text{ F}; \\
\text{U} & \text{else U} 
\end{cases}
\]

\[
r. \text{ If } p \text{ is a member of } L_t, \text{ then}
\]

\[
V[(\text{PAST} p), w, t, f, n, i]
\]

\[
def = \begin{cases} 
\text{T} & \text{if } (V[p, x, t, f, n, i] = \text{T} \text{ for some } x \text{ in } T \cap U_D \text{ such that } x< t) \text{ T}; \\
\text{F} & \text{else if } (V[p, x, t, f, n, i] = \text{F} \text{ for all } x \text{ in } T \cap U_D \text{ such that } x< t) \text{ F}; \\
\text{U} & \text{else U} 
\end{cases}
\]

\[
s. \text{ If } p \text{ is a member of } L_t, \text{ then}
\]

\[
V[(\text{FUT} p), w, t, f, n, i]
\]

\[
def = \begin{cases} 
\text{T} & \text{if } (V[p, x, t, f, n, i] = \text{T} \text{ for some } x \text{ in } T \cap U_D \text{ such that } x> t) \text{ T}; \\
\text{F} & \text{else if } (V[p, x, t, f, n, i] = \text{F} \text{ for all } x \text{ in } T \cap U_D \text{ such that } x> t) \text{ F}; \
\end{cases}
\]
As has been mentioned before, the semantics defined here is compatible with mutual definitions of primitives. For example, $V[(\text{AND } p \ q), w\ t\ f\ n\ i] \equiv V[(\text{COND } ((\text{EQU } p \ T) \ q) \ ((\text{EQU } p \ F) \ F) \ ((\text{EQU } q \ F) \ F) \ (T \ U)), w\ t\ f\ n\ i]$. 

The definitions in (2.4) are far from complete, since many of the pragmatic operations are not yet included. Some of the interactions between the semantics and the pragmatics can, however, be seen in (2.4). For example, in (2.4a), the function $f$ interacts with UDs when $v$ is a free variable. Similarly, the values of constants are determined depending on UDI, which contains only the information relevant to the context. We will see examples of such interaction in Chapter 4. In the quantification, modality and tense rules (2.4n) - (2.4s), starting with $i$ being $n$, each possible value (an individual entity, a possible world, or a moment of time) in UDI is checked one by one whether it satisfies the formula. This process is repeated by decreasing the value of $i$ by 1 until the search succeeds at some level of UD, or fails by unsuccessfully reaching UD0[13].

Remember that the compositionally computed value of $V$ is only the denotation of the translation of the sentence-radical. The total sentence i.e., both the mood and the sentence-radical, is further evaluated by the super-interpreter, as can be seen in Chapter 4. Appendix A gives fuller definitions of some of the important constructions, which include all the pragmatic operations which will be introduced and discussed in Chapters 4 and 6.

Some general remarks concerning the relationship between computation and compositional semantics follow in the next chapter.

[13] At the end of the evaluation of existential quantification, since the interpreter has determined which value satisfies the formula, such a value is registered in the newly created UD, i.e. UDn, as part of the postcondition of these constructions. That is, some entities are re-activated by becoming discourse referents. See Chapter 4 for a detailed discussion of UDs and discourse referents.
Unlike other linguistic systems, the EIL system heavily relies on computational concepts, such as procedurally interpreted model-theoretic semantics, data base as the universes of discourse, etc. It is the main theme of this dissertation that a pragmatically adequate treatment of natural language is nearly impossible without these computational concepts. We will see how discourse is treated via the interaction between semantic (acturally, semantico-pragmatic) interpreters and the universes of discourse in the next chapter. We will see more on pragmatic components concerning "presuppositions" and "implicatures" in the following chapters. This chapter is restricted to more semantic aspects, which, though there have been certain answers from traditional frameworks, a computational re-interpretation or expansion, is expected to provide more straightforward, and more intuitive, understanding.

The concept of computation has long existed even before the advent of digital electronic computers. The word 'algorithm,' which is now used almost synonymously with 'computer program,' owes its origin to the name of a 9th century Persian textbook author (Knuth 1973). In 1936, A.M. Turing defined a theoretical concept which is now called a 'Turing machine' (Turing 1936), which still holds the fundamental position in the theory of computation just as set theory holds the fundamental position in mathematics. A Turing machine has a head which can only move sequentially and which can write or erase a symbol on a tape[1]. It would not be necessarily impossible to build a model-theoretic semantics directly on a Turing machine; we have at least one attempt (Tichý 1969). But, according to the so-called Church-Turing thesis, which states that Turing machines can do anything which is "computable," digital electronic computers are no more or no less powerful than Turing machines, as well as any conceivable concept of computation including that of human beings. Thus to base the EIL system on such modern constructs is no more or no less advantageous than using Turing machines. Moreover, the fundamental concepts remain essentially the same: sequential control and the addition/deletion of information during the interaction with the environment[2]. We will see how the computational environment

---

[1] As is well-known, multi-head Turing machines and multi-tape Turing machines are no more powerful than single-head, single-tape Turing machines in the sense that they can generate and recognize exactly the same class of sets -- recursively enumerable sets.
acts as the universes of discourse in the next chapter. We will see how the concept of sequential control will play an important role in pragmatics later (cf. Chapters 6 and 7). The word 'computational' or 'procedural' is, then, still closely related to the original computational concept by Turing. Note that the concept of Turing machine is at a higher level of abstraction as compared to set theory, even though, mathematically speaking, Turing machines are definable using only the concepts of set theory and the set-theoretically defined concept of function. However, once they are defined, they exist on their own and serve as the next level of primitives to define still higher concepts and constructs, such as digital computers with random-access memory. It is exactly these concepts that I am taking advantage of when coping with semantico-pragmatic aspects of natural language.

Procedural interpretation of model-theoretic, particularly possible-world, semantics has not been uncommon, at least as a metaphor (without implementation on a computer). For example, Tichý (1971), and Kraut (1976), among others identify the model-theoretic concept of 'intension' with an identification, or a search, procedure. Since intensions are functions from the set of possible worlds, this is quite a natural interpretation. As has often been said in support of the introduction of the concept of intension, what we know about a word is not its current referent but the way to determine its referent for any given situation. Thus, we can talk about unicorns without any psychological difficulty even though we have never seen one, and we are sure to be able to recognize a unicorn if we encounter one. In the EIL system, a procedure is a Lisp-EIL function which performs the specified task. Thus, the denotation of an intensional expression is a Lisp-EIL function with one argument over possible worlds that gives the extension depending on the value of the argument; more specifically, a function which is given to \((\text{INT } p)\) has the form \((\lambda(W) \text{ZEVAL } p W t f))\), where ZEVAL is the name of the system function which performs the task of the V function (the value assignment function) defined in Section 2.3, and \(t\) and \(f\) have the same meanings as those in that section (the current moment of time and the variable assignment function, respectively). The evaluation of \(p\) is suspended in this form until the world variable \(W\) is bound to the given world. What is essential for intensional constructs is to preserve (or "freeze" the evaluation of) the original form until the appropriate environment is given. In this sense, intensional constructs are similar to Lisp's FEXPR functions, which suspends the

[2] So-called parallel processing seems to aid computation from a point of view of efficiency, and should be considered in constructing a practical system, which the EIL system is not intended to be at this stage. It doesn't, however, seem to alter any conceptual fundamentals of computation definable in terms of Turing machines. Remember that multi-head Turing machines can be simulated by a single-head Turing machine.
evaluation of their arguments until it is explicitly stated in the procedure. (Cf. the DC function which appears in (3.1) later. The arguments, e.g., MAN, are not evaluated; they are treated as themselves, i.e., as strings.) Note that usual Lisp functions, EXPRs, evaluate their arguments first and then execute whatever procedures defined for them based on the values of the evaluation of the arguments. For example, in (PLUS (TIMES 2 3) 4), the inner formula, (TIMES 2 3), is evaluated first to give 6 and then the formula (PLUS 6 4) is evaluated. This is in accord with the principle of compositionality, where the values of the evaluation of higher constructs depend only on the values of the evaluation of lower constructs. Intensions and their semantics based on possible-world semantics are well-contrived way of circumventing apparent difficulty of compositionality. Likewise, FEXPRs are later addition to the original "pure" Lisp (McCarthy 1960), which was strictly compositional.

Quantification is another complication to simple compositionality. As we have seen in Chapter 2, a formal model theory calls for an additional device -- the variable assignment function $f$ — to conform to compositionality; apparently unbound variables in subformulas will get proper interpretation by the aid of the variable assignment function. From a computational point of view, this is a recursive call of the interpretation procedure with a modified parameter, which is the variable assignment function $f$ in the case of quantification, and is implemented in EIL based on the device of 'association list' in the pure Lisp. This procedural interpretation of quantification may also be called game-theoretic (Hintikka 1974, where the "game" is between Myself and Nature). Quantification is also a search procedure, perhaps more intuitively so than intensions since the domain of search is more intuitive; for the first order logic the domain is the set of individuals.

Search can be performed in any way as one wants using the full capacity of Lisp and its extension by EIL facilities. For example, the denotation of a predicate MAN, which is the translation of man, could be defined either as a set, or a list, of individuals who are men, as in (3.1a) below, or in a more efficient way, which would look like (3.1b):

\begin{align*}
(3.1) \text{a.} & \quad (\text{DC MAN} \, <e,t> \, (X)) \\
& \quad \text{DEN:} \, (\text{MEMBER X} \, (\text{GET 'D-LIST: 'MAN}))) \\
\text{b.} & \quad (\text{DC MAN} \, <e,t> \, (X)) \\
& \quad \text{DEN:} \, (\text{EQ} \, (\text{GET 'SEX: X} \, '\text{MALE})))
\end{align*}

where DC is a system function used for defining constants, $<e,t>$ is the type of the predicate $(X)$ is the list of the lambda variable (in this case there is only one variable: $X$ of type $e$), and DEN: shows that this is a definition of the denotation procedure of MAN (other kinds of procedures will be defined later in Chapter 6). Each
constant in EIL is associated with a 'property list,' which is a standard equipment in Lisp, that describes the properties of the constant. The DC function creates a Lisp-EIL function from the given inputs and puts it in the property list of the constant under the property name DEN'. For example, the value of the property DEN for MAN after the execution of (3.1a) is (LAMBDA (X) (MEMBER X (GET 'D- LIST; 'MAN))). According to (3.1a), the denotation procedure first GETs the denotation list, or the d-list of MAN, which is a list of entities that satisfy the predicate and sees if the individual bound to X is a MEMBER of that list[3]. (The functions GET and MEMBER are also standard equipments of Lisp.) On the other hand, as for (3.1b), the denotation procedure first GETs the value of the property SEX: for the individual bound to X, and sees if it is EQUAL to MALE. If so, then the individual bound to X satisfies the predicate, otherwise the value is false. (3.1b) eliminates a search for the individual in a list of men, which is performed by MEMBER in (3.1a); all the interpreter has to do is check the property list of the individual, which is usually shorter than a list of men. The function EQ is a much simpler operation and doesn’t involve a search. Moreover, the kind of search involved in the GET type function could be realized in a computationally efficient way (e.g., hashing) even if the property list gets long. Note that the definition of the type suggested in (3.1b) is more of an intensional nature than (3.1a), which is a simple, exhaustive search, even though (3.1b) is meant to be a definition of the extension of MAN. To define the intension of MAN in a similar way is straightforward. The PTQ semantics for intensional logic takes this kind of approach; every constant is given an intensional definition rather than an extensional one.

Defining constants in the way suggested in (3.1b) provides a systematic way of imposing "meaning postulates" (Carnap 1956) to the model. For example, Carnap’s example concerning bachelor and unmarried in the form of the meaning postulate (3.2) below can be stated in the meta-language as (3.3).

(3.2) (EVERY (X) (IMP (BACHELOR X) (NOT (MARRIED X))))

(3.3) a. (DC MARRIED <e,t> (X)
    DEN' (ZMARRIED X))

b. (DC BACHELOR <e,t> (X)
    DEN' (COND ((ZMARRIED X) F)
               ((ZMAN X) (ZADULT X))
               (T F)))

[3] Along with a d-list, a negative denotation list, or an nd-list, which is a list of entities that don’t satisfy the predicate will be used later. As is discussed here, these lists are used solely for the reason of their simplicity.
In (3.3), the two predicates are related via a meta-language function \texttt{ZMARRIED}. In (3.3b), since unmarriedness is a necessary condition for bachelorhood, the interpreter first checks the individual's marital status. If s/he is unmarried, then other properties -- sex and adulthood -- will be checked. (Note that in (3.3), the functions \texttt{ZMARRIED}, \texttt{ZMAN}, and \texttt{ZADULT} belong to the meta-language.) On the other hand, (3.2) is a syntactic construction and the utilization of the formula like (3.2) itself in an interpretational system is not straightforward, since such a system usually has no provision for an axiomatic theorem proving facility. (3.2) is best understood as an additional constraint on the model so that only the model where (3.2) is true is permitted, and one of its natural interpretations would be to define part of the model in the way specified by (3.3), which constitute part of the lexicon of EIL.

Similarly, Montague's meaning postulates to get first-order equivalents of extentional predicates, e.g., the translation of \texttt{find}, can be stated in the manner exhibited by (3.3). For example, in PTQ the translation of \texttt{find}, say \texttt{FIND}, has a first-order equivalent, say \texttt{FIND*}, which is related to \texttt{FIND} by the following meaning postulate.

\begin{equation}
\text{(3.4)} \quad \text{(EVERY } (X \text{ PP}) \left( \text{NEC (EQU (FIND } X \text{ PP}) \left( (\text{EXT } PP) \left( \text{INT (LAMBDA } (Y) \text{ (FIND* } Y \text{ X Y)))} \right) \right) \right)))
\end{equation}

where \texttt{PP} is a variable of type \texttt{<s,<e,t>},t>> and \texttt{X} and \texttt{Y} are variables of type \texttt{e}. Alternatively, \texttt{FIND} could be defined directly using a two-place meta-predicate say, \texttt{ZFIND*}:

\begin{equation}
\text{(3.5)} \quad \text{(DC FIND } <e,\texttt{<s,<<e,t>>,t>>},t> \text{ (X PP)} \left( \text{DEN (EXT } PP) \left( \text{INT (LAMBDA } (Y) \text{ (ZFIND* } Y \text{ X Y}))} \right) \right)
\end{equation}

(The current version of the EIL system, however, performs the translation of the kind described above syntactically and after the translation, first-order equivalents, if they exist, appear in the formula, mainly because of the consideration of efficiency; syntactic lambda and EXT-INT conversion is less costly than interpreting unconverted, complicated formulas with lots of INTs and LAMBDA s. Thus, in our system, \texttt{FIND}, for example, designates a predicate of type \texttt{<e,e,t>}, i.e., it corresponds to \texttt{FIND*} above.)

In PTQ, the translation of proper names are sets of properties of individuals, not individual constants. This has the advantage of providing uniform treatment for all the term phrases (noun phrases) including quantified phrases. For example, the translation of \texttt{Jack} is \texttt{(LAMBDA } (P) ((EXT } P \text{) } J))), where \texttt{P} is a variable of type \texttt{<s,<e,t>}, and the translation of \texttt{a man} is \texttt{(LAMBDA } (P) \texttt{(SOME } (X) \texttt{(AND (MAN } X \texttt{) ((EXT } P \texttt{) } X)))}). They both can combine with an intension of an intransitive verb, e.g., \texttt{(INT WALK)}, to make a formula, such as \texttt{(WALK } J)\texttt{) or (SOME } (X) \texttt{(AND (MAN } X \texttt{) (WALK } X)))\texttt{). The individual constant } J \texttt{denotes a
person named Jack, though the linguistic expression Jack itself
denotes a set of properties the person Jack has. As has been seen in
(3.1), a property list is associated with each individual constant in
EIL. Thus, the property list of J contains Jack’s sex, age, etc. By
utilizing this facility, we could directly define the denotation of
the translation of Jack, say JACK, as a constant of type \(<s, <e, t>>, t>\) as in (3.6):

\[
(3.6) \quad \text{DC JACK } <s, <e, t>>, t> \quad \text{(P)}
\]
\[
\text{DEN: } \text{(GET (EXT P) 'JACK))}
\]

Although this approach hasn’t been taken in the current version of
EIL, since it doesn’t seem to increase efficiency, (3.6) exhibits one
way of defining the denotation of a proper name directly as a set of
properties using the facilities of the Lisp-EIL system, without
reducing it to a formula containing individual constants.

Various other simplifications and modifications, as well as
extensions, have been implemented in the EIL system, though I will not
attempt to list all of those details since they will easily lead to
too much technicality, which will obscure the essential contribution
of the EIL system to computational linguistics (and linguistics proper
if there is any difference between the two). In the following
chapters, I will discuss what I think are conceptually more important
components of the system: pragmatic components.
4. SUPER-INTERPRETATION AND THE DISCOURSE STRUCTURE

4.1. LANGUAGE-GAME

At the final stage of the processing of a sentence after all the compositional semantic interpretation is performed, the truth-values activate different actions by the super-interpreter depending on the mood of the sentence. In this sense, the super-interpreter performs a "post-compositional" interpretation. The basic idea behind this two-step interpretation is Stenius (1967)'s distinction between modal elements and sentence-radicals and his concept of "language-game" based on that distinction. The interpreter interprets only the translation of the sentence-radical. The input to the super-interpreter is the mood of the sentence -- indicative, imperative, or interrogative -- and the denotation, i.e., the truth value, of the sentence-radical as well as the translation of the sentence-radical itself. Thus, for example, the translation of the sentence-radical of both "Jack is bald" and "Is Jack bald?" is (BALD J), while the former has the mood indicator <Indicative>, and the latter <Interrogative>.

The strategy of the super-interpreter is based on "a convention of truthfulness and trust" (Lewis 1975, which is an adaptation of Stenius' proposal of "language-game") and also reflects Grice (1975)'s Cooperative Principle, which will be discussed in detail in the next chapter. These principles are reformulated in the environment of computation in terms of the action of the super-interpreter with respect to the current UDs and the utterer of the input sentence; i.e., in terms of the processing of information upon receiving an utterance. In the following, the truth-values and the undefined case U will get uniform interpretations with respect to the UDs regardless of the mood: T designates that the corresponding information is existent in the current UDs; F shows that a contradictory fact is present in the UDs; and U is an indication that there is no relevant information whatsoever in the UDs. Based on this interpretation of the truth-values, the super-interpreter realizes part of the illocutionary act depending on the mood.

Let us examine each mood one by one. The super-interpreter plays a language-game according to the following rules for indicative sentences. Let UDn be the newly created UD upon receiving an utterance.
If the mood is \textless Indicative\textgreater , and
\begin{itemize}
  \item[a.] if the denotation of the sentence-radical is \( T \), then update the lower UDs so that they satisfy (2.2) with respect to UDn.
  \item[b.] if the denotation of the sentence-radical is \( F \), then notify the utterer of the contradiction and require further information to clarify the situation.
  \item[c.] if the denotation of the sentence-radical is \( U \), then add the new information which corresponds to the sentence radical to UDn. All lower UDs are updated accordingly to satisfy condition (2.2).
\end{itemize}

(4.1) is rather strange from the viewpoint of Grice's \textit{maxim of Quantity} which expects the utterer to be "as informative as is required." Since the system already "knows" the content of the sentence-radical in the case of (4.1a), the sentence is surely not informative. In other words, since it exists at some lower level of UDs (thus giving the truth value \( T \)), it is not entirely new information. But, while no new information is added, the discourse referents have been re-activated during the interpretation by making them members of the most active UD. (We will see how discourse referents are added to UDn in Section 4.3.) Thus, the effect of stating a true statement is changing the prominence of entities in the hierarchy of UDs, rather than presenting itself as new information.

(4.1b) is an apparent violation of the convention of the truthfulness, Stenius's (R3), or another maxim of Grice's: the \textit{maxim of Quality}, namely, "Try to make your contribution one that is true." The super-interpreter's reaction can be flexible; (4.1b) only suggests one possibility.

(4.1c) also comes from the above principle. The super-interpreter assumes the truthfulness of the utterer and changes the current UDs so that the utterance becomes true in the updated UDs. For example, for an utterance "Jack is bald," the d-list of BALD in the lexicon is augmented so that it includes \( J[1] \). As Gazdar (1979a, p.45ff) notes, and Lewis (1975) actually formulates, the maxim of Quality is more of a conventional nature than of a conversational principle. At the expense of irony, metaphor, lies, and other interesting aspects of human communication, this maxim is treated as a convention here and cannot be violated in our system.

\[^{[1]}\text{}\] Of course, what the super-interpreter should do depends on how the denotation of the predicate is expressed as a program. In order to enforce uniformity and simplicity, I use the devices d-lists and nd-list mentioned in Chapter 3 in the following. Needless to say, these lists are only some of possible implementations (in fact, not so efficient ones) of the definitions of predicates, as we have seen in the previous chapter.
One final note about the indicative. As has been mentioned, the model is inherently incomplete and allows the occurrence of U. However, as (4.1b) shows, once a piece of information is inserted in the model, it remains intact and cannot be overridden by an indicative statement. In this sense, we are assuming a fixed model, even though it is incomplete and can be augmented. Although it may sometimes be desirable to treat a false indicative statement as an indirect command to update the model, this kind of use of indicative sentences — in fact, the use of an indicative as an imperative (cf. (4.2b) below) — is ignored here. It would necessitate a more satisfactory theory of indirect speech acts and lies beyond the scope of this dissertation.

Rules for the imperative mood are given as:

(4.2) If the mood is <Imperative>, and
a. if the denotation of the sentence-radical is T, then take no action.
b. if the denotation of the sentence-radical is F, then change the UDs so that the sentence-radical becomes true.
c. if the denotation of the sentence-radical is U, then ask more information to settle the situation.

In fact, for (4.2a), some action should be taken, since this is an unusual situation: what is ordered has already been performed. One of the alternatives which the super-interpreter could take would be to report the situation to the utterer and wait for another command. (4.2b) is based on the truthfulness of the hearer (the system), which corresponds to Stenius's rule (R4). (4.2c) is a rather conservative approach to an undefined situation. One could dare to perform what is ordered just as in (4.2b). Unlike the indicative mood, discourse referents are not usually introduced for imperatives and yes-no questions as Karttunen (1969)'s examples show.

(4.3) a. A: Does John have a car?
   B: *It is a Mustang.

b. Give me a hotdog, please. *It looks delicious.

As for (4.3a), the second person cannot use it to refer to John's car unless s/he first confirms the question by uttering yes or some other affirmative phrase. As Schachter (1978) points out, English yes and no are 'pro-sentences' and equivalent to affirmative and negative indicative sentences. Thus, in the case of yes preceding it, it can successfully refer to John's car since the first indicative (pro-)sentence yes establishes the discourse referent. (Note that yes in this context is equivalent to "John has a car.") Similarly, the use of it in (4.3b) is not justified unless an extralinguistic gesture, such as pointing to a particular hotdog, establishes a discourse referent. Such extralinguistic phenomena are ignored here. (As for linguistically established discourse referents, see Section 4.3.)
The rules for interrogatives are straightforward:

\[(4.4)\text{ If the mood is } \langle \text{Interrogative} \rangle \text{ (yes/no question), and }\]
\[\begin{align*}
a. & \text{ if the denotation of the sentence-radical is } T, \text{ then answer 'yes.'} \\
b. & \text{ if the denotation of the sentence-radical is } F, \text{ then answer 'no.'} \\
c. & \text{ if the denotation of the sentence-radical is } U, \text{ then answer 'insufficient information,' or the like.} \\
\end{align*}\]

(As for WH-questions, the translation of the sentence-radical is essentially the same as that of existentially quantified sentences, e.g., who is treated as essentially equivalent to for some person, thus, the translation of the sentence-radical of "Who is bald?" is \((\text{SOME } X) \text{ (AND } \text{PERSON } X) \text{ (BALD } X)\)). As will be seen in Section 4.3, the evaluation mechanism for existential quantification can determine the entity which makes the formula true during the process of the evaluation. Thus, what the super-interpreter has to do for WH-questions is return that entity (in a human-readable format).)

By this distinction of the mood and the sentence-radical, as well as the two-stage interpretation based on that, we can expand the truth-conditional semantics naturally to a pragmatics which can handle wider range of aspects of language use. We will see other tasks which the super-interpreter and other components can perform both post-compositionally and compositionally in the rest of the chapter.

4.2. HIERARCHICAL UNIVERSES OF DISCOURSE

As has been briefly described in Section 2.3, the system uses a set of partial models as an auxiliary device for pragmatic purposes. Each member of UDs corresponds to the universe of discourse activated from some time in the past up to the present time. Thus, for example, if the most active level is \( n \), UD\( i \) is the universe of discourse activated by the \( i \)th, the \( i+1 \)st, ..., and the \( n \)th utterances. The conversational contribution of the \( i \)th utterance is, therefore, UD\( i \) - UD\( i+1 \). Note that UDs are organized in a retrospective manner. The conversational contribution of the \( n \)th, i.e., the most recent, utterance is UD\( n \) itself. That of the previous (i.e., \( n-1 \)st) utterance is UD\( n-1 \) - UD\( n \), that of \( n-2 \)nd UD\( n-2 \) - UD\( n-1 \), etc. By condition (2.2), information flows from the top (UD\( n \)) to the bottom (UDO), and UDO has all the information from the beginning of time. Thus, as a number of sentences are processed, lower UDs, UD\( i \), UD\( 2 \), etc., gradually accumulate more and more information, starting with only the contribution of a few utterances, and ultimately there would be practically no difference among the lower UDs. In fact, it would be reasonable to assume that there is no discernible distinction among lower UDs once \( n \) becomes sufficiently large. Thus, as a practical and natural approximation of a model of the processing of natural
language, it may suffice to maintain a very limited number of UDs, such as UDn, UDn-1, ..., and UDn-k=UD0, with a small k.[2]

4.3. UDS AND DISCOURSE ANAPHORA

Let us consider how new information is put in UDs in more detail. The following are based on Karttunen (1969) s examples.

(4.5) Jack has a car.
     (SOME (X) (AND (CAR X) (HAVE J X)))

(4.6) a. The car is black.
     (SOME (X) (AND (EVERY (Y) (EQU (CAR Y) (EQU X Y)))
               (BLACK X)))
     
b. It is black.
     (BLACK IT)

c. He is bald.
     (BALD HE)[3]

As has been mentioned, the interpretation of SOME causes as a side-effect putting the referent of the bound variable into UDn. Thus, (2.4o) should be revised as follows to explicitly include this pragmatic operation. (The number of variables is reduced to one for simplicity, and comments are enclosed by /* and */).

[2] Sometimes, it may seem desirable to maintain older versions of UDi, namely, the UDi before it is updated, or the universe of discourse before a particular utterance occurred -- memories of old times. This would lead us to an upper-triangular matrix of UDs: UDi,j (i<j) for the universe of discourse activated by the i-th utterance and accumulated information up to the j-th utterance. What we are using as UDi in the current system would correspond to UDi,n in this two-dimensional system. Note that UDi n is the union of UDi,n-1 and UDn,n.

[3] From now on, most of the English sentences are followed by their EIL translations. By convention, English verbs, nouns, and adjectives in capital letters are used as EIL predicates in the translation. An individual name is represented by the capitalized first letter, which is of type e. All the extensional transformations based on the meaning postulates in PTQ will be applied in the following. Note that CAR here is a translation of car, not the standard Lisp function. The translation of the in (4.6a) is from PTQ, which follows Russell (1905) s definite description. IT in (4.6b) and HE in (4.6c) are free variables of type e.
(4.7) If \( v \) is a variable of type \( a \) and \( p \) is a member of \( L_t \), then

\[
V[(\text{SOME } (v) \ p) \ , \ w \ , \ t \ , \ f \ , \ n \ , \ i] \\
\text{def} = \{\text{save}(\text{UD}n); /* save the current UDn for back-up */} \\
\text{for } (\text{dom} = \text{UD}n D_t; \ \text{dom} \nsubseteq \{}; \ \text{dom} = \text{dom} - \{u\}) \\
\text{/* check for each member of type } a \text{ in } \text{UD}n */ \\
\{u = \text{pick}(\text{dom}); /* pick a value */ \\
\text{val} = V[p, w, t, f, u/v, n, i]; \\
\text{/* } f/v/\text{u}(v) = u, f/v/\text{u} = f \text{ otherwise } */ \\
\text{if } (\text{val}) \{\text{UD}n = \text{UD}n \text{D}_t(u); \\
\text{/* add discourse referent */} \\
\text{return}(T); \\
\} \\
\text{else restore}(\text{UD}n); /* keep on checking */ \\
\} \\
\text{return}(F); \\
\}
\]

In (4.7), 'save' temporarily saves the current UDn so that unwanted changes to UDn doesn't become permanent in the case of the failure of the search. The operation 'restore' recovers the saved UDn. The procedure 'pick' takes out an element from a set. Some pragmatic preference, if any, among the elements may be embedded in this procedure, which is beside the point here.

In processing (4.5), the interpreter tries to establish the truth of the utterance (by the convention of truthfulness). Since an indefinite noun phrase in sentences like (4.5) is usually used to introduce new information, or at least information not mentioned recently, it is usually the case that we cannot find the referent of \( X \) in (4.5) near UDn. Thus, the interpreter will have to go down the hierarchy to the vicinity of UD0, and pick up the referent of \( X \), namely, a car which Jack owns, there. This car is added to UDn by the interpreter according to (4.7) and later to all the lower UDns (UDn-1 through UD1) by the super-interpreter according to (4.1). At the same time, this car is put in the d-list of CAR at the level of UDn as the active member. Similarly, the pair of Jack and the car are put in the d-list of HAVE. The denotation of the individual constant \( J \) itself is also put in UDn. See Appendix A for relevant portions of the system for these operations.

The situation is somewhat different when the interpreter processes (4.6a) after processing (4.5) with UDn as the top level of UDns. The semantic condition of the definite description of (4.6a) requires the interpreter to pick up the unique entity which is a car as the referent of \( X \). Taken without any reservation, this condition is usually considered to be too strong since it is hardly the case that the world contains only one car. But in the EIL system, as can be seen in (2.4n) and (2.4o), the interpreter considers successively
more defined UDs, starting with the least defined but most relevant UD, i.e., UDn in the current example. UDn contains only the referents invoked by the most recent utterance, i.e. (4.5). Thus, the interpreter first considers UDn as the domain of X in (4.6a) as if it were the whole world it knew and successfully identifies the only car there, namely, the car which Jack owns and mentioned in (4.5). Since this car is mentioned in (4.6a), it is put in yet another UD, i.e. UDn+1, during the processing of (4.6a). Also, since the blackness of the car is established in (4.6a), the d-list of BLACK at the level of UDn+1 contains the car as the active member.

Essentially the same process establishes the referent of the free variable IT in (4.6b). Before examining the mechanism, some notes on the background of the translations such as (4.6b) and (4.6c) are in order. Pronouns in natural language are often used loosely in the sense that their translations in a logical language, most likely variables, fall outside the scope of the quantifier which is supposed to bind them, or they are used even without any preceding quantifier. In (4.5) and (4.6b), the referents of X and IT are supposed to be the same entity, but the semantics of the quantifier SOME has no way to enforce the identity of the referents. The same kind of difficulty occurs even within a single sentence, as Geach (1962)'s celebrated "donkey sentence" shows.

(4.8) Every man who owns a donkey beats it.
(EVERY (X) (IMP (AND (MAN X)
    (SOME (Y) (AND (DONKEY Y)
      (OWN X Y))))))
(BEAT X IT))

In (4.8), the second argument of BEAT cannot be the Y which appears in the scope of SOME, since SOME can no longer bind it at that position. We must, then, put a free variable as the second argument of BEAT.

Cooper (1979) has proposed a pragmatic treatment of these so-called "pronouns of laziness," within a formal semantic theory based on Montague semantics. He introduces a free variable of a higher type — that of the set of properties — as a translation of these pronouns. Remember that, as we have seen in the previous chapter, personal pronouns belong to the same category as proper names and they are translated as functions on properties. It is only after the lambda conversion that an individual constant of type e, such as J, takes the argument position of predicates. Thus, the translation of these pronouns in Cooper's system would be (in EIL notation):
(LAMBDA (P) (SOME (X) (AND (EVERY (Y) (EQU ((EXT PF) Y) (EQU X Y)))
  ((EXT P) X)))))

where PF is a free variable of type \(<s, e, t>\). Note that this translation is quite similar to the translation of definite noun phrases and could be inversely translated into a pseudo-English phrase as the PFer, i.e. the unique individual who/which has the property PF. The essence of his translation is, thus, to decompose
pronouns into the plus free property, which is reminiscent of another English construction which has the same semantic content, namely, the one. In Cooper's theory the referent of PF is supposed to be determined pragmatically from the context, although he doesn't give any explicit procedure for doing that task.

Considering this proposal in the context of a mechanical system, the use of a free variable of the type of the set of properties is both impractical and unnecessary. If we simply adopt Cooper's formalization, the domain of PF would be a set of procedures, the cardinality of which is unmanageable even for a small set of individual entities. Moreover, since we have a pragmatic device: a set of UDs, which contributes in reducing the amount of search through the domain by virtue of its hierarchical internal structure, the existence and the uniqueness of the referent, as is entailed by Cooper's translation of pronouns, are established as the by-product of pragmatic determination of the referent. Thus, in the EIL system, pronouns can have much simpler translations, in fact much closer to the surface structure; all the complications in Cooper's translation are handled pragmatically by the interpreter and the super-interpreter[4].

Having discussed the background of the translation, let us go back to (4.6b). The referent of IT is successfully determined as Jack's car mentioned in (4.5) by searching the top-level UD, which has been created for the utterance (4.5), since Jack's car is the only nonhuman object in that UD. Similarly, for the referent of HE in (4.6c), since Jack has been put in the top-level UD during the processing of (4.5), and since he is the only human male entity there, it is successfully determined as Jack. As for the donkey sentence (4.8), by the semantic conditions of IMP and AND, (BEAT X IT) is evaluated only after the evaluation of the antecedent of IMP -- (AND (MAN X) (SOME (Y) (AND (DONKEY Y) (OWN X Y)))) -- turns out to be true, which happens only after both conjuncts of AND evaluate to true. Thus, at the time of processing (BEAT X IT), the processing of the existential quantification has successfully completed and the referent of Y has been determined and added to UDn. This referent is naturally a donkey and owned by the referent of X, and used as the referent of IT. Appendix B shows a detailed step-by-step trace of the interpretation of (4.8)[5].

[4] Thus, the translation of it is (LAMBDA (P) ((EXT P) IT), where IT is a free variable of type e, and similarly for she and he. The humanhood and the gender of the referent are assumed to be part of the semantic conditions of IT, HE, and SHE, and does not appear in the object language. Note, incidentally, that free variables are almost like constants, except that their denotation is almost vacuous. After the lambda conversion, we have formulas like (4.6b) and (4.6c).
Note that we are dealing with the meta-language of EIL, not the object language EIL itself, nor the surface English. Thus, the same referent would be invoked if other words which have the same semantic (and pragmatic) interpretation are used even though the semantic representation and/or the surface structure is different. For example, Jack's car would be properly interpreted as the car mentioned in (4.5). Similarly, the four-wheel vehicle will do the job if the denotation of CAR includes four-wheeledness as well as the subset relation as to the denotation of the predicate VEHICLE. In fact, this kind of definite description is quite common; in journalism, phrases like the 6 foot-3 sophomore guard or the Illinois congressman are used frequently to refer to a particular person previously mentioned in the text.

The scope relationship of quantifiers and negation/modal operators is also adequately handled in our system based on EIL translations. Consider the following.

(4.9) Jack doesn't have a car.
(\text{NOT } (\text{SOME } (X) \text{ (AND } (\text{CAR } X) \text{ (HAVE J } X))))

(4.10) a. *The car is black.
(\text{SOME } (X) \text{ (AND } (\text{EVERY } (Y) \text{ (EQU } (\text{CAR } Y) \text{ (EQU X } Y))))
\text{ (BLACK } X))

b. *It is black
(\text{BLACK } IT)

c. He is bald.
(\text{BALD } HE)

[5] Webber (1979) proposes a computational system based on the similar idea as Cooper's, or the current system's. In her system, an object language expression (her "Level-1 representation") having a free variable, such at IT above, is systematically converted to an expression (her "Level-2 representation") where pronoun references are partially resolved by the help of meta-linguistic expressions called IDs ("Invoking descriptions"). An ID is essentially a Russellian definite description augmented by a conjunct which shows which sentence has "evoked" the entity expressed by the description. She gives explicit rules to create such IDs. Kamp (ms) also presents a treatment of pronouns in the discourse, which includes the donkey sentence and other related phenomena. His system is close in spirit, if not in form, to the current treatment. His idea is to create from the surface structure a deeper structure called a "discourse representation (DR)." A DR contains the discourse referents established by the sentence as well as the representation of the sentence itself and is more powerful than an ordinary "semantic representation;" his DR is in a way an amalgamation of the EIL formula and UDs in our system.
In processing (4.9), the interpreter tries to establish its truth by falsifying the subformula inside the scope of NOT. Thus, if the interpreter is successful, it can conclude that there is no car which Jack owns and hence no discourse referent whatsoever, except for the referent of the proper noun Jack, is put in UDn. (Note that, in (4.7), the UDn is restored to the old status if the search fails.) Thus, although it is possible to talk about Jack as in (4.10c), if the definite description the car, or the pronoun it, is used, the interpreter will have to go down the hierarchy, most likely down to the abyss, UDO, where there will be too many cars to single out only one as the referent. Note that if there is a unique car at some level of UDs above UDO but below UDn, it can be used as the referent:

(4.11) a. Paul has a car.
    (SOME (X) (CAR X) (HAVE P X))

     b. Jack doesn't have a car.
    (NOT (SOME (X) (CAR X) (HAVE J X)))

(4.12) a. The car is black.
    (SOME (X) (AND (EVERY (Y) (EQU (CAR Y) (EQU X Y)))
             (BLACK X)))

     b. It is black.
    (BLACK IT)

In processing (4.12a) or (4.12b) after processing (4.11a) and (4.11b) (in this order), the referent of the car or it is correctly identified as Paul's car, which resides two levels below the top-level UD, since it is the first UD where the interpreter can identify a unique car.

A similar remark as for NOT applies to EVERY. Since it is not the case that a single referent establishes the truth of the universally quantified formula, EVERY is not associated with a pragmatic operation to create a discourse referent (cf. Appendix A). Observe the following, where both the man and he are inappropriate after (4.13).

(4.13) Every man is bald.
    (EVERY (X) (IMP (MAN X) (BALD X)))

    (SOME (X) (AND (EVERY (Y) (EQU (MAN Y) (EQU X Y)))
              (EQU X J)))

     b. *He is Jack.
    (EQU HE J)
Other operators such as tense operators, which establish moments of time as discourse referents, and modal operators, which establish possible worlds as discourse referents, a particular moment of time or a particular possible world will be given a first consideration in processing another sentence which contains these operators. Take the tense operator for example. In (4.15) below, the time adverbial then behaves anaphorically (referring to yesterday) as a "pro-adverb" for time:

(4.15) a. Jack broke a car yesterday.
   (AT YESTERDAY (SOME (X) (AND (CAR X) (BREAK J X))))

b. He was in the car then.
   (AT THEN
   (SOME (X) (AND (EVERY (Y) (EQU (CAR Y) (EQU X Y)))
   (IN HE X)))) [6]

In our system, just like the interpretation of free variable IT or HE in the examples above, THEN is interpreted depending on UDs. In processing (4.15b) after (4.15a), since the processing of (4.15a) establishes the moment of time of Jack's breaking a car, which is yesterday, as a discourse referent, this moment is used to determine the referent of THEN. Let us see the process in more detail. After processing (4.15a), the following reside in UDn: the denotation of J, the car which Jack broke yesterday and was mentioned in (4.15a), and the moment of time when Jack broke that car, which is yesterday. Also, the d-list of CAR for yesterday contains that car, and the d-list of BREAK for yesterday contains the pair of Jack and the car. In (4.15b), HE is properly identified as Jack as we have seen above. In interpreting the free variable THEN in (4.15b), the interpreter first tries the moment of time in UDn, that is, yesterday, and checks the d-list of CAR for yesterday, which contains the car mentioned in (4.15a), and successfully picks up the car which is exactly the one mentioned in (4.15a). Note that (4.15b) states the truth of Jack's being in the car at a particular moment, which is identical to the moment when (4.15a) occurred, not some unspecified past time.

[6] These formulas assume the translation of time adverbials, as sentence adverbs, as (LAMBDA (PR) (AT TA (EXT PR))), where PR is a variable over propositions and TA is a constant (or a free variable for the case of then) for moments of time, and V[(AT r p),w,t,f,n,i] def= {V[p,w,t',f,n,i], where t'=V[r,w,t,f,n,i]}. To make the formula simple, the past tense in the English sentences in (4.15) is treated as the result of obligatory syntactic marking by time adverbials. In other words, it is semantically redundant. See Partee (1973) for arguments concerning this point, and Dowty (1980) and Cooper (ms) for compositional attempts to treat tenses and time adverbials independently.
As Partee (1973) points out, sometimes tense itself, not a time adverbial, behaves as if it were a "pronoun" for time. In this case, the only syntactic element to correspond to the variable is the tense marker in English:

(4.16) a. Jack broke a car yesterday.
   \[ (AT \ YESTERDAY \ (SOME \ (X) \ (AND \ (CAR \ X) \ (BREAK \ J \ X))) \] 

b. The car was black.
   \[ (PAST \ (SOME \ (X) \ (AND \ (EVERY \ (Y) \ (EQU \ (CAR \ Y) \ (EQU \ X \ Y))) \ (BLACK \ X))) \]

In (4.16), the moments of time at which (4.16a) and (4.16b) are evaluated are supposed to be the same, though there is no explicit time adverbial or a variable to be identified as yesterday in the logical formula. Moreover, the semantic condition of PAST itself requires only the existence of a moment of time in the past. Such an interpretation is obviously insufficient in this case. The strangeness of mere existence can be clearly seen if we consider a counterpart of (4.16b) in the case of an indefinite article; we cannot replace (4.6a) by (4.17) below to refer to the same car as in (4.5).

(4.17) A car is black.
   \[ (SOME \ (X) \ (AND \ (CAR \ X) \ (BLACK \ X))) \]

Just as it is used in (4.6b) to refer to the car mentioned in (4.5), the past tense operator in (4.16b) seems like something equivalent to a "pronoun" referring to the time mentioned in (4.16a). In the EIL system, the treatment of (4.16) is not so different from (4.15), where an explicit (free) variable THEN is used. Note that the moment of time for (4.16a), i.e., yesterday, is put in UDn during the processing of (4.16a), and it is considered first in evaluating the past tense operator in (4.16b). Thus, the referent of the car is correctly picked up from the d-list of CAR for yesterday in much the same way as in (4.15) above, checking the blackness only for that car and for yesterday.

The reason why the usual semantic condition for Prior-type tense operators (Prior 1967), which are used in PTQ, cannot adequately handle the cotemporality of the two moments in separate but consecutive utterances, is that the time reference is not explicit at the syntactic level of the logic, and that the semantic conditions for the tense operators only require the existence of moments of time to satisfy the formula. One obvious approach to overcome this difficulty would be an introduction of a variable for the moments of time (together with an explicit time reference operator "AT") in the syntax of the logic (e.g., Dowty 1979, 1980)[7]. That is, the quantification in the meta-language is explicitly stated in the object language. The Priorian tense operators can, then, be defined by existentially quantifying these syntactic time variables. Since it is now possible
to use time variables for other purposes, cotemporality can be achieved by the use of a free time variable exactly in the same fashion as in pronoun reference.

As Parsons (1973) and Stalnaker (1973) argue in reply to Partee (1973), the use of variables is not conceptually a new device in Priorian tense logic, since they are used in the meta-language. As far as the anaphoric use of tenses is concerned, it can adequately be treated by introducing an explicit control on the semantic interpretation, which is exactly what the EIL system achieves by virtue of UDs. Thus, in our system, the syntax and the semantics for the tense operators are basically Priorian. What is new to Priorian tense logic is that the semantic interpretation is closely knit with UDs, which are essentially of pragmatic nature and independently motivated by other, but related, phenomena -- namely, definite descriptions and pronoun reference[8]. What Partee (1973) argues as a "striking parallel" between pronouns and tenses, then, occurs at the semantico-pragmatic level, not necessarily at the syntactic level of the logic, or English for that matter.

As has been mentioned in footnotes [6] and [7], there are other, and rather pressing, reasons to introduce time variables at the syntactic level. In fact, they seem to be all the more necessary where the parallel with pronouns breaks down. Note that the time reference is implicit in English (it surfaces only as the tense marking of the verb) if there is no time adverbial, as we have seen in (4.16b). Moreover, the tense seems to be redundant if there is one, as we have seen in (4.15). Thus, the phenomena concerning tenses and time adverbials are more complicated than those concerning pronouns and noun phrases. However, one of the interesting trends in recent sophisticated treatments of tenses (e.g., Dowty 1980, Cooper ms) is that they allow the possibility of free variables, either at the syntactic level (Cooper ms), or at the semantic level as one of the time indices (Dowty 1980)[9]. In this respect, a pragmatic component to actually identify (the referents of) these free variables, either in the object language or in the meta-language, is necessary to complete the system. Although the EIL system is rather at the

[7] There are other, and perhaps more important, motivations for an explicit time variable than cotemporality, particularly the interaction of tense operators among themselves and with time adverbials. I will discuss only the points relevant to the present discussion of the discourse structure here. Also I have ignored the fact that Dowty (1979, 1980) and others, following the proposal by Bennett and Partee (1972), have extended the semantics for tense by making intervals of time as primitive, rather than moments of time as in PTQ.

[8] We will later see another, quite independent, motivation for UDs, namely, the treatment of non-denotational aspects of "meaning." See Chapter 6.
"primitive" level of PTQ concerning the treatment of tense, its pragmatic capacity will be, I believe, sufficiently powerful to accommodate these more refined tense systems.

Essentially the same mechanism works for modal operators, though which linguistic expressions will create a possible world in the meta-language may not be so straightforward as the case of tenses. Possibly and necessarily work exactly like tenses, and modal verbs and auxiliaries are expected to behave similarly, though I have nothing more to say here.

[9] In Cooper's system, tense morphemes are free variables which may be bound by time adverbials. If they remain unbound, they specify only the relative relation to the speech time, and are interpreted pragmatically. In Dowty's system, the semantics uses two indices for time: "speech time" and "reference time," modifying Reichenbach (1947)'s classical distinction among these two plus "event time." The second index -- the reference time -- is used pragmatically in analyzing discourse.
5. SAID, IMPLICATED, AND PRESUPPOSED

5.1. GRICE'S PROGRAM

In the unpublished William James Lectures delivered at Harvard University in 1967, H. Paul Grice opened an entirely new perspective in the formal analysis of natural language[1]. By coining new words -- to implicate, implicatum/implicata, and implicature -- and making distinction between 'what is said' and 'what is implicated,' he enabled researchers of natural language to cope with apparently chaotic aspects of language use, which typically emerge in conversation, without abandoning a formal analysis. That is, 'what is said' -- namely, what is sometimes called by different people (with some amount of shift of meaning) as 'logical form,' 'propositional content,' or, roughly speaking, what can be expressed by formal logic -- is not all of what human communicators mean/understand when they use/encounter natural language utterances. Classical examples include logical connectives. For example, English or is often claimed to be exclusive, not the standard inclusive disjunction symbolized by 'v' or the like. Similarly, English and is sometimes claimed to be asymmetric, since the temporal order matters, as can be seen in (5.1).

(5.1) a. Jack took the pill and got sick.
    b. Jack got sick and took the pill.

Also, English if...then is claimed to be not material implication; English not is claimed to be ambiguous between "internal negation" and "external negation" (cf. Section 2.3), etc. One approach to tackle these phenomena would be to define new, nonstandard logical connectives whose semantic interpretations are exactly what their natural language counterparts mean. This achieves the desired result only at the cost of the virtues of symbolic logic (or any language based on that, including analysis trees used in PTQ), which, despite its relative simplicity has proved to be successful in explaining diverse range of complicated phenomena -- scope ambiguity, entailment relationship, and contradiction, just to name a few. Moreover, we have no guarantee that these additional connectives will allow formal treatment in the same sense as traditional logical connectives have

[1] Parts of his lectures have been published as Grice (1975) (Lecture II) and Grice (1978) (Lecture III). My page reference to the former paper is that of The Logic of Grammar.
been treated. Another approach would be not to change the existing inventory of logical apparatus but to complicate the relationship between natural language and logical language; in other words, given a surface structure, a more abstract underlying structure enriched with a variety of abstract predicates is posited (cf. Morgan (1969) for such a proposal as to the treatment of "presupposition"). This has an obvious defect of arbitrariness. It is not known a priori how the two structures should be related. Moreover, the semantic conditions of those abstract predicates are often left to our intuition without any assurance that they can be given formally. If you put too much mechanism in the underlying structure, you will have to, often arbitrarily, "delete" elements which don't surface, as in the generative semantic approach, where the semantic interpretation is based on the underlying structure since it is the semantic representation. The transformational devices generative semanticists used, such as "transderivational constraints," are often so powerful that it could be used to do almost anything. The situation wouldn't change much if "semantic interpretation rules" (in the sense of interpretive semantics) were to map surface structure to too much enriched 'logical form.' As has emerged in the past decade, it would do more harm than good if one relies too heavily on transformations (cf. Peters and Ritchie 1973, Gazdar to appear).

The breakthrough achieved by Grice's program is the separation of what is 'communicated' but not 'said' from the traditional formal device whose motivation has been primarily to express what is 'said'[2]. This approach is reminiscent of the one taken in natural sciences in which 'ideal gas,' 'vacuum,' or other idealized situations are analyzed first and other factors are later introduced as perturbations. As a perturbative device, independent of traditional formal logical apparatus, Grice posits a principle which govern the human, natural conversation, namely, the Cooperative Principle (CP), which is quite general and seemingly reasonable: "Make your conversational contribution such as is required, at the stage at which it occurs by the accepted purpose or direction of the talk exchange in which you are engaged" (Grice 1975, p.67). That is, CP dictates the participants in the communication to make every effort toward a smooth transfer of information, whether it is a direct transfer or an indirect one. There are four maxims, as corollaries of CP, that Grice mentions: they concern (1) the quantity of the information transferred, (2) its quality, (3) its relevance (relation), and (4) the manner of the transfer. I will cite Grice's specification of the maxims below and briefly comment them in connection with the EIL

[2] I use the term to 'communicate' as an inclusive term to cover both to 'say' and to 'implicate,' that is, what is communicated is any information which is transferred from the utterer to the hearer by way of uttering the utterance whether it is directly said or indirectly implicated, suggested, meant, or whatever. I ignore the so-called non-verbal communication here.
system afterwards

(5.2) Maxim of Quantity
a. "Make your contribution as informative as is required (for the current purposes of the exchange)."
b. "Do not make your contribution more informative than is required."

(5.3) Maxim of Quality: "Try to make your contribution one that is true."
  a. "Do not say what you believe to be false."
  b. "Do not say that for which you lack adequate evidence."

(5.4) Maxim of Relation: "Be relevant."

(5.5) Maxim of Manner: "Be perspicuous."
  a. "Avoid obscurity of expression."
  b. "Avoid ambiguity."
  c. "Be brief (avoid unnecessary prolixity)"
  d. "Be orderly." (Grice 1975, p.67)

First of all, the above is not a closed and complete list. As Grice himself admits, other submaxims may be possible under the title of the maxim of Manner; likewise, (5.2b) might be reclassified as a submaxim of the maxim of Relation, since too much information is misleading and hence irrelevant. Thus, these particular four maxims and their particular formalizations might be understood as some of the possible implementations of CP, allowing some further additions and modifications.

Secondly, the list is not homogeneous in the sense that the first three maxims are about the information transferred itself ('what is said') but the last maxim is about the way of the transfer ('how what is said to be said'). It may be argued that the maxim of Manner should be separated from the other three and treated in a wider perspective where cultural and sociological aspects of language use get proper treatment. Accordingly, this maxim is not utilized much in the EIL system; maxims mostly utilized are restricted to those concerning the information itself.

Third, these maxims seem to depend on an unspecified concept of "rationality" of the utterer and the hearer. We know that the concept of "rationality" varies rather considerably depending on the culture. Moreover, it is important to remember that even languages considered "rational" often function quite "irrationally" when the users are "irrational." Thus, we have this much of vagueness of the concept of "rationality." CP, then, is best to be taken as representing only certain "rational" communicators at a certain "rational" mental state, with a certain meaning of "rationality," which is perhaps highly dependent on the culture.
Fourth, as has briefly been mentioned in Chapter 4, and as Grice himself admits, the maxim of Quality is distinguished in the sense that "other maxims only come into operation on the assumption that this maxim of Quality is satisfied" (Grice 1975, p.67). Since other philosophers like Lewis (1975) take a similar principle as a convention rather than a conversational principle, it has been conventionalized in the EIL system, as we have seen in the operation of the super-interpreter, at the cost of some interesting conversational phenomena such as lies, metaphors, ironies, etc.

Fifth, the maxim of Relation is notoriously vague and hence it has turned out to be extremely powerful in explaining "conversational implicature." This maxim may be taken not as a formulation, but as a guideline for formulation; it deserves the place of a meta-maxim. One of the examples based on this maxim is, I believe, the organization of the universes of discourse in the EIL system, which we have seen in detail in Chapter 4; the most recently registered information is most relevant, hence it is considered first. Perhaps other places to utilize this guideline may be found in a mechanical conversational system, though it would be rather hard to discern any direct influence of such a guideline since every component of a consistent system tends to be integrated in a relevant way.

Finally, we have to be cautious to admit that we may still be at a level of, possibly arbitrary, idealization even if CP is at work -- an idealized conversational system. To be sure, CP is effective in analyzing the "real" communication system -- much more effective than the analysis which relies only on the too much idealized logical form, but we may only have advanced to a higher degree of approximation and not have reached the ultimatum. Thus, we should be prepared to accept a gap between the "real" conversation and the "idealized" one in which CP and the above four maxims are effective[3].

The remaining maxim concerning the information itself, the maxim of Quantity, is, however, seems to be more suitable for mechanical interpretation. All we need is some kind of quantitative scale to measure the "informative"ness of a piece of information, and we have at least one such formalism, namely that by Shannon (Shannon and Weaver 1949). Roughly speaking, the amount of information of an event is defined as a negative entropy -- that is, it is inversely proportional to the logarithm of the probability of the occurrence of the event. The logarithm is used to insures the correct result if two or more pieces of information are added and hence does not matter if we only compare relative informative. Since an event which has smaller probability of occurrence is less expected, the maxim of

[3] This does not necessarily mean that the maxims are always obeyed, since there are cases of "clash" among the maxims. These cases are handled on the assumption that CP is not violated, as we will see later how "conversational implicatures" arise.
Quantity can be understood to require more unexpected information to be supplied\[4\]. In this way of understanding, the maxim of Quantity requires the utterer to provide new information -- information which is not already known, i.e., not already put in the universes of discourse, or at least not recently activated, i.e., put in a lower UDi (with smaller index \(i\)). That is, a piece of information whose value is \(U\) in UDi for \(i > 0\), or at least for a large \(i\), is preferred than the one whose value is some definite value depending on the type of the information. This functional interpretation of the maxim has been taken into account for the operation of the super-interpreter.

There are numerous series of "subscales" which can be defined by the concept of the probability of occurrence of events. For example, a sentence with all is more informative than the counterpart with some since the probability of all's having some property is smaller than that of some's having the same property. Similar arguments hold for pairs like necessarily and possibly, or must and may. Since the absolute value of the amount of information cannot be determined without determining a constant term, it may not be possible to compare all kinds of information on a single scale unless some way of absolutely determining those constants is known. Thus, without knowing the values of the constant terms, we can only partially order things which we have good reason to believe to have the same constant terms, like the ones mentioned above. With this qualification, the maxim of Quantity seems to have a well-formed foundation in a mechanical system and we will see how they are implemented later in Chapter 6.

As for detailed discussions and more examples of the usage of Grice's maxims, from somewhat different points of view, see Horn (1973), Kempson (1975), Harnish (1976), and Gazdar (1979a), among others.

5.2. TAXONOMY OF IMPLICATURES

As we have seen in the previous section, Grice classifies the communicated information into two classes: 'what is said' and 'what is implicated.' In the following, I will use binary features to classify various types of information. For the above dichotomy, the feature \([\text{said}]\) is used, specifying whether the information is 'said' \([+\text{said}]\) or 'implicated' \([-\text{said}]\)[5]. Based on CP and the maxims, the communicated information can also be classified into two classes: 'what is conversationally communicated' exploiting CP and the maxims

\[4\] Lyons (1968) states a similar observation, apparently independently of Grice's maxims: "... it is certainly in accord with general, everyday usage to say that the 'meaningfulness' of utterances, and parts of utterances, varies in inverse proportion to their degree of 'expectancy' in context" (p 415).
and 'what is conventionally communicated' depending on the particular lexical items whose communicative power is conventionally fixed. Introducing another feature [conventional], the former is specified as [-conventional] and the latter as [+conventional]. Note that, as Grice himself mentions, this dichotomy is not the same as the first one using the feature [said], since we can 'implicate' something conventionally without resorting to any conversational principle. That is, [-said] can cooccur with [+conventional]. Thus, the features [said] and [conventional] are independent. However, since one cannot 'say' anything unconventionally, all the [+said] information is conventional, that is, we have a redundancy rule (or an implicational law) (5.6):

\[(5.6) [+\text{said}] \rightarrow [+\text{conventional}],\]

These two features and the redundancy rule (5.6) give us the following three classes of information: [+said,+conventional], [-said,+conventional], and [-said,-conventional]. (Note that [+said,-conventional] is impossible by (5.6).) Let us consider them one by one.

First, [+said,+conventional]. This is what we have simply called 'what is said' and I will identify it with 'denotation' or 'semantic condition' in the following. This is the major concern for semantics and a relatively light stress is laid on it in this dissertation. The remaining two belong to pragmatics.

The second one, namely, [-said,+conventional], is part of what has been called 'what is implicated.' Grice (1975) calls it a 'conventional implicature.' Note that Grice's CP and maxims have nothing to do with this class of implicature, since by definition, it is conventional and not based on any conversational principle. Rather, "the conventional meaning of the words used will determine what is implicated, besides helping to determine what is said" (Grice 1975, p.66). As a typical example carrying a particular 'conventional implicature,' Grice mentions therefore: by uttering "A therefore B" the utterer commits himself that B is "a consequence of (follows from) A" (ibid, p.66). Another typical example is but (Grice 1978) with some kind of contrast, unexpectedness, etc. Since the utterer commits himself to the truth of the implicature, the 'conventional implicature' is not contradictable (Kempson 1975). Kempson's sentence (ibid, p.145) based on Grice (1975, p.66)'s is impossible.

\[(5.7) \text{John is an Englishman; he is therefore brave} \quad \text{--- though I don't believe there's any connection between the two.}\]

[5] This feature, and others introduced later are used solely for the classificational convenience. They don't by any means correspond to "primitives" from which a piece of information is constructed.
I will return to the discussion of contradictability (or cancelability) of implicatures shortly, which will turn out to be an important criterion to classify implicatures.

The third class is specified by [-said,-conventional]. This is what has been called 'conversational implicature' (Grice 1975). For example, take the exclusive interpretation of or, which has been mentioned briefly at the beginning of Section 5.1. This can be explained by the use of the maxim of Quantity. If there are two events A and B, the probability of the occurrence of "both A and B" is smaller than that of the occurrence of "either A or B". Thus, "A and B" is more informative than "A or B," since the amount of information is inversely proportional to the probability of the occurrence of the event. If we assume that the utterer of "A or B" is obeying the maxim of Quantity, s/he should utter "A and B" if both A and B were the case, even though truth-conditionally "A or B" is not false for this case. Thus, the hearer concludes that it is not the case that "A and B," hence either A alone or B alone. and not both, is the case. Note that if the utterer knew which of the two is the case, s/he should utter "A" or "B," depending on which is true, since the probability of the occurrence of A alone, or that of the occurrence of B alone, is smaller than that of the occurrence of "A or B" and hence A alone, or B alone, is more informative than "A or B." Similar arguments can be constructed to produce the implicatures such as "not all" for some, "not necessarily" for possibly, etc. based on the subscales discussed in Section 5.1. These are examples of what Gazdar (1979a) calls "scalar quantity implicatures." Another class of quantity implicatures is what Gazdar (1979a) calls "clausal quantity implicatures," e.g., "not A" and "not B" from "if A then B," or "not A" and "not B" from "A or B," which can be argued quite similarly based on the maxim of Quantity; if the sentence A or B is known to be true, then there is no need to use if, or or, which "weakens" the statement and hence increases the probability of occurrence of the event corresponding to it.[6]. Since this kind of implicature is constructed based on CP and the maxims, "the presence of a conversational implicature must be capable of being worked out; for even if it can in fact be intuitively grasped, unless the intuition is replaceable by an argument, the implicature (if present at all) will not count as a conversational implicature; it will be a conventional implicature" (Grice 1975, pp.69-70).

[6] Cf. Gazdar (1979a, p.58ff). To be precise, in Gazdar’s system, "scalar quantity implicatures" are preceded by the operator intuitively meaning "the speaker knows that." Also, "clausal quantity implicatures" are preceded by the operator intuitively meaning "it is possible, for all the speaker knows, that." The difference arising from these operators will play a major role in his system. Cf. Section 7.4.
Another important characteristic of the 'conversational implicature,' in addition to its "arguability" (capability of being worked out), is its cancelability, or contradictability. Compare (5.8) below with (5.7), where we saw that a 'conventional implicature' is not cancelable.

(5.8) Jack hopes to become either a governor or a senator -- in fact, he hopes to serve both offices during the next two decades.

Although somewhat misleading, the utterer of (5.8) would not be questioned of his credibility, unlike the case of (5.7). 'Conversational implicatures' are cancelable since "a participant in a talk exchange may fail to fulfill a maxim in various ways" (Grice 1975, p.69). There are various reasons why an utterer cannot, or does not want to, observe a maxim. In (5.8), the utterer may simply not have remembered Jack's hope when s/he uttered the first part ("violation"). The maxim of quantity would easily be "opted out" under the threat of life or other conditions ("I cannot say more, my lips are sealed"). If two maxims "clash," s/he has to choose one of them. Or a maxim may even be "exploited" by being "flouted" to convey a highly delicate implicature. (The words in the quotes above are all from Grice 1975, p.69.)

Remember that 'conventional implicatures' are not contradictable, or cancelable, while, by definition, all 'conversational implicatures' are cancelable. Thus, conversationality implies cancelability. If it is also the case that conventionality implies noncancelability, then we are done with the taxonomy of 'what is implicated,' since cancelability is equivalent to conversationality (or nonconventionality) and doesn't add a new dimension to the classification. This is exactly what non-presuppositionalists have apparently concluded in trying to abolish what have traditionally been called 'presuppositions.' "Presuppositions" have been re-analyzed and re-labeled as either entailments of the 'denotations,' 'conversational implicatures,' or 'conventional implicatures'[7]. However, there is at least one phenomenon which seems both conventional and cancelable, namely, the truth of the complement sentences of factive verbs.

[7] See Wilson (1975), Kempson (1975), and Boër and Lycan (1976), among others, for arguments by non-presuppositionalists. Karttunen and Peters have managed to metamorphose from presuppositionalists to non-presuppositionalists during the past decade, while maintaining their overall framework. Cf. Karttunen (1971a, 1971b, 1973, 1974, 1975), Peters (1977), and Karttunen and Peters (1975, 1976, 1977, 1979). I will discuss Karttunen and Peters (1979) later in Chapter 7, since their work, along with Gazdar (1979a)'s, is one of the few well-defined attempts for a formal treatment of 'what is implicated,' which can be compared with the current analysis on more or less equal grounds.
(Kiparsky and Kiparsky 1970). Observe (5.9).

\[\begin{align*}
\text{(5.9) a.} & \text{ Jack regrets being bald.} \\
\text{b.} & \text{ Jack doesn't regret being bald.}
\end{align*}\]

Both (5.9a) and (5.9b) 'communicate' that Jack is bald. Note that, since they are negations of each other, it is impossible that both (5.9a) and (5.9b) 'say' that Jack is bald, unless a very complicated analysis of negation is adopted. Thus, Jack's baldness is 'what is implicated.' Moreover, this implicature is conventionally associated with the verb regret. In fact, this kind of implicature is associated with the whole class of factive verbs including know, ignore, etc., and it doesn't seem to be possible to invoke a conversational maxim to explain the factivity, unless you resort to a fairly generous use of the maxim of Relevance. Thus, this information has at least the set of features [-said, +conventional] and apparently cannot be distinguished from the 'conventional implicature.' However, it behaves differently as to the cancelability:

\[\begin{align*}
\text{(5.10) a.} & \text{ *Jack regrets being bald -- though he isn't bald.} \\
\text{b.} & \text{ Jack doesn't regret being bald -- in fact, he isn't bald.}
\end{align*}\]

(5.10a) is impossible in the same way as (5.7) is impossible, thus the information conveyed by the complement behaves as if it were a 'conventional implicature.' This and similar phenomena have apparently led some authors, e.g., Karttunen and Peters (1979), to conclude that there is no "presupposition" associated with factive verbs; what is implicated is a 'conventional implicature.' (5.10b), however, is acceptable in the same way as (5.8) is acceptable. Although (5.10b) is somewhat misleading, this is exactly how (5.8) is. Thus, the information conveyed by the complement of a factive verb within the scope of a negation operator is as cancelable as a 'conversational implicature' is[8].

This suggests us to posit yet another criterion in the taxonomy of 'what is implicated,' namely, [-cancelable]. [-said, +conventional, -cancelable] is Grice's 'conventional implicature'; [-said, +conventional, +cancelable] is, partly due to the lack of a better term, the 'presupposition' reincarnated in a restricted sense[9]. By the nature of 'what is said,' and 'conversational implicatures,' we have the following redundancy rules.

\[\text{(5.11) [+said] \rightarrow [-cancelable]}\]

[8] Karttunen and Peters (1979) try to avoid the uncomfortable situation where they have to claim that "conventional implicatures" are cancelable by positing a different kind of negation from the ordinary negation for such sentences as (5.10b). See Section 7.3 for a discussion of their system.
Thus, the three binary features and the three redundancy rules (5.6), (5.11), and (5.12) give us the following four classes of 'what is communicated.'

\[(5.12) \quad [-\text{conventional}] \rightarrow [+\text{cancelable}]\]

Thus, the three binary features and the three redundancy rules (5.6), (5.11), and (5.12) give us the following four classes of 'what is communicated.'

\[(5.13) \quad a. \quad [+\text{said},+\text{conventional},-\text{cancelable}] \quad \rightarrow \quad \text{‘denotation’}\]
\[b. \quad [-\text{said},+\text{conventional},+\text{cancelable}] \quad \rightarrow \quad \text{‘presupposition’}\]
\[c. \quad [-\text{said},+\text{conventional},-\text{cancelable}] \quad \rightarrow \quad \text{‘conventional implicature’}\]
\[d. \quad [-\text{said},-\text{conventional},+\text{cancelable}] \quad \rightarrow \quad \text{‘conversational implicature’}\]

Thus, 'presuppositions' and 'conversational implicatures' share \([-\text{said},+\text{cancelable}]\) and differ only in the feature \([\text{conventional}]\); 'denotations' and 'conventional implicatures' share \([+\text{conventional},-\text{cancelable}]\) and differ only in the feature \([\text{said}]\); and 'presuppositions' and 'conventional implicatures' share \([-\text{said},+\text{conventional}]\) and differ only in the feature \([\text{cancelable}]\).

Grice further divides 'conversational implicatures' into a generalized and a particularized ones. The former is arguable even "in the absence of special circumstances" based on "a certain form of words in an utterance" (Grice 1975, p.73). The exclusive interpretation of or is such an example since the argument doesn't depend on any special context. The latter crucially depends on the particular context as Grice's example (5.14) below shows. As Grice says, in the context of (5.14a), (5.14b) "implicates that Smith has,

\[9\] In general, I agree with most of the arguments by non-presuppositionalists. Especially I don't call noncancelable information a 'presupposition'; it is either an entailment of the 'denotation' or a 'conventional implicature.' I will use the term 'presupposition' in the restricted and technical sense specified by the feature matrix shown above, and use only factive verbs as typical examples in the following, though I am willing to include other cases which will produce \([-\text{said},+\text{conventional},+\text{cancelable}]\) information. Traditional "presuppositions" will be briefly reviewed in Section 5.3. Note that my use of the word implicature (without any kind of quotes) is equivalent to \([-\text{said}]\) information, or 'what is implicated,' which includes 'presupposition.' When I use the word implicature in a narrower, Gricean sense, I will mark it by double quotes. Similarly, double quotes for other terms indicate that they are used in a traditional, and somewhat vague, sense. Single quotes are used for technical terms defined in (5.13) below.
or may have, a girl friend in New York" (ibid, p.70):

(5.14) a. Smith doesn't seem to have a girl friend these days.
   b. He has been paying a lot of visits to New York lately.

The implicature of (5.14b) would be completely different in the context of (5.14c):

(5.14) c. Smith doesn't seem to be eager to find a job in a big city.

To make the distinction between two kinds of 'conversational implicature,' we might need yet another feature, say [contextual]. However, since 'particularized conversational implicatures' don't seem to fit an easy mechanical treatment[10], I will completely ignore it in this dissertation, concentrating on 'generalized conversational implicatures.'

The distinction between 'conversational implicature' and 'conventional implicature' may sometimes seem to be too subtle to make. As Grice admits, "it may not be impossible for what starts life, so to speak, as a conversational implicature to become conventionalized" (Grice 1975, p.74) on the condition that special justification is supplied[11]. In fact, he gives the warning that "it is all too easy to treat a generalized conversational implicature as if it were a conventional implicature." In the EIL system based on the feature classification (5.13), these two kinds of implicatures are different at the values of two features: [conventional] and [cancelable]. Since the cancelability of information is one of the most important features of information in the EIL system, the distinction of the two kinds of information will still be clear even in the case that a 'conversational implicature' is highly conventionalized and treated as if it were associated with a particular lexical item (at the level of EIL translation), as the case of exclusive interpretation of or shows. Thus, with Grice's caution in mind, the mechanisms to invoke 'generalized conversational implicatures' will be implemented as particular procedures attached to particular lexical items in the EIL translation as long as it seems possible. In other words the argument to establish a 'conversational implicature' is treated as a ready-made packaged procedure and it is

[10] At least they will face the similar difficulty as in mechanical theorem proving, i.e., explosion of search during the process of trying to establish arguments for the implicatures. What is worse with 'particularized conversational implicatures' is that the maxims are too informally characterized to play the role of axioms in a theorem prover.

[11] See Sadock (1972), where he uses the phrase "speech act idioms" instead of conventionalized 'conversational implicatures,' for such examples.
evoked whenever the EIL lexical item responsible for the implicature is evaluated. See Section 6.1 for more arguments as to which level of the language interpretation system is most appropriate with which to associate implicature. We will see the details of the implementation in Chapter 6.

Sadock (1978) examines the criteria Grice lists (Grice 1975, pp.74-75) for the 'conversational implicature' (cancelability, nondetachability, etc., in addition to arguability) and argues that no single feature by itself is a sufficient condition; we know that the cancelability is a necessary, but not a sufficient, condition for the 'conversational implicature' since 'presuppositions' are also cancelable. Sadock even gives certain amount of doubts to the necessity of cancelability. As Grice's remark above suggests, the degree of conventionalizedness will work toward blocking the cancellation, and hence it seems very difficult to distinguish highly conventionalized 'conversational implicatures' from genuine 'conventional implicatures.' Note particularly that arguability is not sufficient to establish conversationality, since it is all too easy to build a plausible argument based on the liberal use of the maxims. So, one has to be careful in claiming that some kind of information is a 'conversational implicature,' not a 'conventional implicature,' or anything else. Temptation to invoke Grice's maxims is sometimes too strong to resist. See Harnish (1976) and Gazdar (1979a) for further discussions of implicatures.

5.3. ASPECTS OF PRESUPPOSITION

In Section 5.2, the notion of 'presupposition' was defined within the context of implicatures; it is one class of 'what is implicated'[12]. Originally, however, "presuppositions" have a much older history than implicatures, which have only recently (circa 1967) been defined by H.P. Grice. For example, Frege (1892) argues, in connection with sense and reference, that in the assertion "Kepler starb im Elend" ("Kepler died in misery"), it is "presupposed" ("vorausgesetzt") that the reference of the name Kepler exists, but that the sense of the sentence doesn't contain that thought. He also notes that the negation of the sentence: "Kepler starb nicht im Elend" ("Kepler did not die in misery") equally "presupposes" the existence of the reference of Kepler. In this century, Strawson (1950) advocates the same view concerning the treatment of definite descriptions, which Russell (1905) has argued entail the existence of the referents. According to Strawson, the existence is "implied"

[12] As has been mentioned in footnote [9], 'presupposition' (in single quotes) refers to the notion defined in Section 5.2, namely, information with features [-said,+conventional,+cancelable]. The term "presupposition" (in double quotes) refers to more traditional and broader use of the notion.
rather than entailed, and a sentence containing a definite description is neither true nor false if the referent of the definite description doesn’t exist, instead of being false as Russell’s analysis predicts. In 1960s, many linguists have adopted the concept of "presupposition" at least as a descriptive device and applied it to a variety of linguistic phenomena (e.g., Horn 1969, Morgan 1969, also see Keenan 1971, and Boër and Lycan 1976 for extensive lists of phenomena).

One of the common properties of the diverse phenomena labeled as "presuppositions" can be summarized in the following scheme: if a sentence S "presupposes" a sentence S’, then so does not S[13]. This observation has led to a definition, in fact, a class of definitions, of "presupposition," which is now called a semantic (or logical) definition (cf. for example, Keenan 1971). Ignoring minor differences, a semantic definition of "presupposition" looks like (5.15):

\[(5.15) \text{ A sentence S "presupposes" a sentence S' iff} \]
\[a. \text{ S entails S', and} \]
\[b. \text{ not } S \text{ entails } S'. \]

It is not difficult to notice that definition (5.15) is not compatible with the bivalence of the truth-value of the sentence. From (5.15a) we have that not S’ entails not S, and from (5.15b) we have that not S’ entails not not S, which is equivalent to S if we assume bivalence. Since not S’ entails both S and not S, not S’ must always be false, i.e., S’ is always true. This is problematic unless one takes an extreme, and uninteresting, position that all "presuppositions" are tautologies. One way to avoid this difficulty would be to allow S to take the third truth-value if S’ is false, and reformulate the definition as follows.

\[(5.16) \text{ A sentence S "presupposes" a sentence S' iff} \]
\[\text{ if } S' \text{ is false, then } S \text{ is neither true nor false.} \]

The use of a three-valued logic is controversial since there can be indefiniteness as to the assignment of truth-values for standard logical constants, most notorious of which is negation[14]. Another difficulty of the semantic definition of "presupposition" is that it cannot explain the cases where a "presupposition" is not created due to the existence of another utterance or a particular context. Note that there is no "presupposition" that Jack is bald in (5.17a), (5.17b), or (5.17c) as a whole.

[13] \text{not } S \text{ is a shorthand notation for the negation of } S.

[14] \text{It has to be emphasized again that the logical constants of EIL is not based on a three-valued logic. The motivation of U is quite different as we have seen in Chapter 4.}
(5.17) a. Jack doesn’t regret being bald. For he isn’t bald.
b. Jack isn’t bald. So, he doesn’t regret being bald.
c. Jack doesn’t regret being bald, even though he is bald at the age of 20.

In (5.17a), the second sentence cancels the "presupposition" created by the first sentence while, in (5.17b), the first sentence establishes the context which will prevent the "presupposition" of the second sentence from coming about, which I will call an abortion of a "presupposition" to distinguish it from a cancellation as in (5.17a). Note that 'cancellation' always works backward, i.e. the later context abolishes the 'presupposition,' while 'abortion' always works forward. As for (5.17c), the "presupposition" produced by the first half of the sentence is no longer a "presupposition" by the time the second half is understood, since it entails, rather than "presupposes," that Jack is bald. This is a case which I call the consolidation of a "presupposition." Note that the entailment analysis can only be consistent with (5.17c) (since Jack’s baldness could be said to be both "presupposed" and entailed). For the first two cases, the concept of entailment is problematic, since if a "presupposition" is a kind of entailment, as (5.15) says, there is no possibility of cancellation or abortion. Moreover, it is not the case that the first sentence of (5.17a), the second sentence of (5.17b), or the first half of (5.17c) has the third truth-value, as (5.16) says; they are straightforwardly true. Since cancelability is the feature to distinguish 'presuppositions' from 'conventional implicatures,' we cannot adopt a semantic definition, which will blur this important distinction. See Wilson (1975), Kempson (1975), Boër and Lycan (1976), and Gazdar (1979a), among others for further arguments against semantic definitions of "presupposition."

Another way to avoid the difficulty of (5.15) would be to use a weaker relation than entailment in the definition. Remember that A entails B iff B is true in every model where A is true. To weaken this, one could use a truth in a particular model instead of every model. That is, a modified definition would be something like (5.18):

(5.18) A sentence S "presupposes" a sentence S' iff
a. if S is true in the current model, then S' is true in the current model, and
b. if S is false in the current model, then S' is true in the current model.

Here, "the current model" is the model on which the semantic interpretation is based, including the actual world. Now that the truth is a contingent one, S' doesn't have to be a tautology; it only has to be true in the current model. Thus, this definition is compatible with bivalency. However, in addition to the fact that the cancellation and abortion problem with (5.17) still remains, it has an undesirable consequence that everything that happens to be true in the
current model is a "presupposition," since the truth in the current model is the necessary and sufficient condition of a "presupposition." Note that, in Chapter 4, we have used hierarchically ordered partial models as the context to interpret discourse referents. This kind of view of the model as the context, together with the direction suggested in (5.18), eventually leads to so-called pragmatic definitions of "presupposition" (cf. Keenan 1971, Karttunen 1973, 1974, Stalnaker 1974), though they are notorious for relying on some vague and often undefined notions such as "appropriateness," "sincerity," "felicity," etc. (cf. Gazdar 1979a, p. 105). A prototypical definition looks like (5.19), using an unspecified notion of "appropriate":

(5.19) A sentence S "pr-supposes" a sentence S' iff it is "appropriate" to utter S only in the context where S' is true.

I don't list various elaborations in the line of (5.19) by linguists and philosophers[15], but only point out that, as Gazdar (1979a) argues, the cancellation (and the abortion) of "presuppositions" shown in (5.17) will still remain a mystery since both (5.17a) and (5.17b) are still "appropriate," however you define this term. Remember that 'conventional implicatures' and 'presuppositions' are only different in the feature [cancelable]. Definition (5.19) seems to be more appropriate to the former since it apparently doesn't allow cancellation. Moreover, this kind of definition cannot capture the dynamics of communication. For example, (5.20) below can be used to implicate that Jack is bald, if that fact is not yet known to the hearer, i.e., it is not yet in the context.

(5.20) Jack doesn't regret being bald.

In fact, (5.19) is sometimes counterintuitive. Consider the case where 'what is implicated' has already been 'said.' In this case, 'what is implicated' is already in the context and is not "presupposed," as can be seen in (5.21):

(5.21) Jack is bald. But he doesn't regret being bald.

[15] Cf. the works by the authors cited above. It should be emphasized that their elaborated definitions are far more complex and sophisticated than the "straw man" (5.19) and may circumvent the difficulties particular to the crude form (5.19). In particular, if it is the speaker rather than the sentence that "presupposes" something, as Stalnaker (1974) stresses, the "presupposition" and the context may be coupled more strongly and may produce more satisfactory results. As a technical term, the 'presupposition' in our system is something which is produced by (the utterance of) sentence in the interaction with the context.
Note that the second sentence doesn't literally "presuppose" that Jack is bald, since that fact is already "known" as a solid piece of information, not "supposed" in any way. In this case, the "presupposition" is discarded due to the existence of the solid fact which corresponds to the first sentence. Thus, pragmatic definitions like (5.19), which requires "presuppositions" to be already in the context, cannot give satisfactory explanation for this kind of phenomenon, which I will call absorption.

See Gazdar (1979a) for further discussion concerning the inadequacies of the previous pragmatic definitions of "presuppositions." In short, none of the previous definitions of "presupposition" or the verb to "presuppose" seems satisfactory[16].

In the EIL system, the 'presupposition' has been defined based on the feature system developed in Section 5.2. Using this concept, the verb to 'presuppose' is defined operationally, using the pragmatic devices developed so far in Chapters 2 and 4 and some more which will be introduced in Chapter 6, particularly their interaction with the context. A definition would look like (5.22) if it is stated in ordinary words.

(5.22) A sentence S 'presupposes' a sentence S' iff the processing of S produces the information which corresponds to S' and has the feature matrix [+said,+conventional,+cancelable], i.e., a 'presupposition' corresponding to S'.

We have seen what the feature matrix means in Section 5.2. The technical details of the processing of sentences which will produce a 'presupposition' will be described in Chapter 6, based on the concepts developed in Chapters 2 and 4 and some new ones. To briefly preview the essence of the mechanism, the processing of (5.20), for example, will create a piece of new information corresponding to the sentence "Jack is bald" in the top-level universe of discourse (UDn). This information has the feature [+cancelable]. Thus, in the case of cancellation, (5.17a), the processing of the second sentence deletes this new information. In the case of abortion, (5.17b), on the other hand, since the contrary information to the one that Jack is bald is already in UD's, the new information will be deleted from the top-level UD. Hence no 'pr-supposition' is created; it is aborted before being born. As for consolidation, (5.17c), the 'presupposition' created by the first half will be immediately deleted since the processing of the second half will create a solid, noncancelable information that Jack is bald, which is favored over a piece of cancelable information. In this way, noncancelable information consolidates cancelable information. Finally, as for absorption, (5.21), since the

[16] Later in Chapter 7, I will show that even Gazdar (1979)'s analysis is inadequate in treating the absorption case like (5.21).
information that Jack is bald already exists with the feature [-cancelable] ('denotation' of the first sentence), this information blocks the addition of the duplicate information with the feature [+cancelable], since it suffices to retain only the [-cancelable] information in UDs. Thus, the mechanism described in Chapter 6 can explain all the properties of "presupposition" observed so far -- being created (5.20), canceled (5.17a), consolidated (5.17c), aborted (5.17b), and absorbed (5.21). These aspects of "presuppositions" are summarized in (5.23):

<table>
<thead>
<tr>
<th></th>
<th>Existing Context</th>
<th>Later Context</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Abortion</td>
<td>-</td>
<td>N/A</td>
<td>(5.17b)</td>
</tr>
<tr>
<td>b. Absorption</td>
<td>+</td>
<td>N/A</td>
<td>(5.21)</td>
</tr>
<tr>
<td>c. Creation</td>
<td>0</td>
<td>N/A</td>
<td>(5.20)</td>
</tr>
<tr>
<td>d. Cancellation</td>
<td>N/A</td>
<td>-</td>
<td>(5.17a)</td>
</tr>
<tr>
<td>e. Consolidation</td>
<td>N/A</td>
<td>+</td>
<td>(5.17c)</td>
</tr>
</tbody>
</table>

where - shows that the context contains contradictory information, + shows that the context contains duplicate information, and 0 shows that the context is neutral. Note that (5.23a) - (5.23c) show the interaction between the newly produced information and the existing context. The new information will go through one of the three processes. Only in the case of creation, (5.23c), can the new information survive, i.e., not deleted, and remains in the form of cancelable information, which still has the chance of being deleted by one of the two later processes (5.23d) and (5.23e). A satisfactory theory of "presupposition" would have to be able to explain at least these five aspects of "presupposition"[17]. We will examine how each aspect is treated in the EIL system in the next chapter.

5.4. PSYCHOLOGICAL REALITY OF CANCELABILITY

Harris (1974) presents an interesting experimental psychological finding concerning the implicature. In his terminology, the "presupposition" of the sentence pattern "At T, A began doing X" is "Just before T, A was not doing X" and its "implication" is "Just after T, A was doing X." While almost all the subjects judged both the "presupposition" and the "implication" as necessary consequences of the affirmative sentence, the situation was drastically different when the sentence was negated, i.e., "At T, A didn't begin doing X." Remember that, according to the property of "presupposition" discussed

[17] Note that 'conversational implicatures' also show these aspects since they share cancelability with 'presuppositions,' cf. Section 6.6. We also will see the cases of the abortion, the absorption, and the creation of 'conventional implicatures' in Section 6.3. Note that 'conventional implicatures' are not cancelable and hence there is no case of cancellation or consolidation.
in Section 5.3, negation is not supposed to change the status of the "presupposition;" if the "presupposition" is true for the affirmative sentence, it is also true for the negative sentence. Thus, it is expected that most subject still consider the "presupposition" to be true even if the sentence is negated. However, the predominant judgement was 'indeterminate' rather than 'true.' That is, although almost no one judged the "presupposition" as 'false,' it was judged more as 'indeterminate' than as 'true.' Similarly, as for the "implication," although almost no one judged it as 'true,' it was judged more as 'indeterminate' than as 'false.' Moreover, there was no significant difference between the number of subjects who judged the "presupposition" as 'indeterminate' and that of those who judged the "implication" as 'indeterminate;' whether it is a "presupposition" or an "implication," it was equally 'indeterminate.' Thus, he concludes that "no such distinction as presupposition-implication exists in memory" (Harris 1974, p. 596).

This conclusion, however, is rather hasty, since the distinction do exist between the two; the "presupposition" was judged more as true than as false and the "implication" more as false than as true, if we consider only the determinate response for the negation. The indeterminate response seem to indicate the nonsolid, i.e., cancelable, status of the information conveyed as the "presupposition" or the "implication," though it is not clear whether Harris's "implication" corresponds to our (entailment of) 'denotation' or 'conventional implicature,' which is not cancelable, or 'conversational implicature,' which is cancelable. If we can take the indeterminacy as the sign of cancelability, the "implication" that "Just after T, A was not doing X" from "At T, A didn't begin doing X" is most likely a 'conversational implicature' of the sort which Geis and Zwicky (1971) call an "invited inference." Although Geis and Zwicky themselves claim that "invited inferences" are not 'conversational implicatures,' there is at least one attempt to devise a Gricean explanation for them based on the maxims of Quantity and Relevance, as Boër and Lycan (1973) has argued[18].

[18] As Gazdar (1979a) points out in citing Harris's result, the distinction between affirmative and negative sentences are significant. As will be seen in the next chapter, the mechanism based on the interaction among pragmatic procedures and the super-interpreter achieves this result by making a close correspondence between cancelability and negation.
6. PRECONDITIONS AND POSTCONDITIONS

6.1. REPRESENTATION AND INTERPRETATION

Before starting the detailed description of pragmatic mechanisms implemented in the EIL system, I would like to make a brief remark on the important distinction between the semantico-pragmatic representation and interpretation. Although the phrase structure grammar in the sense of Gazdar (to appear), on one of whose variants the parser and the translator of the EIL system is based, is basically a surface grammar, the level of the EIL translation is not necessarily equivalent to the surface structure. As has briefly been mentioned in Chapter 2, various morphological and syntactic operations to the English sentences are performed to get a translation in EIL. For example, if the surface structure is ambiguous, e.g., as to the scope of the quantifiers, then it is mapped to more than one EIL translations. If, on the other hand, two or more sentences are synonymous (within the limit of simplification), then the mapping can be many-to-one. (Though it does not necessarily have to be so, that is, different EIL translations can have the same semantico-pragmatic interpretation, see below.) In general, the translation on the whole is a many-to-many mapping. Thus, the two structures -- the surface structure and the EIL translation -- are distinct levels of representation. The EIL translation (ET) is a semantico-pragmatic representation where most of the syntactic idiosyncrasies particular to individual languages are filtered out. ET is close to a universal representation, though it is not completely universal because the lexicon of EIL is dependent on the language-particular property as to which lexical items carry information with the feature [+conventional], i.e. 'denotation,' 'conventional implicature,' and 'presupposition,' which should depend on each surface lexical item since they are conventional. Thus, ET is not so abstract as the purely "semantic" representations, such as some of the underlying structures proposed by generative semanticists[1].

ET serves as the input to the interpreter and the super-interpreter. The output is the semantico-pragmatic interpretation (SPI), which includes the denotations such as truth values, change of UDs as side effects such as adding discourse referents, and other actions such as user-inquiry, etc. Again, the relationship between ET and SPI is many-to-many. The interpretation
process takes the effect of the context (UDs) into account, thus the same ET may result in different SPIs. On the other hand, different ETs can have the same SPI in the model if the context so designates.

Thus, we have three levels of language processing in the system: the surface structure, ET as the semantico-pragmatic representation, and SPI as the interpretation of ET. The question is which is the appropriate level with which to associate mechanisms to produce 'what is implicated.' Gazdar (1979a, pp.56-57) argues that "conversational implicatures" should be associated with the level of the "semantic representation," not the surface structure, nor the deepest level of semantic interpretation (the level where truth values, etc. belong). On one hand, to associate them with the level of the surface structure would be equivalent to treating them as conventional, since they would then be determined solely depending on the surface lexical items. On the other hand, to associate them with the level of the semantic interpretation would blur the distinction between 'what is said' and 'what is implicated,' since the level of the semantic interpretation is the level for 'what is said.' I take a similar approach in the EIL system; ET is the level of semantico-pragmatic representation and each lexical item of EIL carrying an implicature is associated with additional pragmatic procedures which are responsible for producing implicatures (in the broader sense, i.e., [-said]). There is, however, a subtle difference in how 'conventional implicatures' and 'conversational implicatures' are treated. Since 'conventional implicatures' are detachable, i.e., they depend on particular surface lexical items, this much of distinction among surface lexical items is carried over to EIL lexical items. Thus, for example, BUT and AND are separate lexical items in EIL, though the mechanisms to produce the denotation are identical, since they have exactly the same semantic condition. The difference is in the pragmatic procedures; while the former has an additional pragmatic procedure to check the contrast of the two arguments, the latter has no such additional procedure. As for 'conversational implicatures,' they are not detachable, i.e., they depend solely on 'what is said' and not how 'what is said' is said.

[1] See Ross (1970) for the universal base hypothesis from a generative semanticist's point of view, within the context of the "performative analysis." His analysis can be understood as an attempt to let the underlying structure represent every information that both the interpreter and the super-interpreter need and hence eliminate the additional level of super-interpretation. Although interesting, the "performative analysis" is not compatible with a model-theoretic semantics, at least with the one adopted in the EIL system, since every indicative sentence automatically becomes true if we take the higher verb as something equivalent to say. To make the higher verb more abstract and give it an appropriate semantic condition is tantamount to adopting the level of super-interpretation. Cf. Kempson (1977), Gazdar (1979a), and Boër and Lycan (1980) for arguments concerning "performative analysis."
Thus, there can be no independent EIL lexical item, say XOR, which has exactly the same semantic condition as OR but has an additional pragmatic procedure to achieve the exclusive interpretation. Moreover, if different surface phrases are translated into EIL formulas with OR, they all 'conversationally implicate' exclusiveness. Once a surface sentence is translated into an EIL formula which contains OR, it has to implicate the exclusiveness, regardless of the origin of OR at surface. This exclusiveness is realized by an additional pragmatic procedure of OR, which produces a piece of cancelable information to deny one disjunct if the other is true. The cancelability of the information will explain the distinctive behavior of 'conversationally implicates.'

6.2. PRAGMATIC PROCEDURES

As we have seen in Chapter 3, each constant in EIL is associated with a denotation (DEN) procedure, which embodies the semantic condition of the constant as a procedure (a Lisp program). As has been discussed in Chapter 5, this is only part of 'what is communicated,' namely, 'what is said.' As for 'what is implicated,' we have seen that there are three types of implicatures ([=said]) in our system: a 'conventional implicature,' a 'presupposition,' and a 'conversational implicature,' of which only the last is nonconventional, and only the first is not cancelable. To take care of these implicatures, we use a new kind of procedure in addition to the DEN procedure - pragmatic procedures. We have two types of pragmatic procedures: a precondition (PREC) procedure and a postcondition (POSTC) procedure. The former is responsible for producing the first two kinds of implicatures, i.e., 'conventional implicature' and 'presupposition.' while the latter for producing the third, i.e., 'conversational implicature.' Corresponding to the feature system (5.13), we have the following correspondence between the information types and the procedures which produce them:

\begin{align*}
\text{(6.1) a.} & \quad [+\text{said},+\text{conventional}] \quad \text{DEN procedure} \\
\text{b.} & \quad [-\text{said},+\text{conventional}] \quad \text{PREC procedure} \\
\text{c.} & \quad [-\text{said},-\text{conventional}] \quad \text{POSTC procedure}
\end{align*}

By the redundancy rules, all the information produced by DEN procedures has the features [+said,+conventional,-cancelable], i.e., a 'denotation,' whereas the information produced by POSTC procedures has the features [-said,-conventional,+cancelable], i.e., a 'conversational implicature.' The information produced by PREC procedures can be either cancelable or noncancelable: [-said,+conventional,+cancelable] is a 'presupposition,' while [-said,+conventional,-cancelable] is a 'conventional implicature.' A marker is attached to each piece of information in the model to indicate its cancelability. This marker distinguishes the 'conventional implicature' from the 'presupposition.' Thus, a
'conventional implicature' is a noncancelable piece of information produced by a PREC procedure, a 'presupposition' is a cancelable piece of information produced by a PREC procedure, and a 'conversational implicature' is a (cancelable) piece of information produced by a POSTC procedure[2].

In the EIL system, an evaluation of a lexical item causes the execution of the PREC, DEN. and POSTC procedures attached to the lexical item. They are executed in this order. Thus, the execution of the PREC procedure of a lexical item precedes that of the DEN procedure of the same lexical item, and it in turn precedes that of the POSTC procedure of the lexical item. A 'conventional implicature' is produced by a PREC procedure, not a POSTC procedure, since if the context (UDs) is contradictory to the implicature, the whole interpretation process should be suspended to investigate the reason for the 'error' before the execution of the DEN procedure, which may result in a fatal situation. In an interactive conversational system, the utterer (i.e., the human being who typed the sentence on the keyboard) would be asked to supply further information from the keyboard. Thus, this kind of check of the consistency should be performed by the PREC procedure before the execution of the computation of the denotation. On the other hand, a 'conversational implicature' cannot be produced by a PREC procedure, since it is calculated depending on 'what is said,' i.e., depending on the result of the execution of the DEN procedure, as in the case of exclusive interpretation of OR (cf. Section 6.6). Moreover, a 'conversational implicature' need not cause the suspension of interpretation in the case that it contradicts the context, since it is simply aborted. Thus, it is the task of a POSTC procedure to produce a 'conversational implicature.' As for 'presuppositions,' since a factive verb both 'conventionally implicates' the complement sentence (if it is not in the scope of negation, etc.) and 'presupposes' it (if it is in the scope of negation, etc.), and since a mechanism attached to factive verbs should be unitary 'presuppositions' are also produced by the same PREC procedures which would produce 'conventional implicatures' in the affirmative context. Note that, although a 'presupposition' is produced by a PREC procedure, it never causes a suspension of interpretation, due to its cancelability; it is simply canceled, aborted, or absorbed, if it is not newly created[3].

[2] With regard to this correspondence, more informative names might be "conventional" procedures for PREC procedures and "conversational" procedures for POSTC procedures. However, since 'denotations' are also conventional, and [-said,+conventional] information includes 'presupposition' in our system, I have adopted less perspicuous names. Moreover, in addition to producing implicatures, these procedures perform other tasks concerning inheritance of implicatures, as will be seen later. The motivation of "pre-" and "post-" in the names of the pragmatic procedures will become clear shortly.
The major task of PREC procedures and POSTC procedures is to manipulate the context, i.e., UDs. In other words, their main purpose is to cause side-effects on the database during the process of evaluation. (Note that the database is the read-only memory for the DEN procedure; it cannot modify the content of the model.) While PREC, DEN, and POSTC procedures are executed in this order, only the result of the DEN procedure is returned as the value of the denotation of the constant to be used in recursively computing the denotation of the higher construction. The effects of PREC and POSTC procedures are carried over indirectly via the changes in UDs they modify, and do not directly participate in the compositional semantics. In other words, 'what is said' is compositional, while 'what is implicated' is essentially cumulative. We will see more on this point in Chapter 7. In this respect, (2.4f) should be revised as follows (cf. Appendix A).

\[
(6.2) \text{If } p \text{ is a member of } L < a_1, a_2, \ldots, a_m, b > \text{ and } q_1, q_2, \ldots, q_m \text{ are members of } L_{a_1}, L_{a_2}, \ldots, L_{a_m}, \text{ respectively, then}
\]

\[
V[(p \ q_1 \ q_2 \ \ldots \ q_m), w, t, f, n, i] =
\begin{align*}
\text{def} &= \{g = V[p, w, t, f, n, i]; \\
\text{if } &\text{ (g is defined)} \\
&\{d_1 = V[q_1, w, t, f, n, i]; \\
&d_2 = V[q_2, w, t, f, n, i]; \\
&\ldots \\
&d_m = V[q_m, w, t, f, n, i]; \\
&\text{prec}(p)(d_1, d_2, \ldots, d_m); \\
&\text{val} = g(d_1, d_2, \ldots, d_m); \\
&\text{postc}(p)(d_1, d_2, \ldots, d_m); \\
&\text{return (val)}
\}
\end{align*}
\]

As has been mentioned in Section 5.2, POSTC procedures based on Gricean maxims exist in the system as ready-made packages, in order to avoid indefiniteness due to the informality of the maxims. To devise a dynamic model to calculate 'conversational implicatures' purely on the basis of the maxims is an interesting topic of research for the future, which is ignored here[4].

These pragmatic procedures, which are attached to each lexical item, could be considered to be the "experts" on each lexical item, a

[3] Of course, this argument does not preclude any future attempt for procedural explanation of implicatures without relying on the explicit ordering of the procedures. Without any empirical evidence against the current approach, and since this formulation gives us at least a reasonable interaction between semantic (DEN) and pragmatic (PREC and POSTC) procedures in a mechanized natural language understanding, I will use this troika system for the moment.
concept which is widely used in natural language understanding systems and other kind of systems developed in artificial intelligence[5].

6.3. IMPLICATURES OF FACTIVE VERBS

As has been mentioned, factive verbs 'conventionally implicate' or 'presuppose' the complement sentence depending on the environment in which they occur. Let us take (5.10b) and (5.10a), which are repeated as (6.3) and (6.4) here, as paradigmatic examples of factive verbs and see how new information is or is not added to the model. An EIL translation is provided for each of the following sentences (in fact and though are not translated in favor of the simplicity of the presentation).

(6.3) a. Jack doesn't regret being bald
     (NOT (REGRET J (INT (BALD J))))
     b. (in fact) he isn't bald
     (NOT (BALD J))

(6.4) a. *Jack regrets being bald
     (REGRET J (INT (BALD J)))
     b. (though) he isn't bald
     (NOT (BALD J))

With regard to the cancelability of information, we will make use of the following terminology to distinguish different kinds of information in the system: nc-info for [-cancelable] information, which is the normal state of information in the model; c-info for [+cancelable] information, which can be overridden by later nc-info if it contradicts with the c-info (cancellation) or if it repeats the c-info (consolidation); and a-info for "abortable" and "absorbable" information, which only temporarily exists during the process of compositional interpretation and will eventually be either deleted (abortion or absorption) or changed to c-info (creation).

[4] Wilson (1975) presents arguments concerning this point. Schwarz (1979) offers a criticism of her arguments and an alternative proposal, though, as he himself admits, it is based on a vague and informal concept and doesn't seem to be mechanizable. See also footnote [10] in Chapter 5.
[5] Kaplan (1978) 's system performs a similar pragmatic computation in order to make his question-answering system more "friendly." A simple negative answer would not be adequate if the falsehood of the inquiry resulted from the failure of the "presupposition." His system would detect this kind of failure and gives such additional information to the inquirer.
As can be seen from the semantics of EIL, the compositional interpretation of formulas takes place from the most embedded construction to the outermost one. Thus, in (6.3a), when the interpreter tries to evaluate the whole formula, it first evaluates the argument of NOT, i.e., \((\text{REGRET} \ J \ (\text{INT} \ (\text{BALD} \ J)))\), which is the formula corresponding to the sentence "Jack regrets being bald." It has a predicate REGRET and two arguments \(J\) (Jack) and \((\text{INT} \ (\text{BALD} \ J))\), the formula which corresponds to the intension of the complement sentence: "Jack is bald." Note that the second argument is an intension, since what participates in the compositional semantics is the proposition (function on possible worlds) that the sentence represents, not its truth-value.

The PREC procedures for factive verbs operate according to the following schema.

\[(6.5)\text{ For PREC procedures of factive verbs for 'presuppositions' and 'conventional implicatures' } \Rightarrow \text{ Produce a piece of a-info corresponding to the second argument.}\]

In general, the task of PREC procedures attached to the translations of factive verbs is to produce a piece of a-info which corresponds to the extension of the second argument of the verb. That is, they add to the context a piece of information corresponding to the complement sentence of the factive verb with a reservation ("abortable"). Thus, for REGRET in (6 3a) and (6.4a), before the evaluation of its DEN function, which checks if Jack is in the regret-relation with his baldness, its PREC procedure nominates the denotation of \(J\) (i.e., the person Jack) as a candidate for a member of the d-list of BALD, since the intension of (BALD \(J\)) occurs as the second argument[6]. At this stage, Jack's baldness is neither "asserted" nor "presupposed;" it is only conceived and still have the chance of abortion. The final status of such information is determined at the level of super-interpretation.

Let us consider the case of 'presupposition,' (6.3a), first. In the case of (6.3a), the factive verb is in the scope of the negative operator. In general, if a factive verb is in the scope of negation, or a similar higher-level operator, (which I call a cancelability-inducing operator, or a ci-op for short), such as a verb of "propositional attitude" (e.g., suspect, think), or a verb of saying (e.g., say, claim), the abortable information produced by the

[6] Technically speaking, the PREC procedure has to be defined as an FEXPR rather than an EXPR, i.e. the arguments should not be evaluated prior to the execution of the function. Note that if the second argument is evaluated before the execution of the PREC procedure of REGRET, the PREC procedure has access only to a set of possible worlds, not the predicate BALD nor the constant \(J\).
PREC procedure becomes cancelable. This is taken care of by the POSTC procedures attached to these operators which work in the following way (cf. Appendix A, particularly the definition of 'create-c-info')[7].

(6.6) For POSTC procedures of ci-ops for 'presuppositions' =>

a. If the a-info is contradictory to the model then delete the a-info (abortion of 'presupposition').

b. If the a-info is repeated by nc-info in the model then delete the a-info (absorption of 'presupposition').

c. If the a-info is neither contradictory nor duplicate then change the a-info to c-info (creation of 'presupposition').

Thus, if the a-info produced by the PREC procedures in the scope of the operator is contradictory to the nc-info in the current model at some level of UDs, then it is aborted by the POSTC procedure of the ci-op, i.e., it is deleted from the model since the nc-info in the current model overrides it. If, on the other hand, the same information as the newly produced a-info is already existent in UDs in the form of nc-info, the new a-info is absorbed by the nc-info, i.e., it is also deleted from the model, retaining only the solid nc-info in the model. Finally, if no relevant information whatsoever concerning the a-info is in UDs (no contradictory or duplicate nc-info), then the POSTC procedure of the ci-op changes the status of the newly produced a-info to c-info, i.e., from "abortable" to "cancelable." Thus the information is maintained in the model but the possibility of its being overridden by later nc-info still remains (creation). There are two cases of overriding: cancellation and consolidation. The former occurs when a piece of contradictory nc-info is later added to the model, and the latter when a piece of duplicate nc-info is later added to the model. In either case the nc-info is favored over the c-info. (Cf. the definitions in Appendix A for details. Appendix C shows how UDN is changed during the processing of (6.3a) as well as that of (6.4a), which will be explained shortly, based on simple models.)

Note that, after the processing by POSTC procedures of ci-ops, no a-info remains in the model; it is either deleted (aborted due to the existence of contradictory nc-info, or absorbed by duplicate nc-info) or changed to c-info (created as a 'presupposition'). The c-info is put in UDN and all the lower UDs as cancelable information by the super-interpreter. Thus, in (6.3a), Jack is added to the d-list of BALD as c-info, and that Jack is bald is implicated as a 'presupposition.' Since (6.3b) contradicts with this c-info, and the

[7] This kind of POSTC procedure works as a filter of possible implicatures produced by PREC procedures attached to factive verbs. In a sense, it is similar in function to POSTC procedures attached to lexical items which produce 'conversational implicatures' such as the disjunction and the conditional, in that they both manipulate the information coming from lower levels.
information supplied by (6.3b) is nc-info, the 'presupposition' is canceled after processing (6.3b). The various aspects of the "presupposition" are thus correctly described based on the pragmatic mechanisms developed in the EIL system. Shown below is the modified version of chart (5.23), which is augmented with the concepts defined in this section, with the example sentences in Chapter 5 repeated below.

<table>
<thead>
<tr>
<th>(6.7)</th>
<th>Existing Context</th>
<th>Change of Information</th>
<th>Later Context</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Abortion</td>
<td>-</td>
<td>a-info -&gt; deleted</td>
<td>N/A</td>
<td>(5.17b)</td>
</tr>
<tr>
<td>b. Absorption</td>
<td>+</td>
<td>a-info -&gt; deleted</td>
<td>N/A</td>
<td>(5.21)</td>
</tr>
<tr>
<td>c. Creation</td>
<td>0</td>
<td>a-info -&gt; c-info</td>
<td>N/A</td>
<td>(5.20)</td>
</tr>
<tr>
<td>d. Cancellation</td>
<td>N/A</td>
<td>c-info -&gt; deleted</td>
<td>-</td>
<td>(5.17a)</td>
</tr>
<tr>
<td>e. Consolidation</td>
<td>N/A</td>
<td>c-info -&gt; deleted</td>
<td>+(0)</td>
<td>(5.17c)</td>
</tr>
</tbody>
</table>

(5.17b) Jack isn’t bald. So, he doesn’t regret being bald.
(5.21) Jack is bald. But he doesn’t regret being bald.
(5.20) Jack doesn’t regret being bald.
(5.17a) Jack doesn’t regret being bald. For he isn’t bald.
(5.17c) Jack doesn’t regret being bald, even though he is bald at the age of 20.

The newly produced a-info is either deleted, as in (6.7a) and (6.7b), or transformed into c-info as in (6.7c). This c-info may still be deleted depending on the later context, as in (6.7d) and (6.7e). As for consolidation, (6.7e), a neutral context for c-info seems almost like a positive context. That is, even if there is no positively consolidating information, the very fact that there is no contradictory information seems to serve to confirm the c-info. See section 6.4 for a discussion of this phenomenon.

As for the case of 'conventional implicature,' (6.4a), since the factive verb is not in the scope of a ci-op, the a-info still remains without any change of its status after all the compositional computation is performed. Its consistency with the model is checked post-compositionally by the super-interpreter according to (6.8) below. (Cf. Appendix A, particularly the definition of 'reate-nc-info.')

(6.8) For super-interpreter for 'conventional implicatures' =>

a. If the a-info is contradictory to the model then notify the utterer of the contradiction (abortion due to the violation of convention).

b. If the a-info is repeated by nc-info in the model then delete the a-info (absorption of 'conventional implicature').

c. If the a-info is neither contradictory nor duplicate then change the a-info to nc-info (creation of 'conventional implicature').
The super- interpreter considers all the nominations by PREC procedures and establishes the following. If the a-info is contradictory to the current model, the super-interpretation is suspended and the utterer is required to clarify the situation. This is a violation of the conventional use of the word, since what is supposed to be a 'conventional implicature' is in contradiction to the context (cf. (4.1b)). Note that (6.6a) and (6.8a) differ in this respect (in (6.6a) the a-info is simply aborted without inquiring the utterer). The second case of (6.8) is similar to (6.6b). If the a-info is repeated by nc-info in the model, it is absorbed by the nc-info, i.e. it is deleted from the model. In the remaining case, (6.7c), if there is no contradiction or duplication, the a-info is changed to nc-info, i.e., it becomes noncancelable, and is put in UDn and the lower UDs (cf. (4.1c)). Note that in the case of 'conventional implicature' the a-info becomes nc-info, not c-info as in (6.6c). (Cf. Appendix C. Note that the difference of UDn's after the processing of (6.3a) and that of (6.4a).) Thus, in (6.4a), Jack is added to the d-list of BALD as nc-info, and that Jack is bald is implicated as a 'conventional implicature.' Since (6.4b) contradicts this nc-info, the total utterance (6.4) is unacceptable. The utterer is inquired while the interpreter is processing (6.4b). There is no case of cancellation or consolidation since the information is not cancelable. Note the similarity between the addition of a 'conventional implicature' and that of information 'denoted' by an indicative sentence as have been seen in (4.1). The only difference is the way the information is communicated; as the feature says, the former is 'implicated,' while the latter is directly 'said.'

In summary, positive factive verbs trigger addition of noncancelable information, and negative factive verbs trigger addition of cancelable information. The former kind of information has the features [-said,+conventional,+cancelable] and called a 'conventional implicature,' while the latter kind of information has the features [-said,+conventional,+cancelable] and is called a 'presupposition.' Note that the mechanism to produce the additional, implicated information is the same for both affirmative and negative formulas; it comes from the PREC procedure of the factive verb in the formula in both cases. Only the cancelability of the produced information is affected by pragmatic (POSTC) procedures attached to higher operators. In this sense, POSTC procedures behave as filters of implicatures created by PREC procedures of factive verbs; we will discuss some similarity and difference of pragmatic procedures to Karttunen and Peters (1979)'s "heritage functions" in Section 7.3. Note that in our system, the production, the filtration, and the addition of the implicature are taken care of by three different components, namely, PREC procedures of factive verbs, POSTC procedures of ci-ops, and the super-interpreter, respectively. As will turn out, the dynamic addition and deletion of newly-created information is one of the major features to distinguish the current system from others.
6.4. DEGREE OF CANCELABILITY

By definition, c-info is cancelable, but its cancelability seems to change as time goes by. That is, if a piece of nc-info which is contradictory to a piece of c-info in UDn is put in UDn+1 (i>0) later, it may cancel the c-info only in the case that i is sufficiently small[8]. That is, if several layers of UDn are added on top of UDn and a piece of c-info in UDn is not challenged by any information in those additional UDn, the c-info will apparently be promoted to nc-info, and becomes no longer cancelable. In this sense, the nonexistence of contradictory (canceling) information is tantamount to the existence of consolidating information. The average time length that a piece of information remains cancelable may not be so long, and every bit of information in UDn seems to have eventually the same status regardless of its origin. namely noncancelable.

Or, to regard the situation from a different point of view, the property of cancelability may not be a binary property but there is a continuum between the solid, completely noncancelable information and the very fragile information subject to easy cancellation. One way to approximate this situation would be to make the feature [cancelable] a multi-valued feature and decrease the value by one as another layer of UD is put on top of the older ones. As has been mentioned in Section 4.2, in a practical application, the number of different UDn may not have to be so large. Thus, if there are only k UDn: UDn, UDn-1, ..., and UDn-k=UDO, the degree of cancelability of a piece of information newly put in UDn can start with k, i.e., the feature assignment will be [kcancellable], and it will decrease one by one as the information "sinks" into the deeper UD, until it becomes 0, i.e. [0cancellable] or noncancelable, in UDO. Thus, in a more refined system, this kind of modification will produce more satisfactory results. I will, however, use the binary feature for cancelability throughout the dissertation, since it is sufficient for the scope of this dissertation.

6.5. GRICEAN IMPLICATURES

Let us quickly see how other kinds of implicatures are treated in the EIL system. Since the details of the mechanism are similar to the case of factive vabs, the descriptions are only cursory. As for 'conventional implicatures' we have seen that factive verbs 'conventionally implicate' the complement if it is not in the scope of a ci-op. As a different kind of example of 'conventional implicatures,' take BUT. As has been mentioned, the truth condition

[8] The determination of the exact bound of the smallness of i is an empirical, and possibly idiolectal, matter and of no concern here. There seems to be no problem for i=1, since (6.3a) and (6.3b) could be separate but consecutive utterances. (7.23) in the next chapter is an example with a larger i.
for **BUT** is the same as **AND** since the sentence is true only when both conjuncts are true, as can be seen in (6.9).

(6.9) a. *Jack is bald and so is Paul, though Jack isn't bald.*
    \[(\text{AND} (\text{BALD J}) (\text{BALD P})) (\text{NOT} (\text{BALD J}))\]

    b. *Jack is bald and so is Paul, though Paul isn't bald.*
    \[(\text{AND} (\text{BALD J}) (\text{BALD P})) (\text{NOT} (\text{BALD P}))\]

d. *Jack is bald but so is Paul, though Paul isn't bald.*
    \[(\text{BUT} (\text{BALD J}) (\text{BALD P})) (\text{NOT} (\text{BALD J}))\]

c. *Jack is bald but so is Paul, though Jack isn't bald.*
    \[(\text{BUT} (\text{BALD J}) (\text{BALD P})) (\text{NOT} (\text{BALD P}))\]

Unlike **AND**, **BUT** has a 'conventional implicature,' which, in the case of (6.9c) and (6.9d), implicates that the fact that Paul is bald is in contrast with Jack's baldness. This kind of contrast is not cancelable, as (6.10) sounds odd.

(6.10) *Jack is bald but so is Paul, although I don't mean any contrast in the two facts.*

In the EIL system, PREC procedures attached to lexical items which bear 'conventional implicatures' add appropriate information (e.g., the fact that the two conjuncts are in contrast) to the model. Since it is not cancelable, it has the feature [¬-cancelable], i.e., it has the status of nc-info, from the beginning, that is, no a-info is produced in this case, since there is no parallel 'presupposition' even if **BUT** is embedded in a ci-op. If the new information contradicts the current model, the interpretation is suspended and the utterer would be required to supply more information to clarify the situation. If, on the other hand, the new information is consistent with the model, the interpretation goes on to the next utterance without any interruption.

As for 'conversational implicatures,' as has been argued, some of the 'conversational implicatures' can be considered to be associated with particular lexical items of EIL for all practical purposes. For example, **or**, or other lexical items with the same truth condition, 'conversationally implicates' that it is an exclusive-or rather than an inclusive-or. This implicature is a conversational one since it is cancelable, as can be seen seen in (6.11):

(6.11) a. Jack is bald or he is fat
    \[(\text{OR} (\text{BALD J}) (\text{FAT J}))\]

    b. (in fact) he is both bald and fat
    \[(\text{AND} (\text{BALD J}) (\text{FAT J}))\]

In our system, the POSTC procedure attached to **OR** will add new cancelable information that one of the disjuncts is false if the other has turned out to be true. For example, if the truth of (6.11a) is established by confirming that Jack is bald, e.g., by finding out that
the denotation of J is in the d-list of BALD, then that Jack isn't fat is produced as a-info, i.e., the denotation of J is put into the nd-list of FAT as a-info. (Note that, according to the semantic condition of OR, (2.41), if the first disjunct evaluates to T, the second disjunct is not evaluated in the process of establishing the truth of the disjunction.) The treatment of this newly produced a-info is quite parallel to the case of the information created by factive verbs in the scope of negation, except that, in this case, the POSTC procedure of OR takes care of all the necessary manipulations of a-info, i.e., absorption, abortion, and creation, since there isn't, and need not be, any higher operator to associate with the appropriate pragmatic procedure; a 'conversational implicature' is created regardless of the environment of the implicature-bearing lexical item. Thus, if the model happens to contain nc-info that Jack is not fat, then the a-info is deleted (absorbed). If, on the other hand, the model contains nc-info that Jack is fat, then the a-info is deleted (aborted). Finally, if there is no information as to Jack's fatness in the model, then the a-info is changed to c-info (created). (Cf. the definition of 'create-c-info' in Appendix A.) Since this new creation is cancelable, (6.11b) is allowed to cancel the c-info that Jack isn't fat. Note that the addition of new information has to wait until the DEN procedure associated with OR evaluates at least one of the arguments. Since the POSTC procedure is executed after the DEN procedure, this is exactly what happens in the system. Similar implicatures, e.g., "not all" for some, "not necessarily" for possibly, etc. ("scalar quantity implicatures," cf. Section 5.2), are realized by attaching similar POSTC procedures to SOME, POS, etc.

The clausal connectives if...then, or its translation IMP, either...or, OR, etc. can also be associated with a POSTC procedure to add new cancelable information that the clauses connected by these are not true ("clausal quantity implicatures," cf. Section 5.2).

(6.12) a. If Jack is bald, then so is Paul
(IMP (BALD J) (BALD P))

b. (in fact) Jack is bald
(BALD J)

The mechanism is quite parallel to the above case; the POSTC procedure attached to IMP will create a-info which corresponds to the negation of the arguments, which will be either absorbed, aborted, or created depending on the model. In the case of (6.12), (6.12b) cancels the c-info that Jack is not bald, if it survives abortion and is newly created by (6.12a).

This concludes the detailed examination of the pragmatic procedures utilized in the EIL system. In the next chapter, the current system is compared with the recent proposals from a slightly different perspective, although they share the sufficient degree of formal nature so that the comparison is meaningful. The chapter
begins with a discussion of the general property of implicatures whose treatment is one of the most important tasks of the systems compared.
7. PROJECTION PROBLEM

7.1. CUMULATIVE HYPOTHESIS

As has been seen in the previous chapter, the behavior of DEN procedures and that of PREC and POSTC procedures are quite different. A DEN procedure evaluates its arguments, computes the denotation, and hands on the result to another DEN procedure which is in a higher position in the hierarchy i.e., a procedure which calls the first one. This is both a usual way of evaluating (pure) Lisp functions (EXPRs) and a compositional structure of model-theoretic semantics. Although DEN procedures do have access to the model (data base), the access mode is read-only, i.e., they cannot modify the model. On the other hand, both PREC and POSTC procedures are not "functions" in the usual sense. The values which are given as the results of these procedures are immaterial to the system, since they are never used in the compositional semantics. These are "pseudo-functions" in the sense that what is important about them is their side-effects to the environment when they are executed. The environment is the model, or UDs, in our system. Thus, the access mode for pragmatic procedures is read-and-write. This kind of characterization of PREC and POSTC procedures are both necessary and sufficient for the treatment of 'what is implicated,' since it behaves differently from 'what is said.' The characteristic behavior of implicatures in complex sentences have been known by the name of the "projection problem" (Langendoen and Savin 1971).

Recall the treatment of 'conventional implicatures' and 'presuppositions' associated with factive verbs. One of the notable features of 'what is implicated' arising from the complement sentence of a factive verb is that it survives negation, that is, the implicature is still communicated even though it becomes cancelable. This property is remarkably different from 'what is said,' which once negated, ceases to exist; if a sentence A entails a sentence B, then not A doesn't entail B or not B, whether cancelable or not. In other words, not A is neutral to B. the entailment of A. On the other hand, not A still shows some degree of commitment to the implicature of A. In fact, this peculiarity of 'what is implicated' is not limited to negation. For example, take a modal operator.

(7.1) It is possible that Jack regrets being bald. (POS (REGRET J (INT (BA LD J))))
Implications that Jack is bald. Note that the formula expressing the information that Jack is bald, i.e., \((\text{BALD} \ J)\), is in the scope of \(\text{REGRET}\), which is in turn in the scope of \(\text{POS}\). This cannot be 'said,' since the modal operator \(\text{POS}\) doesn't entail the truth of its argument, \((\text{REGRET} \ J (\text{INT} (\text{BALD} \ J)))\). That is, the implicature cannot be attributed to the truth of the embedded sentence, "Jack regrets being bald," which is not entailed by (7.1), even though it is entailed to implicate Jack's baldness. Thus, even though the entailment is blocked by the modal operator, the implicature seems to freely pass through such an operator; it doesn't need an intermediate stage to surface. Thus, it seems that operators, such as negation, modals, etc., behave as transparent filters to implicatures (with the only possible effect being the introduction of cancelability in the case of negation)[1].

Langendoen and Savin (1971), analyzing complex sentences and examining how the "assertions" and "presuppositions" are projected (inherited) from the component sentences, conclude that only the "assertions" of the component sentences are used compositionally by higher constructions while the "presuppositions" of the component sentences by themselves become the "presuppositions" of the higher complex sentence. This generalization, or of a similar kind, which Morgan (1969) calls the "cumulative hypothesis," can be schematized as (7.2):

(7.2) Let a complex sentence have a functional structure \(A(B)\), i.e., \(B\) is an embedded sentence within the scope of the predicate of the higher sentence \(A\). Let \(\text{ass}_A\) and \(\text{ass}_B\) be "assertions" associated with \(A\) and \(B\), respectively, and let \(\text{pre}_A\) and \(\text{pre}_B\) be "presuppositions" associated with \(A\) and \(B\), respectively. Then,

\[
\begin{align*}
\text{a. } \text{ass}_A(\text{ass}_B) \\
\text{b. } \text{pre}_A(\text{ass}_B) \\
\end{align*}
\]

Note that only \(\text{ass}_B\) is used as an argument of either \(\text{ass}_A\) or \(\text{pre}_A\). \(\text{pre}_B\) by itself becomes the "presupposition" of the complex sentence \(A(B)\), together with the "presupposition" associated with \(A\), namely, \(\text{pre}_A(\text{ass}_B)\).

The scheme described above is actually the fundamental principle on which the EIL procedures operate. That is, in the EIL system, the above principle can be restated in the following way.

[1] It may be argued that 'what is implicated' by (7.1) is not that Jack is bald, but that it is possible that Jack is bald. However, this kind of treatment, which embeds the implicature in the operator of the higher sentence, wouldn't apply to negation, since what "Jack doesn't regret being bald" implicates is not that Jack is not bald. As will be discussed in Section 7.2, Karttunen (1973) calls operators which let implicatures of the lower constructs emerge "holes."
Let \((p q)\) be a formula of EIL, where \(p\) is a constant and \(q\) is another formula. Then,

- the 'denotation' of \((p q)\) is the result of applying the DEN procedure of \(p\) to the 'denotation' of \(q\).
- the implicature of \((p q)\) is the implicature of \(q\) and the result of applying the pragmatic procedures of \(p\) to the 'denotation' of \(q\).

It should be stressed that (7.3) is merely a summary of how DEN, PREC, and POSTC procedures behave in the system, and not a newly introduced principle (7.3a) is a paraphrase of the semantic condition (2.4f), or (6.2) in the revised version. (7.3b) comes from the fact that PREC and POSTC procedures cause side-effects in UDs, which are naturally inherited by higher constructions up to the top level of evaluation if no further change is performed. Thus, the EIL system is designed to achieve the cumulative aspect of implicatures by making a fundamental distinction between DEN and PREC/POSTC procedures as "real functions" and "pseudo-functions."

The cumulative hypothesis, however, cannot be maintained with its naive form as shown above, where implicatures are always inherited from the deepest structure to the topmost one without any modification. So far, we have seen that 'what is implicated' gets a subtle modification as to its cancelability when factive verbs are embedded in negation or similar operators (ci-ops). Thus, the implicatures do not always emerge with their original forms. Moreover, Morgan (1969) cites more direct deviations, where the implicatures are completely lost, in the construction in which a sentence is embedded in a conditional:

\[(7.4)\]

- If Jack is bald, then he doesn't regret being bald.  
  \[\text{IMP (BALD J) (NOT (REGRET J (INT (BALD J))))}\]
- If Jack is bald, then he regrets being bald.  
  \[\text{IMP (BALD J) (REGRET J (INT (BALD J))}\]

In (7.4a), the consequent of IMP, when considered by itself, 'presupposes' that Jack is bald, but the whole sentence doesn't have this 'presupposition.' This might be explained by appealing to the cancelability of 'presuppositions' and indeed Gazdar (1979) explains sentences like (7.4a) along this line (cf. Section 7.4). However, cancelability is of no help in the case of (7.4b), since the consequent of IMP, when considered by itself, 'conventionally implicates' that Jack is bald. This information is, as we have seen, noncancelable, but the whole sentence lacks this implicature. It is not correct to include conditionals in ci-ops since, as we will see in (7.10a) later, conditionals do not always induce cancelability. Thus, we need a new type of explanation.
In this chapter, we will examine some of the formal approaches to the projection problem, focusing on how cases like (7.4) are treated in those systems as well as in the EIL system. As will be shown, it turns out that in the EIL system we do not essentially need any conceptually new mechanisms to the determination of the implicatures of compound sentences; the computational characteristics of the interpreter, which have already been described in previous chapters— particularly its sequential evaluation of the constituents and its dynamic interaction with the model (UDs)— will naturally solve the projection problem. First, let us begin with Karttunen's descriptions of the phenomena and see how the EIL system explains them. Later in Sections 7.3 and 7.4, Karttunen and Peters (1979)'s treatment of these phenomena and Gazdar (1979)'s solution will be discussed.

7.2. "PLUGS," "HOLES," AND "FILTERS"

Karttunen (1973) classified lexical items into three classes depending on how the "presuppositions" of the complement sentence is inherited to the higher sentence[2]:

\[(7.5)\]
\[\text{a.} \quad \text{"plugs"} - \text{verbs which block all the "presuppositions" of the embedded sentence, e.g., say, claim, promise.} \]
\[\text{b.} \quad \text{"holes"} - \text{verbs which let all the "presuppositions" of the embedded sentence become those of the higher sentence, e.g., know, regret, stop.} \]
\[\text{c.} \quad \text{"filters"} - \text{connectives which block some of the "presuppositions" of the embedded sentence, e.g., if...then, and, or.} \]

Karttunen's paper (1973) serves as a guideline to tackle the projection problem, particularly his formalization of the behaviors of logical connectives. Although some of his descriptive devices have been severely attacked subsequently, particularly the idiosyncratic classification of lexical items into "plugs" and "holes," e.g. there are two kinds of negations: one as a "plug" (the so-called external negation) and the other as a "hole" (the so-called internal negation), his characterization of "filters" seems to be essentially correct at least as a general description of the phenomena, even though it may not be an explanation. Any formal theory therefore, has to be able to achieve at least the similar result as Karttunen's "filtering" effect within its framework.

[2] Karttunen later, with Peters (1977), writes a requiem for "presuppositions" and renames most of them "conventional implicatures." I refer to them as implicatures in the broader sense, i.e., [-said] information. Note that Karttunen's "presupposition" (before circa 1975) includes 'presupposition' and 'conventional implicature' here.
In this section we will see how the above description is interpreted and modified in the EIL system, with much emphasis on the "filtering" phenomena and their explanations by the mechanisms of EIL. Note that if the cumulative hypothesis were the only principle governing the projection of implicatures, all verbs would be "holes," and, as we have seen above, the EIL system is already capable of handling them since implicatures are newly created information which exists in the model and are naturally carried over to the higher sentence. Since there are cases where the cumulative hypothesis does not hold, we have to be able to treat these cases. Let us see whether these cases will necessitate a major modification to the system.

As for "plugs," which are English verbs, since they are translated into predicates in EIL (e.g., say to EIL's SAY, claim to EIL's CLAIM), we can make use of pragmatic procedures attached to these predicates to specify their properties as "plugs." For example, SAY will have a POSTC procedure, which will be similar to the POSTC procedure attached to negation (cf. Appendix A), to cancel any a-info which has been created by the complement sentences. Thus, by virtue of the pragmatic procedure which is capable of deleting information, the system can achieve the effect of the class described above as "plugs" without adding any new device. Moreover, what a pragmatic procedure can do is much more general and flexible than simply deleting information and we can make any finer distinction of the inherited information depending on each lexical item. That is, there is no reason to believe that there are only two modes of the inheritance of information (deletion or no change). We have already seen how the POSTC procedures of ci-ops modify the cancelability of information as well as delete the information. As Gazdar (1979, p.109) criticizes, it is doubtful whether the "plugs" and the "holes" form natural classes. They seem to exemplify only the extreme cases at both ends of the spectrum. In fact, Karttunen himself set up a third class which lie somewhere between the "plugs" and the "holes" in his later work[3]. It seems that the degree of transparency varies continuously (and idiosyncratically) depending on each lexical item. Since the pragmatic procedure is attached to each lexical item at the EIL level, it could take care of any finer classification of verbs, if any, and it could offer a much more flexible specification of the transparency. Since Karttunen's later papers, particularly those in collaboration with Peters, don't seem to put much emphasis on phenomena other than "filtering," I will not go into any further

[3] Karttunen (1974) reclassifies verbs which belong to the classes of "plugs" and "holes" into three types: I for verbs of saying (e.g., say, ask); II for verbs of "propositional attitude" (e.g., believe, fear); and III for everything else (e.g., factive verbs, modals). Type I is the same class as "plugs" and Type III as "holes." Type II verbs modify the "presupposition" of the complement as the belief of the subject of the verbs.
"Filters" are not verbs, but what are translated into logical connectives. Consider (7.6):

(7.6a) Jack doesn't regret being bald.
   (NOT (REGRET J (INT (BALD J)))))

(7.6b) Jack regrets being bald.
   (REGRET J (INT (BALD J)))

(7.6c) Baldness is hereditary and Jack doesn't regret being bald.
   (AND (HEREDITARY BALDNESS)
   (NOT (REGRET J (INT (BALD J)))))

(7.6d) Baldness is hereditary and Jack regrets being bald.
   (AND (HEREDITARY BALDNESS)
   (REGRET J (INT (BALD J))))

(7.6e) Jack is bald and he doesn't regret it.
   (AND (BALD J)
   (NOT (REGRET J (INT (BALD J)))))

(7.6f) Jack is bald and he regrets it.
   (AND (BALD J)
   (REGRET J (INT (BALD J))))

(7.6a) 'presupposes' and (7.6b) 'conventionally implicates' that Jack is bald. As has been discussed in Chapter 6, these 'presupposition' and 'conventional implicature' are created by a PREC procedure which is associated with REGRET, which is the translation of the factive verb regret. (7.6c) - (7.6f) are cases where (7.6a) or (7.6b) is embedded in a complex sentence. (7.6c) also 'presupposes' and (7.6d) also 'conventionally implicates' that Jack is bald. Thus, the cumulative hypothesis gives correct predictions for these cases. However, (7.6e) does not 'presuppose' and (7.6f) does not 'conventionally implicate' that Jack is bald; Jack's baldness is rather 'said' by these sentences, since the same information is explicitly stated in the first conjuncts in both sentences. Accordingly, Karttunen (1973) summarizes the observation in the form of the following "filtering" rule for and:

(7.7) Let S be "A and B." If A "presupposes" C then S "presupposes" C. If B "presupposes" C then S "presupposes" C unless A entails C.

According to (7.7), (7.6c) and (7.6d) "presuppose" that Jack is bald since the second conjuncts "presuppose" it and the first conjuncts don't entail it. On the other hand, in the cases of (7.6e) and (7.6f), they don't "presuppose" that Jack is bald, since the first conjuncts entail (in fact, are identical to) that Jack is bald.
Although (7.7) gives the correct prediction, it simply summarizes the phenomena and doesn’t explain why that is the case, nor does it give any formal mechanism to achieve the described result. However, as Steedman (1977) argues, this phenomenon is rather an inherent consequence of the sequential evaluation mechanism of logical connectives. That is, in the EIL system, or any system whose evaluation mechanism of logical connectives is sequential, we can have a natural explanation for this rule based on the semantics of logical connectives. Remember the interpretation rule for AND in the EIL semantics, (2.4k), which is repeated here as (7.8):

\[ (7.8) \text{If } p \text{ and } q \text{ are members of } L_t, \text{ then} \]
\[ V[(\text{AND } p \ q), w, t, f, n, i] \]
\[ \text{def} = \begin{cases} 
\text{val} = V[p, w, t, f, n, i]; \\
\text{if } (\text{val}) V[q, w, t, f, n, i]; \\
\text{else if } (!\text{val}) F; \\
\text{else if } (!V[q, w, t, f, n, i]) F; \\
\text{else } U 
\end{cases} \]

The point in (7.8) is that the second conjunct \( q \) is evaluated only when the first conjunct \( p \) evaluates to true (the fourth line of (7.8)), or it is undefined (the sixth line). In either case, the evaluation of \( q \) follows that of \( p \), not vice versa nor are they evaluated at the same time. This means that, in (7.6e) and (7.6f), the second conjuncts, (NOT (REGRET J (INT (BALD J)))) and (REGRET J (INT (BALD J))) respectively, are evaluated only after the first conjuncts, (BALD J) for both sentences, are evaluated. Thus, if Jack is bald in the model, then the second conjuncts are evaluated according to the fourth line of (7.8). Since Jack’s baldness exists in the model as nc-info, the a-info produced by the second conjuncts is deleted (absorbed by the duplicate nc-info), hence no ‘presupposition’ or ‘conventional implicature’ is created by these sentences. If on the other hand, Jack is not bald in the model, the interpreter immediately concludes that the conjunction is false according to the fifth line of (7.8), without ever bothering to evaluate the second conjuncts, hence there will be no creation of new information, either. (Remember that, according to (2.3b), exp2 is not evaluated if expl evaluates to F.) If Jack’s baldness is undefined in the model, it will be added to the model as nc-info by the super-interpreter according to (4.1c), which is based on the truthfulness of the utterer immediately after the processing of the first conjunct. Thus, by the time the second conjunct is evaluated, Jack’s baldness exists as nc-info in the model and it absorbs the a-info produced by the second conjunct. In this way, in any case, no new information is added to the model as a ‘presupposition’ of (7.6e) or as a ‘conventional implicature’ of (7.6f). This is why (7.6e) doesn’t ‘presuppose’ and (7.6f) doesn’t ‘conventionally implicate’ that Jack is bald. (Absorption doesn’t occur in (7.6c) and (7.6d) because the first conjuncts don’t produce the nc-info that Jack is bald to absorb the a-info.) Note that if AND were an ordinary truth
function, which always evaluates both conjuncts before calculating the truth-value of the conjunction, there would have to be an additional mechanism to suspend the creation of implicatures in cases like (7.6e) and (7.6f). We will see two of such attempts by Karttunen and Peters and by Gazdar in Sections 7.3 and 7.4. Defining AND, and other connectives, in the way as (2.4) gives "filtering" rules like (7.7) the status of corollaries of the semantics in the overall system, not independently motivated pragmatic rules. Note that in the above examples, exactly the same concept as in the intersentential analysis we have seen in the previous chapter is used intrasententially, namely, absorption of a-info by nc-info.

Similar arguments apply to other connectives. (7.9) are examples for the disjunction and (7.10) are for the conditional:

(7.9) a. Either baldness isn't hereditary or Jack doesn't regret being bald.
   \[(\text{OR} (\text{NOT} (\text{HEREDITARY BALDNESS}))
   (\text{NOT} (\text{REGRET J (INT (BALD J))})))\]

b. Either baldness isn't hereditary or Jack regrets being bald.
   \[(\text{OR} (\text{NOT} (\text{HEREDITARY BALDNESS}))
   (\text{REGRET J (INT (BALD J))}))\]

c. Either Jack isn't bald or he doesn't regret it.
   \[(\text{OR} (\text{NOT (BALD J)})
   (\text{NOT} (\text{REGRET J (INT (BALD J))})))\]

d. Either Jack isn't bald or he regrets it.
   \[(\text{OR} (\text{NOT (BALD J)})
   (\text{REGRET J (INT (BALD J))}))\]

(7.10) a. If baldness is hereditary, then Jack doesn't regret being bald.
   \[(\text{IMP (HEREDITARY BALDNESS)}
   (\text{NOT (REGRET J (INT (BALD J))))})\]

b. If baldness is hereditary, then Jack regrets being bald.
   \[(\text{IMP (HEREDITARY BALDNESS)}
   (\text{REGRET J (INT (BALD J))))}\]

c. If Jack is bald, then he doesn't regret it.
   \[(\text{IMP (BALD J)}
   (\text{NOT (REGRET J (INT (BALD J))))})\]

d. If Jack is bald, then he regrets it.
   \[(\text{IMP (BALD J)}
   (\text{REGRET J (INT (BALD J))))}\]

As in (7.6c) through (7.6f), (7.9a) and (7.9b) implicate that Jack is bald, since the second disjuncts implicate it, while (7.9c) and (7.9d) don't possess the same implicature. In this case, the reason is that the first disjuncts have explicitly denied it. Karttunen (1973) gives the following rule for or separately from (7.7):
(7.11) Let $S$ be "Either $A$ or $B$." If $A$ "presupposes" $C$ then $S$ "presupposes" $C$. If $B$ "presupposes" $C$ then $S$ "presupposes" $C$ unless not-$A$ (or the context) entails $C$.

Again, this additional rule is a natural consequence in the EIL system, since the evaluation procedure for OR, (2.41), evaluates the second disjunct only after the first disjunct is evaluated. Thus, if Jack is bald, the second disjuncts in (7.9c) and (7.9d) are evaluated in a model where Jack is bald and naturally no 'presupposition' or 'conventional implicature,' is created as new information; the $a$-info that Jack is bald is absorbed by the nc-info. If Jack isn't bald, then the interpreter concludes that the sentences are true without evaluating the second disjuncts, hence no new information is added to the model. Either way, (7.9c) and (7.9d) don't implicate that Jack is bald[4].

Similarly, we can explain the following Karttunen's rule for if...then, which applies to (7.10). (Note that (7.10c) and (7.10d) have been presented as paradigmatic examples as (7.4a) and (7.4b) in Section 7.1.)

(7.12) Let $S$ be "If $A$ then $B$." If $A$ "presupposes" $C$ then $S$ "presupposes" $C$ if $B$ "presupposes" $C$ then $S$ "presupposes" $C$ unless $A$ entails $C$.

[4] Karttunen (1974) mentions the possibility of symmetric condition for disjunction in the form:

(i) Let $S$ be "Either $A$ or $B$." If $A$ "presupposes" $C$ then $S$ "presupposes" $C$ unless not-$B$ (or the context) entails $C$ if $B$ "presupposes" $C$ then $S$ "presupposes" $C$ unless not-$A$ (or the context) entails $C$.

This modification would discribe sentences as (ii):

(ii) Either Jack doesn't regret being bald or Jack isn't bald.

In the EIL system, the second disjunct of (ii) is evaluated only when the first disjunct evaluates to false, i.e., in a model where Jack regrets being bald. Thus the processing of the first disjunct creates a 'presupposition' that Jack is bald, which is subsequently canceled at the time when the second disjunct is processed. A problem would arise when the first disjunct evaluates to true, in which case the second disjunct is not evaluated and thus the 'presupposition' is not canceled. To determine whether or not (ii) has the 'presupposition' that Jack is bald in the case that Jack doesn't regret being bald is a delicate judgement, about which I am not so sure. At any rate (ii) doesn't seem to be completely equivalent to (7.9c).
According to our rule (2.4m), the consequent of the conditional IMP is evaluated only after the antecedent is evaluated. Thus, if Jack is bald, the consequents of (7.10c) and (7.10d) are evaluated in a model where Jack is bald, leading to no creation of the 'presupposition' or 'conventional implicature' that Jack is bald. If Jack isn't bald, the sentences become true without the consequents being evaluated, i.e., without new information being created. Thus no implicature comes from (7.10c) or (7.10d) in this case, either.

Note that in either case, AND, OR, or IMP, the lack of a 'presupposition' or a 'conventional implicature' has been explained by one of the two reasons: either the a-info produced by the evaluation of the second conjunct, the second disjunct, or the consequent is absorbed by the duplicate nc-info in the model, or the a-info is not produced in the first place due to the lack of evaluation itself. It is not the case that an implicature is first created as c-info and then canceled, as we have seen in previous cases in Chapter 6 (cf. (6.3), (6.15), and (6.16)). This kind of explanation wouldn’t work in our system since even conventional implicatures are filtered out though they are not cancelable. The concept of absorption, however, applies equally well to 'presuppositions' and 'conventional implicatures' (cf. (6.6b) and (6.8b)).

Similar explanation is possible for subjunctives. We could deal with subjunctive conditional sentences in EIL by introducing a new logical connective, say SIMP, whose semantics would be:

\[(7.13) \text{If } p \text{ and } q \text{ are members of } L_t, \text{ then} \]
\[
V[\text{SIMP } p \land q] = \begin{cases} \{\text{if (there is a counterfactual world } x \text{ with respect to } p \text{ and } w) } \\
\quad V[q, x, t, f, n, i]; \\
\quad \text{else if } (V[p, x, t, f, n, i] = F \text{ in all } x \text{ in } W \cap UD_i) F; \\
\quad \text{else } U \}
\end{cases}
\]

In (7.13) it is assumed that there is a selection function \( h \) from the set of pairs of a formula in EIL and a possible world such that \( h(p, w) \) is a counterfactual world with respect to \( p \) and \( w \) in the sense that the only possible difference between \( h(p, w) \) and \( w \) is that \( p \) is true in \( h(p, w) \) (cf. Stalnaker 1968, Lewis 1973, and Nute 1975 for various definitions of the selection function). The point in (7.13) is that \( q \) is evaluated only in such a counterfactual world, where \( p \) is true by definition. Thus, (7.14) below doesn’t 'presuppose' that Jack is bald, either.

\[(7.14) \text{If Jack were bald, then Jack wouldn’t regret it.} \]
\[
\text{(SIMP (BALD J))} \\
\text{(NOT (REGRET J (INT (BALD J)))))}
\]
In summary, we have seen that Karttunen's observations about the projection problem necessitate no addition to our computational framework. As for "holes," creation of implicatures by pragmatic procedures are inherently cumulative since they are realized as changes to the model. Pragmatic procedures provide general and flexible mechanisms for various degrees of "plugs" as distinct classes from "holes"; in fact, any finer classes, which would be highly dependent on each lexical item, could easily be distinguished in our system. The computational semantics of logical connectives in our system and the concept of absorption based on it provide a natural explanation of the "filtering" process in terms of the way new information is added or deleted[6].

Before concluding this section, we could observe an interesting interaction between the discourse referent, which has been discussed in Chapter 3, and the implicature. The surface occurrences of it in (7.6e) and (7.6f), (7.9c) and (7.9d), (7.10c) and (7.10d), and (7.14), which have been analyzed as syntactically derived from the embedded sentences and the original formula, i.e., (Bald J), has been recovered, could be treated as something like "deep anaphora" (Hankamer and Sag 1976), and they could be translated simply into a free variable IT. Note that, by the semantic conditions of the connectives, IT would be interpreted only in the model where Jack is bald, as we have seen above. Thus, the fact that Jack is bald would be successfully picked up by the interpreter as the referent of IT.

7.3. "EXTENSION," "IMPLICATURE," AND "HERITAGE"

Having laid out the general properties of implicatures in the way we have seen in the previous section, Lauri Karttunen started

[5] That Jack isn't bald in the actual world is a 'conversational implicature' (a "clausal quantity implicature," cf. Section 5.2) of (7.14) mentioned in Section 5.2. Perhaps in the case of subjunctive, the implicature is highly conventionalized and hardly cancelable. This fact could be reflected in the definition of the counterfactual world by postulating that p is false in w, which makes the counterfactuality part of 'what is said.' To treat it as a 'conventional implicature' would be equally possible.

[6] See Kempson (1975, p.191) and Steedman (1977) for the explanation of the "filtering" process from a similar point of view. The former gives an informal discussion based on the concept of the "Pragmatic Universe of Discourse," which is similar to our UDs, and a pragmatic definition of "presupposition" based on it. The latter is closely related to the EIL system, or the concept of computation, where the status of the machine serves the equivalent role of linguistic context and the arguments of logical connectives are evaluated sequentially. In his system, the same kind of explanation given above is naturally obtained.
reformulating his theory on 'what is implicated' in his 1975 paper. The reformulation has resulted in a series of papers written with Stanley Peters (1975, 1976, 1977, 1979)[7]. The direction of the reformulation is characterized by the following aspects.

First, by adopting Montague semantics, they have made explicit what 'what is said' refers to: the 'semantic, or truthconditional aspect of meaning' is identified with 'a certain relation between the sentence and the external world' (Karttunen 1975, p.51); they identify 'Grice's notion of what is actually said with the logical form of the sentence that was uttered' (Karttunen and Peters 1975, p.267). Here, "logical forms" are "expressions of some model-theoretically interpreted auxiliary language, say, the language of intensional logic" (Karttunen 1975, p.51). Thus, as with the current system, they adopt model-theoretic semantics of natural language with translations into intensional logic as the descriptive device. As has been mentioned, the translation into an intermediate language such as intensional logic is not necessarily an integral part of Montague semantics. However, as will turn out, their system seems to rely heavily on the translation process.

The second aspect is that, in coping with 'what is implicated,' they extend Montague semantics in such a way that each linguistic expression is systematically assigned two kinds of expressions: an expression for the semantic representation in the sense mentioned above and an expression which corresponds to a pragmatic representation. That is, either each sentence has two analysis trees, one for the semantic aspect of meaning and the other for the pragmatic aspect (Karttunen 1975), or each sentence has one analysis tree which is translated into two formulas of intensional logic, one for the semantic aspect and the other for the pragmatic aspect (Karttunen and Peters 1975, 1979). Note that both the analysis tree and the formula of intensional logic belong to the level of object language, not the level of meta-language where semantico-pragmatic conditions are defined. Thus, their approach can be said to be an attempt to express the distinction between semantic and pragmatic aspects of meaning at the syntactic level. Each syntactic expression, whether translated into intensional logic or not, is assumed to play a dual role.

The third aspect of their approach is the expulsion of the traditional notion of "presupposition" from 'what is implicated' (Karttunen and Peters 1975, 1977, 1979). Traditional instances of "presuppositions" are reanalyzed as other kinds of implicatures, namely, "particularized conversational implicatures," "generalized conversational implicatures," and "conventional implicatures." Moreover, as the titles of their 1975 and 1979 papers suggest, they

[7] Karttunen and Peters' 1975 paper is absorbed and extended in the second half of their 1979 paper, which includes their 1977 paper as the first half. Their 1976 paper is an attempt to apply their approach to "conventional implicature" to questions.
treat almost exclusively "conventional implicatures" in their formal system, since, they argue, "it can be a mistake to leap from the observation that speakers know a certain fact about their language to the conclusion that this fact must be recorded in a grammar an account of their linguistic competence," and thus "conversational implicatures need not be dealt with in the grammar of language" (Karttunen and Peters 1977, pp.362-363). Since their descriptive apparatus is meant to deal with implicatures "in the grammar of language," only "conventional implicatures," which depend on particular lexical items in the grammar, have a place in their system.

Finally Karttunen (1973, 1974)'s "plugs," "holes," and "filters" have not been completely lost in the new system, though these labels are now obsolete. The first two classes are realized as the "heritage functions" associated with English phrases, which presumably have more general expressive power than the mechanism which relies on only two (or three) classes. Thus, the added generality is expected to eliminate the problems associated with these classes; "plugs" and "holes" have been known to be too restricted a characterization of the inheritance phenomenon. The "filtering" conditions are explicitly stated in the grammar as translation rules. As in PTQ, logical connectives are synchronized in a syntactically represented manner with their system and concomitantly translation rules are free to add special conditions to the pragmatic part of the translations involving these connectives. These conditions are based on Peters (1977)'s truth-conditional formulation of Karttunen's "filters." However, Karttunen and Peters offer no essentially new explanation of the "filtering" phenomenon; they remain at the same descriptive level as that in Karttunen (1973, 1974).

In the following, I will discuss two kinds of difficulties with their system. The first kind is about their terminology, which may be relatively a minor problem. As has been mentioned above, they make considerable effort to motivate their "Requiem for presupposition" (Karttunen and Peters 1977), which seems to be played without waiting for its natural death. I will discuss problems they encounter based on the feature system developed in Chapter 5. The second kind involves more substantial problems arising from their particular choice of descriptive devices.

Let us review the terminology used in our system first. According to the feature system introduced in Chapter 5, an implicature is [-said] information, i.e., the information which is communicated but does not participate in the compositional semantics. There are three kinds of implicatures: 'conversational implicatures,' 'conventional implicatures,' and 'presuppositions.' A 'conversational implicature' has features [-said,-conventional,+cancelable]. This is an implicature which is established with the aid of Grice's conversational maxims as indispensable ingredients. A 'conventional implicature,' which has features [-said,+conventional,-cancelable],
an implicature which is established solely based on the conventional usage of the lexical item without relying on the conversational maxims, and is not cancelable. The third one, namely, a 'presupposition,' with features [-said,+conventional,+cancelable], is very similar to a 'conventional implicature,' in the sense that it is an implicature which is also established solely based on the conventional usage of the lexical item without relying on the conversational maxims, but the difference is that it is cancelable. Note that conventionality and cancelability are not mutually exclusive properties due to the existence of 'presuppositions,' which are both conventional and cancelable. In other words, conventionality alone doesn't qualify a 'conventional implicature' and cancelability alone doesn't qualify a 'conversational implicature.'

Karttunen and Peters (1977, 1979)’s attempt to carefully reexamine the examples of traditional "presuppositions" is important, since, as they point out, what have been called "presuppositional" phenomena don’t constitute a homogeneous set. But, in so doing, they seem to have become too hasty in discarding "presuppositions." As the result, they seem to have stretched the concept of "conventional implicature" so much as to be used in rather uncomfortable situations and possibly somewhat devient from the original intention of Grice (1975). First, let us note that they explicitly state that "conventional implicatures are NOT CANCELABLE; it is contradictory for the speaker to deny something that is conventionally implicated by the sentence he has uttered" (Karttunen and Peters 1979, p.2, footnote 3). This statement is in accord with the traditional usage of the word "conventional implicature." However, they seem to go further than this and identify conventionality and noncancelability, which are separate concepts. In addition, they claim that the traditional "presuppositions" associated with factive verbs, such as forget, realize, take into account, etc., "are really instances of conventional implicature" (Karttunen and Peters 1979, p.11). Thus, they cannot allow the cancellation of the implicature associated with factive verbs when they are in the scope of negation and other similar operators. They have to resort to a different explanation for the apparent cancellation. Remember that we have seen that these operators have the ability to cause the cancelability of the information produced by the embedded sentence which contains a factive verb. Karttunen and Peters (1979) do give an example of apparent cancellation themselves (as their (77b), p.46):

(7.15) Bill hasn’t already forgotten that today is Friday, because today is Thursday.

The most natural explanation for the felicity of (7.15) seems to be that the implicature that today is Friday is canceled by the second half of the sentence, which is exactly the approach taken in the EIL system. As we have seen, if a factive verb is negated, ‘what is implicated’ becomes cancelable. If this kind of explanation prevails,
which I do believe, the implicature that today is Friday in (7.15) cannot be called a "conventional implicature" according to even Karttunen and Peters' own definition (as well as Grice's or EIL's). They seem to be in an awkward situation for explaining this case, since their translation rule for negation only negates the part which expresses 'what is said' and 'what is implicated' is passed on without any change[8].

As has been discussed in Chapter 5, the determination of what kind of information belongs to 'conversational implicatures' and what else to 'conventional implicatures' is not straightforward. No single criterion establishes membership. Furthermore, the case of factive verbs is rather intricate since both cancelable and noncancelable kinds of information are produced depending on the environment of the verb. To call one 'conversational' and the other 'conventional' would be schizophrenic since the Gricean maxims should apply either to both cases or to none of them. In this case, they are both clearly conventional (Karttunen and Peters are right in this judgement), but

[8] Near the end of the paper, Karttunen and Peters (1979, p.46ff) do propose a separate negation rule motivated by the "external negation" in three-valued logic -- a "contradiction negation" rule -- which is supposed to explain cases like (7.15). Since their logic is bivalent, the "contradiction negation" cannot be exactly the same as the "external negation." According to their definition, if A is a "contradiction negation" of B, then 'what is said' of A is the negation of the conjunction of 'what is said' of B and 'what is implicated' of B, while 'what is implicated' of A is the disjunction of 'what is implicated' of B and the negation of 'what is implicated' of B, i.e. a tautology. Although they claim they have some linguistic evidence for this, they say they don't include this rule in the main body of the framework due to the lack of formal characterization of such negation in their formal system. Even if this "contradiction negation" can somehow be formally treated, their characterization of it is far from satisfactory. Since the implicature of a contradictory negation is a tautology, their analysis predicts that the first half of (7.15) implicates a tautology, or equivalently, implicates no new information. This is contrary to the property of the first half of (7.15), since if the second half does not follow, we do get an implicature that today is Friday. The first half of (7.15) cannot be neutral; it somehow commits the utterer to the proposition that today is Friday, not that either today is Friday or today is not Friday. Without a formal way to tell why the negation in (7.15) is an ordinary one if the second half does not follow, and is a contradictory one if the second half does follow their treatment of (7.15) reveals very little about the information communicated by (7.15). Their treatment of negation suffers the same kind of criticism as that given to the distinction of "external" and "internal" negations in three-valued logic. See Gazdar (1979a, p.110ff) for more criticism of Karttunen and Peters' treatment of negation.
we have to admit the existence of a cancelable kind of implicature which is communicated conventionally not conversationally. This is what has been called a 'presupposition,' which is different from both 'conventional implicature' and 'conversational implicature' and marked as [-said,+conventional,+cancelable]. Karttunen and Peters's use of "conventional implicature," thus, has the danger to obscure the important distinction between cancelable and noncancelable kinds of information; the "conventional implicatures," as long as it is defined in the standard way including the property of noncancelability, cannot cover all the aspects of the implicatures associated with factive verbs. They have expected too much of "conventional implicatures." Note that in the present system, a 'presupposition' is expected to exist rather than it is added as a totally new concept since the feature system developed in Chapter 5 gives exactly four possible combinations of the three features (cf. (5.13)). In other words, having a 'presupposition' is no additional burden to the system. On the other hand, Karttunen and Peters's system has to postulate a different kind of negation (in fact, of the whole class of operators which are capable of inducing cancelability, i.e., ci-ops) to explain the cancelability. In the following, I will call what they mean by a "conventional implicature" simply an implicature, since it can cover both 'presuppositions' and 'conventional implicatures.'

The problems with Karttunen and Peters's system is not limited to mere terminology. In their system of the 1979 version, the syntactic and semantic theory of English based on intensional logic is modified in a special way; instead of extending the semantics of intensional logic to incorporate pragmatic aspects, which is done in the EIL system, they essentially maintain the syntax and the semantics of intensional logic, but extend the syntax of English by associating two translation rules with each syntactic rule. The first translation, which is called the "extension expression," is the usual translation which corresponds to 'what is said.' The second, the "implicature expression," corresponds to 'what is implicated' and is composed of three kinds of elements --- extension expressions of the constituents, implicature expressions of the constituents, and the "heritage function" which is associated with the constituents based on the syntactic rule[9].

For a simple example, (7.16a) below is translated into the pair of (7.16b), the extension expression of (7.16a), and (7.16c), the implicature expression of (7.16a)[10].

(7.16) a. Jack walks.
   b. (e_JACK (INT e_WALK))
   c. (AND (1_JACK (INT 1_WALK)) (h_JACK (INT h_WALK)))

In general, if there is a functional application rule which produces the extension expression of the form (7.17a) below the implicature
expression corresponding to it is (7.17b).

\[(7.17)\]
\[
a. (e \_ p (\text{INT} \ e \_ q)) \\
b. (\text{AND}^* (i \_ p (\text{INT} \ e \_ q)) (h \_ p (\text{INT} \ i \_ q)))
\]

where AND* takes appropriate expansion depending on the type of the conjuncts to make a well-formed formula as follows.

\[(7.18)\]
\[
a. (\text{AND} \ p \ q) \text{ if } p \text{ and } q \text{ are of type } t \\
b. (\text{LAMBDA} (X) (\text{AND} \ ((p \ X) \ (q \ X)))) \text{ if } p \text{ and } q \text{ are of type } \langle e, t \rangle \\
c. (\text{LAMBDA} (P) (\text{LAMBDA} (X) (\text{AND} \ ((p \ P) X) ((q \ P) X)))) \text{ if } p \text{ and } q \text{ are of type } \langle e, t \rangle, \langle e, t \rangle \\
e. etc.
\]

In the role which the schema (7.17) is intended to play, it is essentially the same formulation as the cumulative hypothesis (7.2) for the case where \( p \) is a "hole," i.e. if \( h \_ p \) is a function which simply gives the extension of the argument with the re-interpretation of "assertion" in (7.2) as "extension" here and "presupposition" in (7.2) as "implicature" here. In this sense, Karttunen and Peters's (7.17) is very close to EIL's (7.3), also. As we have seen in (7.3), the cumulative hypothesis, within the limit of its effectiveness, is naturally realized in the EIL system by the use of PREC and POSTC procedures. Moreover, I have argued that Karttunen (1973, 1974)'s "plugs" and "holes" are more flexibly realized by pragmatic procedures. Thus, as in EIL, Karttunen (1973, 1974)'s "plugs" and "holes" are expressible in their new framework by defining the heritage functions appropriately. It is interesting to note that in addition to the fact that Karttunen and Peters's extension expressions correspond to, naturally, EIL's DEN procedures, their implicature expression corresponds to EIL's PREC procedures, since only 'conventional implicatures' (and 'presuppositions'), for which PREC procedures are responsible, are considered in their system, and their heritage functions correspond to EIL's POSTC procedures. Thus, (7.17)

\[\text{[9]}\] Originally, Karttunen and Peters (1975) used a translation into a triple of expressions -- extension, implicature, and heritage expressions. The role of the third expression has been reduced in their 1979 version and only occasionally appears as an appendix to some of the translation rules as the value of the heritage function.

\[\text{[10]}\] Karttunen and Peters use superscripts to distinguish three kinds of expressions. In writing their formulas in the EIL notation, extension, implicature, and heritage expressions are prefixed by \( e_\_ \), \( i_\_ \), and \( h_\_ \), respectively. Note that \( e_\_ \text{JACK} \) is of type \( \langle e, t \rangle, t \rangle \), i.e., a set of properties. In the previous formulas in EIL, this type of constants have been expanded as \( (\text{LAMBDA} (P) ((\text{EXT} P) I)) \), etc., in order to get a formula like \( (\text{WALK} \ I) \) from a formula like \( (\text{JACK} (\text{INT} \ \text{WALK})) \).
can be considered as a counterpart of (7.3) in the EIL system, which
is one way of interpreting the cumulative hypothesis (7.2).

However, the similarity ends exactly here. One of the major
differences between the two systems is that, while (7.3) is merely a
summary of the behavior of the interpreter concerning the interaction
of the procedures and hence independent of each syntactic
construction, Karttunen and Peters have to introduce essentially the
same translation of the form (7.17b) each time they give a translation
to a rule of functional application (thus, in the Appendix of
Karttunen and Peters (1979), they repeat essentially the same
implicature expression seven times for Rules 4 through 10). This is
because they base their system on the syntactic level; instead of
giving a single syntactic expression and making the semantics more
powerful, they produce double syntactic structures (i.e., double
translations) while maintaining the existing semantics. What I have
argued so far in the previous chapters is that by letting the
semantico-pragmatic component take care of the integration of the
semantics and the pragmatics, such as creating implicatures in
addition to computing denotations, we can capture a wider variety of
generalizations while maintaining a straightforward surface-based
syntax and an essentially compositional semantics of natural language.
On the other hand, what Karttunen and Peters attempt in their system
is not in the direction toward a more refined semantico-pragmatics
but in the direction of a more complicated syntax (translation rules)
while keeping the traditional semantics intact; in fact, their
extension of PTQ is essentially that of a set-theoretical model
theory; it has no provision for a dynamic interpretation mechanism of
the interaction between the utterance and the discourse.

This defect of their system might be amended by making the three
expressions not syntactic elements but semantico-pragmatic elements --
call them "extension condition," "implicature condition," and
"heritage condition," say. A semantico-pragmatic rule for functional
applications of the form $(p\ (\text{INT}\ q))$ would then look like:

\begin{align*}
(7.19) & \quad \text{Let } e[p], i[p], \text{ and } h[p] \text{ be the extension, the implicature,}
& \quad \text{and the heritage conditions of a syntactic expression } p. \\
& \quad \text{Then} \\
& \quad \text{a. } e[(p\ (\text{INT}\ q))] \text{ is the result of applying } e[p] \text{ to the}
& \quad \text{intension of } e[q], \\
& \quad \text{b. } i[(p\ (\text{INT}\ q))] \text{ is the conjunction of the result of}
& \quad \text{applying } i[p] \text{ to the intension of } e[q] \text{ and the result of}
& \quad \text{applying } h[p] \text{ to the intension of } i[q].
\end{align*}

Naturally, this amended system becomes more indistinguishable from the
part responsible for implicatures in the EIL system if one substitutes
the DEN procedure of $p$ for $e[p]$, and the pragmatic procedures of $p$ for
$i[p]$ and $h[p]$. 
However, even (7.19) does not always produce the desired result; their system leads to the open problem mentioned in their Note (Karttunen and Peters 1979, p.53). They have the difficulty in establishing coreferentiality between the extension expression and the implicature expression. For example, consider (7.20). Let us assume that (7.20b) is the extension expression of (7.20a), and (7.20c) is the implicature expression of (7.20a)[11].

(7.20) a. A man regrets being bald.
   b. (SOMEx AND (MAN X))
      (REGRET X (INT (Bald X)))))
   c. (SOMEx AND (MAN X) (Bald X))

The problem is that, so long as (7.20b) and (7.20c) are separate formulas and evaluated separately, there is no way to assure that the man who regrets being bald is the same person who is implicated to be bald. There seems to be no easy way to solve this problem within their system. This is, however, not a problem in EIL at all since, as we have seen, there is only one formula, which has the form of (7.20b), as the translation of (7.20a). Note that the semantic (DEN) and the pragmatic (PREC) procedures are executed in the middle of the compositional computation of the denotation of the whole formula. Thus, when the PREC procedure attached to REGRET is executed, the variable X has been bound to the appropriate person. Since the PREC procedure creates an implicature that the person denoted by X is bald with the current referent of X, the man who is bald is necessarily the man who regrets his baldness. (Cf. Appendix D for a step-by-step trace of the evaluation of this formula.) As Karttunen and Peters may have noticed in the Note this is rather a methodological problem arising from their particular decision to associate two independent propositions to a single surface sentence[12].

[11] (7.20c) would be obtained by defining i_REGRET as (LAMBDA (PR) (LAMBDA (X) (EXT PR))). Although other expressions may be conjoined to (7.20c), they are ignored here since the exact form of the implicature expression is immaterial; only the existence of the existential quantification is relevant for the discussion here. The prefix e_ is omitted in (7.20).
[12] Weischedel (1979) describes a computational system where the parser produces dual expressions which correspond to the semantic representation and the "presupposition" of the sentence (actually, a third expression -- the "entailment" -- is also produced). Unless the coindexing of variables is forced by some additional mechanism, which is apparently done by assigning the same index to the variable, his system would have a similar problem as Karttunen and Peters's. Note that in the EIL system, the parser produces only one representation. Creation of 'presuppositions' is the task of the semantico-pragmatic interpreter, not the syntactic parser.
Another problem concerns itself with the treatment of "filtering" associated with connectives. For example, conjunction of the form (7.21a) below is associated with the implicature expression (7.21b).

(7.21) a. (AND e_p e_q)
    b. (AND i_p (IMP e_p i_q))

As Peters (1977) and Karttunen and Peters (1979) argue, (7.21b) achieves the same effect of Karttunen (1973, 1974)'s "filtering" rule (7.7). Thus, they do have the provision for "filtering." But (7.21b) itself is not explained. Their arguments in support of the form (7.21b), or any possible revision of it, remain an informal statement of the generalizations observed in the phenomena; they don't present any necessary formal connection between (7.21a) and (7.21b). In the EIL system, however, as we have seen in Section 7.2, (7.21b) is derived from the semantics for (7.21a), namely from the fact that, in (7.21a), the first conjunct is evaluated before the second. The fundamental assumption in our system has been that, unlike a truth-functional logic, the processing of conjunction by human (and artificial) beings is sensitive to the left-to-right order of the conjuncts, which affects at least 'what is implicated,' if not 'what is said.' Karttunen and Peters's system, which is based on truth-functional semantics, cannot capture this aspect of pragmatics and has to postulate rules like (7.21b) as separate rules from the semantics.

In summary, what is missing in Karttunen and Peters's system is a utilization of dynamics when utterances are interpreted by the hearer. Cancellation of some of the implicatures is a typical example of such a process; thus the lack of implicature in (7.15) above can be naturally explained by the concept of cancellation of the implicature in the interaction of the utterance with the context rather than by postulating a negation of a different kind whose validity is somewhat dubious. In the next section, we will examine a system based exactly on this concept of cancellation, namely, that by Gazdar (1979).

7.4. "IMPLICATURE" AND "PRE-SUPPOSITION"

As we have seen in the previous section, Karttunen and Peters's system is constructed on a compositional basis. The implicature of the higher construction is computed using the extension of the lower constructions; e.g., for functional application the implicature expression has the form (i_p (INT e_q)). The cumulative aspect of the implicatures is only reflected in the heritage function of the higher construction, which takes the implicature of the lower constructions. The implicatures computed in this way will be recursively used by still higher constructions. In the EIL system, PREC and POSTC procedures are attached to lexical items at the EIL level and accordingly evaluated compositionally along with DEN procedures. The
results of these pragmatic procedures are, however, not compositionally inherited to the higher constructions. They are inherited rather as environmental changes caused in the model by these procedures, which only indirectly affect the evaluation of higher constructions. Thus, these pragmatic procedures cause cumulative effects, even though they are evaluated on a compositional basis.

Gerald Gazdar has constructed a totally cumulative system, which was formulated in 1976 as his dissertation and was published in 1979 (Gazdar 1979a)[13], the excerpt of which concerning technical aspects was published also in 1979 (Gazdar 1979b).

His mechanisms to create implicatures are global in the sense that they take the entire utterance as an input. There is no compositional element in his mechanisms to treat implicatures. This has a rather undesirable consequence since his pragmatic mechanisms seem to work independently of the semantics which he assumes to be a Fregean/Montaguean compositional one. As we have seen in Karttunen and Peters's system or in the EIL system, the implicature may depend on the denotations and/or implicatures of the lower constructions (cf. (7.17) for Karttunen and Peters's system in the previous section and the recursive definition (6.2) for the EIL system in Section 6.2). Even in Gazdar's system, the definition of potential implicatures depend on the concept of entailment. Moreover, as will be mentioned, in the case of factive verbs in affirmative sentences, he assumes that the complements are both entailed and "presupposed." Thus, he has to rely on two separate mechanisms to explain an apparently unitary phenomenon; one in the semantics as to the entailment and the other in the pragmatics as to the "presupposition." Note that, in EIL, this case has been treated as 'conventional implicatures," which are created by the same pragmatic mechanism to create 'presuppositions.' namely, the PREC procedure attached to factive verbs. I will argue the inadequacy of the entailment analysis later.

Another aspect of his system is that not only does he admit the existence of pragmatic "presuppositions," but the distinction between Gricean "implicatures" and his pragmatic "presuppositions" is given a crucial role in coping with the projection problem. His mechanisms produce and process potential "implicatures" and potential

[13] Gazdar (1979a) contains well-organized and detailed arguments for or against traditional treatments of "presuppositions" and Gricean "implicatures." I have cited his work in several places in this dissertation during the description of the EIL system whenever I have thought it appropriate. In this section, I will concentrate on the technical and methodological aspects of his system; I have little to say concerning his terminological or philosophical aspects. Some methodological problems concerning his treatment of cancellation will be discussed in detail. See Stalnaker (1980) for a brief review and some criticism of Gazdar (1979a).
"presuppositions" separately in a specific order; when they conflict the order of processing becomes significant to obtain the correct prediction. As I have pointed out concerning Karttunen and Peters’s system, to treat everything as "conventional implicatures" is not satisfactory. In this sense, Gazdar’s system, as well as the current one, can escape this kind of difficulty. Moreover, since he has laid carefully organized arguments against the traditional concept of "presupposition," his "presuppositions" are relatively restricted, and seem less controversial.

His "implicatures" are also restricted to only "conversational implicatures," particularly those based on the maxim of Quantity. Thus, somehow "conventional implicatures" have evaded the formal treatment in his system. He only suggests a linguistic treatment in which the dictionary entry for such a lexical item that bears a "conventional implicature" would have "some pragmatic component that would specify its implicature potential" (Gazdar 1979a, p.38), and mentions Karttunen and Peters’s system as an example, although he points out their confusion about "conversational" and "conventional" implicatures. The treatment of 'conventional implicature' in the EIL system is exactly as Gazdar has suggested, without the problem of Karttunen and Peters.

As for terminology, I will use terms presupposition and implicature (without quotes) in Gazdar’s sense in the rest of this section. Note that the word implicature will have a narrower sense in the following than the usage of this word so far in this dissertation, i.e., [-said] information, or 'what is implicated,' which has included 'presupposition.' In the rest of this section, an implicature means a 'conversational implicature.'

As we have seen, the cumulative hypothesis with its original form encounters the projection problem. Thus, an essential component in Gazdar’s system is the "cancellation" mechanism. In his system, only potential implicatures and potential presuppositions are created in the beginning, which he calls "im-PLICATURES" and "pre-SUPPOSITIONS," respectively. These im-PLICATURES and pre-SUPPOSITIONS will subsequently go through the cancellation processes and only those which have survived these processes are called genuine implicatures and presuppositions. (I will call this process as dehyphenation.) In this sense implicatures and pre-suppositions play a similar role as EIL’s a-info; both exist solely for technical reasons.

Note that Gazdar doesn’t have a counterpart of our c-info. This will give us the following distinction between the two systems. As has been mentioned above, his pragmatic components consider only a single sentence; they work only intrasententially. Accordingly, cancellation of im-PLICATURES and pre-SUPPOSITIONS is restricted within a single sentence. Depending on his definition of a sentence, which is not clear in his presentation, his theory would have to make
different predictions to (7.22a) and (7.22b) below if the former were treated as one sentence and the latter as two; the pre-supposition that Jack is bald would be canceled in (7.22a), but not in (7.22b), according to his theory, which is counterintuitive.

(7.22) a. Jack doesn’t regret being bald, in fact, he isn’t bald.
    b. Jack doesn’t regret being bald. For he isn’t bald.

Even if he manages to treat (7.22b) as a single sentence and succeeds in canceling the pre-supposition also in (7.22b), I have no idea how he would explain sentences like (7.23):

(7.23) Jack doesn’t regret being bald. I can show that quite easily. Look at the man in the corner. He is Jack, though it’s a bit too dark to see him clearly. But look at him carefully. Now you know what I mean. He isn’t bald; he only regrets being gray-haired.

Unlike Gazdar’s implicatures and pre-suppositions, as I have argued in Chapter 6, a-info in the EIL system can be transformed into c-info, which remains in the model and has the property of cancelability even after it is put in the context, though there might be gradual decay of the degree of cancelability (cf. Section 6.4). That is, unlike a-info, c-info can be canceled inter-sententially. Thus, even if (7.22b) is treated as two sentences, the ‘pre-supposition’ can be canceled since it is c-info. (7.23) is simply an extreme case of such a delayed cancellation. Since Gazdar’s system gives every bit of information the same status (i.e., noncancelable) once it is added to the context (either entailed, presupposed, or implicated), inter-sentential (backward) cancellation of implicatures is not possible in the scope of his system. (Note, however, that abortion, which might be called (either inter-sentential or intra-sentential) "forward cancellation," is properly treated in Gazdar’s system.)

Let us examine the technical details concerning how Gazdar’s formalism can give a solution to the projection problem. As for pre-suppositions, Gazdar admits at least three kinds: the factive pre-supposition, i.e., the truth of the proposition which corresponds to the complement of a factive or semifactive verb, the existential pre-supposition, i.e., the existence of the referent of the nominal after the, and, finally, the aspectual pre-suppositions, i.e., the occurrence of the fact described by the sentence after before (Gazdar 1979a, p.125ff). Needless to say, these are only samples to make the discussion concrete and he welcomes any reasonable addition. Since the pre-supposition by the affirmative factive verb is noncancelable, he assumes that a sentence with an affirmative factive verb both entails and pre-supposes the complement (Gazdar 1979a, p.119). For example, in (7.10c) and (7.10d), which are repeated here as (7.24a) and (7.24b) below the consequent of (7.24a) only pre-supposes (7.24c), while the consequent of (7.24b) both entails and pre-supposes
(7.24) a. If Jack is bald, then he doesn't regret being bald.
b. If Jack is bald, then he regrets being bald.
c. \( K(\text{Jack is bald}) \)

where \( K \) is an operator intuitively meaning "the speaker knows that" (Hintikka 1962). In (7.24a) and (7.24b), his mechanism works in such a way that the im-plication of the sentences cancels this pre-supposition, so that no presupposition comes from (7.24a) or (7.24b). I will come back to (7.24) later after describing im-plicatures and the context of utterance.

As for im-plicatures, there are two kinds: scalar quantity im-plicatures and clausal quantity im-plicatures (Gazdar 1979a, p.58ff). The former is based on a quantitative scale mentioned in Section 5.1. For example, some is "weaker" than all in one of such scales and his rule produces from a sentence containing some an im-plication which denies the case for all. Thus, (7.25a) below im-plicates (7.25b).

(7.25) a. Some men are bald.
b. \( K\neg (\text{all men are bald}) \)

where \( \neg \) is a negation operator. The clausal quantity im-plicature is more complicated. Basically, given a matrix sentence \( M \), a subsentence \( S \) of \( M \) and \( S \)'s negation are im-plicated if (1) neither \( S \) nor its negation is entailed by \( M \) and (2) \( S \) does not occur in a position of presupposition with regard to \( M \) where a "position of presupposition" is a position in a sentence such that, if some other string is substituted in that position, it becomes a pre-supposition of the sentence. Thus, (7.26a) below im-plicates (7.26b) through (7.26e).

(7.26) a. If baldness is hereditary, then Jack is bald.
b. \( P(\text{baldness is hereditary}) \)
c. \( P\neg (\text{baldness is hereditary}) \)
d. \( P(\text{Jack is bald}) \)
e. \( P\neg (\text{Jack is bald}) \)

where \( P \) is an operator intuitively meaning "it is possible, for all the speaker knows, that" (Hintikka 1962). As can be seen from this intuitive specification, \( P\neg \) is equivalent to \( \neg K \).

Condition (1) of the definition of the clausal quantity im-plicatures rules out both conjuncts of a conjunction or their negations as im-plicatures of this type, since both conjuncts are entailed by the matrix sentence. Thus, (7.27) below doesn't im-plicate (7.26b) - (7.26e).

(7.27) Baldness is hereditary and Jack is bald.
Condition (2) rules out the complement of the factive verb in sentences such as (7.28) as an im-plicature, since it occurs in a position of presupposition.

(7.28) Jack regrets that he is bald.

Condition (2) is necessary to prevent unwelcome cancellation of pre-suppositions by im-plicatures, since, as will be seen below, his im-plicatures are capable of canceling pre-suppositions. 'What is implicated' by (7.28) is a presupposition in his system, so the second condition is required to suspend an im-plicature which would cancel the pre-supposition. Note that (7.24a) and (7.24b) above im-plicate that Jack is bald, P(Jack is bald), among others, due to the antecedent, not the consequent, since the embedded position in the consequent is a position of presupposition, while that in the antecedent is not.

Another essential device in Gazdar's system is the "context of utterance," which is a set of propositions. Each utterance increments the old context in the following way. First, the proposition that the speaker knows the utterance is added to the old context. This corresponds to the super-interpreter's addition of new information for the indicative mood in our system. Second, the clausal quantity im-plicatures are added so long as they are consistent with the existing context. The precise definition of this "add-if-consistent" operation, which he calls the "satisfiableincrementation," and is said to be "the single most important definition in the present theory" by him (Gazdar 1979a, p.131), is rather complicated. Informally a member of the set of clausal quantity im-plicatures produced by the utterance is added to the existing context if, for every consistent subset of the union of the existing context and the whole set of im-plicatures, adding that im-plicature to that subset doesn't jeopardize the consistency of the subset. Thus, an im-plicature which contradicts with some member of the existing context is not added, nor are mutually contradicting im-plicatures, among others. After the clausal quantity im-plicatures are added, the same operation is repeated for the scalar quantity im-plicatures with the result of the satisfiable incrementation of the old context by the clausal quantity im-plicatures as the new context. Finally, the pre-suppositions are added to the context with the result of the above two steps as the new context. The final result acts as the new context for the next utterance. As can be seen, his satisfiable incrementation achieves similar tasks as those in the EIL system which have been called abortion (deletion of a-info due to the existence of contradictory nc-info in UDs) and creation (addition of a-info to UDs by changing it to c-info). What is missing in his system is the concept of absorption (deletion of a-info due to the existence of duplicate nc-info in UDs), which will be essential to explain some of the data below (cf. (7.35)).
The ordering of addition mentioned above -- clausal quantity im-plicatures first, then scalar quantity im-plicatures. finally pre-suppositions -- is completely extrinsic and, as Gazdar himself admits, "is not itself explained" (Gazdar 1979a, p.132). This ordering is necessary in his system because, as far as clausal quantity im-plicatures and pre-suppositions are concerned, it plays a crucial role in the explanation of the "filtering" phenomenon. Let us go back to (7.24) to see this point. In Gazdar's system, (7.24a) im-plicates (7.29a) through (7.29d) below.

(7.29) a. P(Jack is bald)
    b. P~(Jack is bald)
    c. P(Jack doesn't regret being bald)
    d. P~(Jack doesn't regret being bald)

(7.24a) pre-supposes on the other hand, (7.24c), which is repeated here as (7.30) below.

(7.30) K(Jack is bald)

Since (7.29b) is added to the context first, (7.30) is rejected at the time of adding pre-suppositions, since (7.29b) is equivalent to (7.31) below, which is the negation of (7.30) and hence not consistent with (7.30).

(7.31) ~K(Jack is bald)

In Gazdar's system, if a pre-supposition is not added to the context, it is simply canceled, or deleted, and is not dehyphenated. The same explanation applies to (7.24b). Note that, on the other hand, it doesn't apply to (7.10a), which is repeated here as (7.32):

(7.32) If baldness is hereditary, then Jack doesn't regret being bald.

(7.32) im-plicates the following.

(7.33) a. P(baldness is hereditary)
    b. P~(baldness is hereditary)
    c. P(Jack doesn't regret being bald)
    d. P~(Jack doesn't regret being bald)

Jack's baldness is not im-plicated since the only position in which the proposition that Jack is bald occurs is a position of presupposition. Thus, there is no im-plicature to cancel the pre-supposition of (7.32), which is (7.30), and (7.30) becomes a presupposition. By the same mechanism, (7.10b) presupposes (7.30). The same kind of explanation as for (7.24) applies to disjunction too, since, for example, (7.9c), which is repeated here as (7.34) below, also im-plicates (7.29a) through (7.29d).
Either Jack isn't bald or he doesn't regret it.

Thus, Gazdar's system has the virtue of eliminating the need for separate "filtering" rules for different connectives. However, his apparently general explanation itself is defective from at least two aspects. First, he has to rely on the unexplained ordering of the additions of implicatures and presuppositions. Recall that Karttunen's "filtering" rule eliminates a possible presupposition of the consequent of a conditional if the antecedent entails it. This phenomenon has been explained in the EIL system based on the left-to-right order of the processing of the conditional, i.e. by the fact that, if the antecedent entails the possible presupposition, and if the antecedent is true, then the possible presupposition is absorbed by the entailment, since there is no need for any kind of duplicate "supposition"; if, on the other hand, the antecedent is false, then the utterance is judged to be true without interpreting the consequent which would have produced the presupposition. Gazdar's explanation of the phenomenon is less revealing, since it relies on, in addition to the deliberate choice of P and K operators so that the cancellation mechanism works (note that to give K to pre-suppositions and P to clausal quantitiy implicatures has been crucial in his cancellation mechanism), the unexplained ordering of the additions of implicatures and pre-suppositions. Note that if the pre-supposition (7.30) were added first, it would cancel the implicature (7.29b) since they are mutually contradictory. This would only mean that (7.24a) doesn't implicate that Jack isn't bald, which is not exactly what is wanted. Thus, this extrinsic ordering is crucial in Gazdar's system to obtain the desired result predicted by Karttunen's "filtering" rule.

Secondly Gazdar's system doesn't cover all the cases of Karttunen's "filters." Note that we have only seen so far that conditional and disjunction are given a uniform treatment. As for conjunction, which has been given the same "filtering" rule as for conditional by Karttunen, Gazdar's mechanism gives a wrong prediction. Note that (7.6e), which is repeated here as (7.35) below, doesn't implicate any of (7.29).

(7.35) Jack is bald and he doesn't regret it.

As has been mentioned, by virtue of Condition (1) for the clausal quantity implicatures, none of (7.29a) through (7.29d) are implicated since they or their negation are entailed by (7.35). Thus, apparently there is no implicature available to cancel the pre-supposition that Jack is bald. Moreover, since K(7.35), which entails (7.30), is added to the context first, the pre-supposition is consistent with the context (unless the utterance (7.35) itself is inconsistent with the context) and should become dehyphenated, i.e., should be added to the context as a presupposition in his system. But, as Karttunen's rule and our intuition tell us, (7.35) doesn't "presuppose" that Jack is
bald in any sense. That is, the pre-supposition that Jack is bald should be deleted. Note the difference between (7.36a) below which "presupposes" that Jack is bald, and (7.36b), which doesn't.

(7.36) a. It is possible that Jack doesn't regret being bald.
 b. It is possible that Jack is bald and that he doesn't regret being bald.

If the possible presupposition is not deleted in the conjunction in the scope of the modal operator, it should surface as a presupposition because modals are "holes" for presuppositions, which is the case in (7.36a) and is not the case in (7.36b). If it is absorbed by the context (i.e., deleted) and only the entailment remains, it does not surface since the modal operator blocks the entailment, which is the case in (7.36b). Thus, the possible presupposition should be absorbed by the context as the treatment in the EIL system predicts. There is, however, no place for pre-suppositions to be absorbed in Gazdar's system. Note that the absorption of implicatures has been deliberately implanted in his mechanism for creating clausal quantity implicatures since Condition (1) explicitly rejects any entailment as an implicature. He cannot, however, add a similar condition to the definitions of pre-suppositions, i.e., to explicitly eliminate entailments from pre-suppositions, since positive factive verbs are supposed to both entail and pre-suppose the complements to explain the noncancellability.

Thus, Gazdar's cancellation mechanism is defective in that it fails to offer consistent explanations to all of Karttunen's "filtering" rules; he has trouble in treating the absorption and the abortion of pre-suppositions and implicatures with equal ease, in addition to the lack of intersentential (backward) cancellation mechanism and need for an extrinsic ordering. His mechanism based on the satisfiable incrementation explains only part of the phenomena which I have called abortion and (intersentential) cancellation[14].

In summary, Gazdar's system has no sensitivity to the internal structure of the linguistic expression analyzed in his system, including the semantic representation on which he bases his system. "Filtering" effect is in most cases nothing but the by-product of the left-to-right ordering of the interpretation processes. A system which lacks this kind of concept of ordering, which presumably reflects part of human way of doing things with words, has to state the phenomenon as independent rules, as Karttunen and Peters do, or introduce a different, unexplained kind of ordering, as Gazdar does.
There may be a way of amending Gazdar’s system to incorporate absorption. According to him, a (dehyphenated) presupposition is a member of the intersection of the new context and the set of pre-suppositions (Gazdar 1979, p.133). If the set of old context and the utterance itself are subtracted from this intersection, then only the pre-suppositions which are consistent with the new context but are not entailed by the previous context or the utterance will remain and they may be called presuppositions. This will, however, lead to another problem. Since an affirmative factive verb is assumed to both entail and pre-suppose the complement in Gazdar’s system, and an entailment and a presupposition cannot co-exist in the amended system, such a sentence will only entail the complement. But this is clearly a kind of ‘what is implicated’ since it behaves cumulatively rather than compositionally as has been seen in (7.1), with which we started this chapter. Perhaps he will have to abandon the idea that affirmative factive verbs entail the complement, if the sole motivation for that is the explanation of its noncancelability, and introduce a "conventional implicature" for such a case. The resultant system, naturally, very much resembles the EIL system.
8. CONCLUDING REMARKS

In this paper, I have presented a computational framework to augment a formal semantics so that it can handle several aspects of pragmatics uniformly. The very basic idea behind the whole approach is that the semantico-pragmatic interpreter executes three procedures in sequence with only the second contributing to the recursive determination of the denotation. The other two procedures directly work on the model and manipulate information in the model, which in turn will affect the calculation of the denotation at the current and the following stages. Thus, the semantic procedure -- the DEN procedure -- behaves compositionally to take care of 'what is said,', i.e., [+said] information, while the pragmatic procedures -- the PREC and POSTC procedures -- behave cumulatively to take care of 'what is implicated,' i.e., [-said] information. The overall control of these procedures are performed by the recursive interpreter; even the cumulative pragmatic components are embedded in a larger compositional semantic component.

The bridge to connect the compositionality and the cumulativity is the data base on which the above procedures work. By introducing hierarchical universes of discourse (UDs), both discourse referents and implicatures have received a uniform treatment as various kinds of information added to UDs. Pragmatic use of pronouns are possible only when the discourse referents have been put in UDs by the processing of the previous and/or the current utterances. The addition and deletion of information is not infrequent in processing utterances, and these phenomena, which have been termed as abortion (deletion of temporary information due to contradiction), absorption (deletion of temporary information due to duplication), or creation (addition of new information due to no contradiction or duplication), give clear explanations to pragmatic aspects of natural language. Cancellation and consolidation of 'presuppositions' and 'conversational implicatures' are intersentential and backward counterparts of abortion and absorption, where a piece of information added to the context later has the right to delete an older piece of cancelable information due to contradiction (for cancellation) or duplication (for consolidation). The feature [cancelable] has played a crucial role in describing the behavior of cancelable information in the above situations.

Grice's maxims and the concept of conversationality based on those maxims have provided another dimension in our taxonomy of
communicated information by an utterance, which results in four classes. A 'denotation' is conventionally communicated information which is directly said and hence is not cancelable, i.e., [+said,+conventional,-cancelable]. Among [-said] information, i.e., 'what is implicated,' a 'conventional implicature' is equally conventionally communicated and is not cancelable, i.e., [-said,+conventional,-cancelable]. Factive verbs not in the scope of negation or similar operators typically produce a 'conventional implicature.' A 'presupposition' is also conventionally communicated but is cancelable, i.e., [-said,+conventional,+cancelable]. It is typically produced by factive verbs in the scope of negation, etc. The fourth and remaining class, a '(generalized) conversational implicature' is not conventional but "calculated" based on Grice's maxims, and hence cancelable, i.e., [-said,-conventional,+cancelable].

We have seen the exclusive interpretation of or as a typical example. DEN procedures take care of 'denotations,' while PREC procedures are responsible for producing 'conventional implicatures' and 'presuppositions,' which are both [-said +conventional], and POSTC procedures are in control of 'conversational implicatures,' as well as the manipulation of temporary information produced by PREC procedures. In this way, we can achieve the four-way distinction -- 'denotations,' 'presuppositions,' 'conventional implicatures,' and 'conversational implicatures' -- by the use of two kinds of mechanism -- PREC/DEN/POSTC procedures, and cancelable/noncancelable information.

There are several aspects which are intentionally left out from the relatively confined range of pragmatics treated in this dissertation, not to speak of the other varieties of language use in general. One of them concerns with the syntactic analysis. Although the description of the parsing of English sentences has been minimal in this dissertation, one of the obvious difficulties which will be encountered in the future when the system is expanded is the determination of the mood from the surface structure. As is notorious, "indirect speech acts" (Searle 1975) are sometimes too slippery to capture. In addition to the fixed syntactic types (e.g., subject-auxiliary inversion for interrogatives), some linguistic forms, e.g., will you, can I, etc., are conventionally used to perform particular illocutionary acts, but the majority of the cases are ambiguous without the help of the context and the native speakers' intuition plus their common knowledge concerning the culture in which the participants in the communication live. It may be the case that we would have a similar kind of variety of direct and indirect speech acts in the sense that we have the variety of communicated information such as 'denotations,' 'conventional implicatures,' and generalized and particularized 'conversational implicatures.' This is a complicated matter which may have to wait future research for mechanization, as Grice's "particularized conversational implicatures," which are another intentional omission from this dissertation, are difficult to mechanically manipulate.\[1\].
In this dissertation, I have favored informal description about the behavior of the mechanism of EIL. It might not be impossible to completely formalize what is achieved in this dissertation using only the mathematical terms, e.g., Turing machines. However, as has been discussed in Chapter 3, the modern digital electronic computer does not offer any miracle as computing machinery; its most significant utility is its relative ease of understanding as compared with mathematical formulas. As far as the contents of the EIL system are concerned, I have relied only on two rather basic concepts which have been shared by almost every computing machinery since Turing machines: sequential control and addition/deletion of information. Yet I believe that I have demonstrated that these computational concepts are useful tools to produce both a satisfactory level of formalization and a flexible and comprehensible level of presentation for the study of language; even if one does not have a computer at hand, these concepts could serve as good metaphors for describing the indispensable constituent in linguistic activities, namely, the human being.

[1] See Brown (1980) and Perrault and Allen (1980), for example, for computational attempts to treat indirect speech acts.
APPENDIX A: SOME OF THE DEFINITIONS OF THE SYSTEM PROCEDURES

The following are formal definitions of some of the system procedures described informally in the dissertation. The definition is written in the style of the programming language C with several informal notations and a lot of unspecified functions which will depend on the details of the implementation of the data structure, which is not the concern here. All the type checks of syntactic constructions are omitted for the sake of simplicity, and the number of arguments in the functional application and the number of variables in quantifications are reduced to one. (The string between */ and */ is a comment.)

super-eval(<m,s>,w,t,f,n)

/* This is part of the definition of the super-interpreter, as
described in (4.1) etc. Only the case for the indicative
mood is shown below. Other mood will look similar */

{if (m=='<Indicative>')
   {++n;
    UDn=[];
    for (i=n; i>=0; —i)
       {val=V[s,w,t,f,n,i];
        if (val)
           {create-nc-info();
            /* val==T, set 'conventional
             implicatures' in UDn. cf. below */
            update();
            /* update UDn-1 through UDO
             based on (2.2) */
            return;
            }
        else if (!val)
           {—n;
            inquire-user();
            /* val==F, violation of the
            convention of truthfulness */
            return;
           }
   }
   else if (val)
      {—n;
       inquire-user();
       /* val==T, set 'conventional
       implicatures' in UDn. cf. below */
       update();
       /* update UDn-1 through UDO
       based on (2.2) */
       return;
      }
   
   /* Other mood will look similar */
}

102
create-nc-info()

/* This procedure performs the manipulation of a-info within the super-interpreter (see above) as described in (6.8) */

{for (dom=UDn; dom!={}; dom=dom-{d})
  {d=pick(dom);
    /* check for each member in UDn */
    if (is-a-info(d))
      {UDn=UDn-{d}; /* delete the a-info */
       c=unmark(d); /* remove the a-marker */
      }
      for (i=0; i>=0; --i) /* for each UDi */
      {for (donp>UDi; dom!={}; dom=dom-{d})
        {d=pick(dom);
          /* check for each member in UDi */
          if (d==c-mark(c)) UDi=UDi-{d}; /* consolidation of previously added 'presupposition' and 'conversational implicature' */
          else if (d==c-mark(neg-mark(c)))
            UDi=UDi-{d}; /* cancellation of previously added 'presupposition' and 'conversational implicature' */
        }
      }
    }
}

for (i=n; i>=0; --i) /* for each UDi */
  {for (dom=UDi; dom!={}; dom=dom-{d})
    {d=pick(dom);
      /* check for each member in UDi */
      if (d==c-mark(c)) UDi=UDi-{d}; /* consolidation of previously added 'presupposition' and 'conversational implicature' */
      else if (d==c-mark(neg-mark(c)))
        UDi=UDi-{d}; /* cancellation of previously added 'presupposition' and 'conversational implicature' */
    }
  }

\[ \{ \text{if} \ (\neg \text{mark}(c) \& \text{UDi}) \ \text{inquire-user()} ;
\]
/* contradictory info exists,\nabsortion of 'conventional\nimplicature' */
else if (c \& \text{UDi}) continue;
/* duplicate info exists,\nabsorption of 'conventional\nimplicature' */
\}
\}
UDn=UDn\cup \{c\};
/* no contradiction nor duplication,\nadd as nc-info, creation of\n'conventional implicature' */
if (is-c-info(d))
\{
UDn=UDn-\{d\};
UDn=UDn\cup \{a\text{-mark(unmark(d))}\};
/* case of double negation, etc.\nc-info becomes a-info and vice\nversa */
\}
\}
\}
\}
V[s,w,t,f,n,i]
/* This is part of the definition of the interpreter proper.\nIn addition to the definitions in (2.4), pragmatic\nprocesses described in later chapters are included. */
if (is-constant(s))
/* constants */
\{d=\text{lex}(s); /* get denotation of the\nconstant from lexicon */
UDn=UDn\cup \{d\}; /* add discourse referent\nto UDn */
return(d);
\}
else if (is-pronoun(s))
/* anaphoric pronouns */
\{d=\text{pick(UDi)}; /* pick a value from UD1 */
UDn=UDn\cup \{d\}; /* add discourse referent\nto UDn */
return(d);
\}
else switch(s)

    {case '(p q)':
        /* functional application */
        d=V[q,w,t,f,n,i];       /* evaluate argument */
        g=V[p,w,t,f,n,i];       /* evaluate function */
        prec(p)(d);             /* perform PREC first */
        val=g(d);               /* calculate denotation */
        postc(p)(d);            /* then perform POSTC */
        if (val) UDn=UDn^\{rep(g,d)\}; /* add d to d-list of g */
        else UDn=UDn^\{neg-mark(rep(g,d))\}; /* add d to nd-list of g */
        return(val);
    }

    case '(NOT p)':
        /* negation */
        val=V[p,w,t,f,n,i];
        create-c-info();        /* set 'presuppositions' in UDn, cf. below */
        update();
        if (val) return(F);
        else return(T);

    case '(EVERY (v) p)':
        /* universal quantification */
        save(UDn);              /* save the current UDn for back-up */
        for (dom=UDimDa; dom!={} ; dom=dom-{u})
            /* check for each member of type a (type of v) in UD_i */
            {u=pick(dom);        /* pick a value */
                val=V[p,w,t,f/v/u,n,i];
                /* f/v/u=v, f/v/u= otherwise */
                restore(UDn);
                /* no discourse referent added */
                if (val) continue;   /* keep on checking */
                else return(val);   /* failure return */
            }
        return(T);            /* every value in UD_i checked */
\textbf{case '}(\textsc{some} (v) p)\textbf{'}:
  
  /* existential quantification */
  \textbf{save}(\textsc{Udn});
  \textbf{for} (\text{dom}=\textsc{UdnDa}; \text{dom}!={}; \text{dom}=\text{dom}-{u})
  
  \{u=pick(\text{dom});
  \text{val}=V[p,w,t,fv/u,n,1];
  \textbf{if} (\text{val}) \{\text{Udn}=\text{Udn}^v\{u\};
  
  /* add discourse referent */
  \textbf{return}(T);
  \}
  \textbf{else} \textbf{restore}(\text{Udn}); /* keep on checking */
  \}
  \textbf{restore}(\text{Udn}); /* search failed */
  \textbf{return}(F);

\textbf{case '}(\text{past} p)\textbf{'}:
  
  /* past tense operator */
  \textbf{save}(\text{Udn});
  \textbf{for} (\text{dom}=\text{UdnT}; \text{dom}!={}; \text{dom}=\text{dom}-{x})
  
  \{x=pick(\text{dom});
  \text{val}=V[p,w,x,f,n,1];
  \textbf{if} (\text{val}) \{\text{Udn}=\text{Udn}^v\{x\};
  
  /* add discourse referent */
  \textbf{return}(T);
  \}
  \textbf{else} \textbf{restore}(\text{Udn});
  \}
  \textbf{restore}(\text{Udn});
  \textbf{return}(F);
\}

\textbf{create-c-info()}

/* This is a procedure to manipulate a-info as part of the pragmatic procedure associated with negation (see above) etc. as described in (6.6). See also the definition of create-ng-info. */

\{\textbf{for} (\text{dom}=\text{Udn}; \text{dom}!={}; \text{dom}=\text{dom}-{d})
  \text{d}=\text{pick}(\text{dom});
  \textbf{if} (\text{is-a-info}(d))
  \{\text{Udn}=\text{Udn}-{d};
  \text{c}=\text{unmark}(d);
  \textbf{for}(i=n; i>=0; --i)
  \}
if (neg-mark(c)∈UD1) goto exit;
/* contradiction with the context, 
  abortion of 'presupposition' */
else if (c∈UD1) goto exit;
/* duplication in the context, 
  absorption of 'presupposition' */
}
UDn=UDn∩{c-mark(c)};
/* no contradiction nor duplication, 
  add to UDn as c-info, 
  creation of 'presupposition' */
if (is-c-info(d))
  {UDn=UDn−{d};
   UDn=UDn∩{a-mark(unmark(d))};
   /* case of double negation, etc. 
      c-info becomes a-info and vice versa */
  }
exit: ; /* check another member in dom */
What follows is a step-by-step trace of the interpreter for the interpretation of the "donkey sentence" mentioned in Chapter 4. The description of the interpretation in that chapter is informal, and this appendix is meant to present a rather cumbersome, detailed description based on the definitions in Appendix A.

Let us start, however, with a little simpler sentence, which only involves a conditional.

(B.1) If Jack owns a donkey, then he beats it.

\[
\text{(IMP (SOME (Y) (AND (DONKEY Y)) (OWN J Y)))}
\]

The subscript shows the label of each subformula, which will appear below. We assume the following simple model in which to interpret (B.1).

The model:

UDn-1={M1,M2,D1,D2}

M2 is Jack, D1 and D2 are donkeys, M2 owns D2, and M2 beats D2.

In the following, the first column shows which subformula we are evaluating at each step, the second shows which individual is bound to the variable Y, the third and the fourth show what are picked up as the referents of the free variables (anaphoric pronouns) HE and IT, respectively, and the last column is the state of UDn after the evaluation of the subformula at each step. Note that UDn both grows and shrinks as the interpretation goes along since the 'save' and the 'restore' operations in the definitions in Appendix A can cancel any temporary UDn so far constructed. The determination of the referents of HE and IT is, of course based on the gender, which is not detailed here. As for the notation, \(<a b>\) means that b is in the d-list of a, while \(<^*a b>\) means that b is in the nd-list of a, i.e., \(\text{rep}(a,b)\) in Appendix A is represented as \(<a b>\) and \(\text{neg-mark}(<a b>)\) as \(<^*a b>\). This is, needless to say, an arbitrary representation in order to have
some kind of (printable) representation and may be drastically different (and more sophisticated) in actual implementation.

<table>
<thead>
<tr>
<th>step</th>
<th>subformula</th>
<th>Y</th>
<th>HE</th>
<th>IT</th>
<th>UDn</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SO (IMP S1 S5)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>2</td>
<td>S1:(SOME (Y) S2)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>3</td>
<td>S2:(AND S3 S4)</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>4</td>
<td>S3:(DONKEY Y)=F</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{&lt;DONKEY M1}&gt;</td>
</tr>
<tr>
<td>5</td>
<td>S2=F</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>6</td>
<td>S2:(AND S3 S4)</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>7</td>
<td>S3:(DONKEY Y)=F</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{&lt;DONKEY M2}&gt;</td>
</tr>
<tr>
<td>8</td>
<td>S2=F</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>9</td>
<td>S2:(AND S3 S4)</td>
<td>D1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>10</td>
<td>S3:(DONKEY Y)=T</td>
<td>D1</td>
<td>-</td>
<td>-</td>
<td>{DONKEY D1}&gt;</td>
</tr>
<tr>
<td>11</td>
<td>S4:(OWN J Y)=F</td>
<td>D1</td>
<td>-</td>
<td>-</td>
<td>{M2,&lt;DONKEY D1}&gt;</td>
</tr>
<tr>
<td>12</td>
<td>S2=F</td>
<td>D1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>13</td>
<td>S2:(AND S3 S4)</td>
<td>D2</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>14</td>
<td>S3:(DONKEY Y)=T</td>
<td>D2</td>
<td>-</td>
<td>-</td>
<td>{DONKEY D2}&gt;</td>
</tr>
<tr>
<td>15</td>
<td>S4:(OWN J Y)=T</td>
<td>D2</td>
<td>-</td>
<td>-</td>
<td>{M2,&lt;DONKEY D2}&gt;</td>
</tr>
<tr>
<td>16</td>
<td>S2=T</td>
<td>D2</td>
<td>-</td>
<td>-</td>
<td>{D2,M2,&lt;DONKEY D2}&gt;</td>
</tr>
<tr>
<td>17</td>
<td>S1=T</td>
<td>M2</td>
<td>-</td>
<td>D2</td>
<td>{D2,M2,&lt;DONKEY D2}&gt;</td>
</tr>
<tr>
<td>18</td>
<td>S0=T</td>
<td>-</td>
<td>-</td>
<td>D2</td>
<td>{D2,M2,&lt;DONKEY D2}&gt;</td>
</tr>
</tbody>
</table>

Note that at step 15, after the binding of Y to D2 gives satisfaction, D2 is added to UDn, and at step 16, after the entire evaluation of the existential quantification successfully terminates, D2 remains in UDn, even though the binding of Y itself is lost. M2 is also added to UDn at the time J is evaluated at step 14, and remains after step 16. Thus, at the time HE and IT are evaluated at step 17, the referents of these pronouns is correctly picked up (considering the gender restriction) from UDn as M2 and D2, respectively, even though the older UDn-1 contained additional M1 and D1.

Now the donkey sentence As Kamp (ms) argues, there is much similarity between the conditional sentence and the universally-quantified sentence. In the EIL translation, they share the IMP operator. The difference is that the conditional in (48) is evaluated with regard to each possible binding of the variable of the universal quantifier, EVERY. The following is the trace of the donkey sentence with the model assumed as shown below (some of the repetitive steps are not shown below).
(4.8) Every man who owns a donkey beats it.

\[(\text{EVERY } (X)) \text{ (IMP } (\text{AND } (\text{MAN } X)) \text{ (DONKEY } Y) \text{ (OWN } X \ Y)) \]\n
\[(\text{BEAT } X \ \text{IT})\]

\[S8\]

The model:

\[U_{Dn-1} = \{M1, M2, M3, D1, D2\}\]

M1, M2, and M3 are men, D1 and D2 are donkeys, M1 owns D1, M2 owns D2, M1 beats D1, M2 beats D2, and M3 beats D2.

<table>
<thead>
<tr>
<th>step</th>
<th>subformula</th>
<th>X</th>
<th>Y</th>
<th>IT</th>
<th>UDn</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S0: (EVERY (X) S1)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>2</td>
<td>S1: (IMP S2 S8)</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>3</td>
<td>S2: (AND S3 S4)</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>4</td>
<td>S3: (MAN X) = T</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{&lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>5</td>
<td>S4: (SOME (Y) S5)</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{&lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>6</td>
<td>S5: (AND S6 S7)</td>
<td>M1</td>
<td>M1</td>
<td>-</td>
<td>{&lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>7</td>
<td>S6: (DONKEY Y) = F</td>
<td>M1</td>
<td>M1</td>
<td>-</td>
<td>{&lt;\text{DONKEY M1}&gt;, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>8</td>
<td>S5 = F</td>
<td>M1</td>
<td>M1</td>
<td>-</td>
<td>{&lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>S5: (AND S6 S7)</td>
<td>M1</td>
<td>D1</td>
<td>-</td>
<td>{&lt;\text{MAN M1}&gt;, &lt;\text{DONKEY D1}&gt;, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>16</td>
<td>S6: (DONKEY Y) = T</td>
<td>M1</td>
<td>D1</td>
<td>-</td>
<td>{&lt;\text{OWN M1 D1}&gt;, &lt;\text{DONKEY D1}&gt;, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>17</td>
<td>S7: (OWN X Y) = T</td>
<td>M1</td>
<td>D1</td>
<td>-</td>
<td>{&lt;\text{MAN M1}&gt;, &lt;\text{DONKEY D1}&gt;, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>18</td>
<td>S5 = T</td>
<td>M1</td>
<td>D1</td>
<td>-</td>
<td>{&lt;\text{DONKEY D1}&gt;, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>19</td>
<td>S4 = T</td>
<td>M1</td>
<td>D1</td>
<td>-</td>
<td>{&lt;\text{OWN M1 D1}&gt;, &lt;\text{DONKEY D1}&gt;, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>20</td>
<td>S2 = T</td>
<td>M1</td>
<td>D1</td>
<td>-</td>
<td>{&lt;\text{DONKEY D1}&gt;, &lt;\text{MAN M1}&gt;, &lt;\text{DONKEY D1}, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>21</td>
<td>S8: (BEAT X IT) = T</td>
<td>M1</td>
<td>-</td>
<td>D1</td>
<td>{&lt;\text{MAN M1}&gt;, &lt;\text{DONKEY D1}, &lt;\text{MAN M1}&gt;}</td>
</tr>
<tr>
<td>22</td>
<td>S1 = T</td>
<td>M1</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>23</td>
<td>S1: (IMP S2 S8)</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>24</td>
<td>S2: (AND S3 S4)</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{}</td>
</tr>
<tr>
<td>25</td>
<td>S3: (MAN X) = T</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{&lt;\text{MAN M2}&gt;}</td>
</tr>
<tr>
<td>26</td>
<td>S4: (SOME (Y) S5)</td>
<td>M2</td>
<td>-</td>
<td>-</td>
<td>{&lt;\text{MAN M2}&gt;}</td>
</tr>
<tr>
<td>27</td>
<td>S5: (AND S6 S7)</td>
<td>M2</td>
<td>M1</td>
<td>-</td>
<td>{&lt;\text{MAN M2}&gt;}</td>
</tr>
<tr>
<td>28</td>
<td>S6: (DONKEY Y) = F</td>
<td>M2</td>
<td>M1</td>
<td>-</td>
<td>{&lt;\text{DONKEY M1}&gt;, &lt;\text{MAN M2}&gt;}</td>
</tr>
<tr>
<td>29</td>
<td>S5 = F</td>
<td>M2</td>
<td>M1</td>
<td>-</td>
<td>{&lt;\text{MAN M2}&gt;}</td>
</tr>
<tr>
<td>Line</td>
<td>Sentence</td>
<td>Refs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>----------</td>
<td>------</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>S5: (AND S6 S7) M2 D1 -</td>
<td>{&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>S6: (DONKEY Y)=T M2 D1 -</td>
<td>{&lt;DONKEY D1&gt;,&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>S7: (OWN X Y)=F M2 D1 -</td>
<td>{&lt;OWN M2 D1&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>S5=F M2 D1 -</td>
<td>{&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>S5: (AND S6 S7) M2 D2 -</td>
<td>{&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>S6: (DONKEY Y)=T M2 D2 -</td>
<td>{&lt;DONKEY D2&gt;,&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>S7: (OWN X Y)=T M2 D2 -</td>
<td>{&lt;OWN M2 D2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>S5=T M2 D2 -</td>
<td>{D2,&lt;OWN M2 D2&gt;,&lt;DONKEY D2&gt;,&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>S4=T M2 D2 -</td>
<td>{D2,&lt;OWN M2 D2&gt;,&lt;DONKEY D2&gt;,&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>S2=T M2 D2 -</td>
<td>{D2,&lt;OWN M2 D2&gt;,&lt;DONKEY D2&gt;,&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>S8: (BEAT X IT)=T M2 - D2</td>
<td>{D2,&lt;OWN M2 D2&gt;,&lt;DONKEY D2&gt;,&lt;MAN M2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>S1=T M2 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>S1: (IMP S2 S8) M3 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>S2: (AND S3 S4) M3 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>S3: (MAN X)=T M3 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>S4: (SOME (X) S5) M3 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>S5 (AND S6 S7) M3 M1 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>S6: (DONKEY Y)=F M3 M1 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>S5=F M3 M1 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>S5 (AND S6 S7) M3 D1 -</td>
<td>{&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>S6: (DONKEY Y)=T M3 D1 -</td>
<td>{&lt;DONKEY D1&gt;,&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>S7: (OWN X Y)=F M3 D1 -</td>
<td>{&lt;OWN M3 D1&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>S5=F M3 D1 -</td>
<td>{&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>S5: (AND S6 S7) M3 D2 -</td>
<td>{&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>S6: (DONKEY Y)=T M3 D2 -</td>
<td>{&lt;DONKEY D2&gt;,&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>S7: (OWN X Y)=F M3 D2 -</td>
<td>{&lt;OWN M3 D2&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>S5=F M3 D2 -</td>
<td>{&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>S4=F M3 -</td>
<td>{&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>S2=F M3 -</td>
<td>{&lt;MAN M3&gt;},</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>S1=T M3 -</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>S0=T</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note that the referent of IT is picked up as D1 at step 21 and as D2 at step 46 depending on who is bound to X at that point. D1 and D2 are established as the discourse referents at steps 18 and 43,
respectively, at the time the binding of Y to D1 and D2, respectively, satisfies the formula. Thus, the interpretation of IT can vary depending on the binding of X, which gives different temporary UDns.
APPENDIX C: INTERPRETATION OF (6.3a) and (6.4a)

This appendix shows how new c-info and nc-info are added to UDn in the case of factive verbs exemplified by (6.3a) and (6.4a). The notational convention is the same as that in Appendix B. The "a-" marker shows that the information temporally added to UDn is the a-info. This marker is eventually changed to "c-" (for c-info) or removed at the final step by the pragmatic procedure associated with the negation or by the super-interpreter (cf Appendix A).

(6.3a) Jack doesn't regret being bald.
(NOT (REGRET J (INT (BALD J))))
S0  S1  S2

The model:
M1 is Jack. M1 doesn't regret being bald. M1's baldness is not known.

<table>
<thead>
<tr>
<th>step subformula</th>
<th>UDn</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 S0: (NOT S1)</td>
<td>{&lt;REGRET M1 S2', a-BALD M1&gt;, M1}</td>
</tr>
<tr>
<td>2 S1: (REGRET J S2) = F</td>
<td>{&lt;REGRET M1 S2', c-BALD M1&gt;, M1}</td>
</tr>
<tr>
<td>3 SO=T</td>
<td>{&lt;REGRET M1 S2', c-BALD M1&gt;, M1}</td>
</tr>
</tbody>
</table>

Note that the status of <BALD M1> changes from that of a-info to that of c-info during the processing of the negation operator. At step 3, we have Jack's baldness as cancelable information.

(6.4a) Jack regrets being bald.
(REGRET J (INT (BALD J)))
S0  S1

The model:
M1 is Jack. M1 regrets being bald. M1's baldness is not known.

<table>
<thead>
<tr>
<th>step subformula</th>
<th>UDn</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 S0: (REGRET J S1)</td>
<td>{&lt;REGRET M1 S1', a-BALD M1&gt;, M1}</td>
</tr>
<tr>
<td>2 (after super-interpretation)</td>
<td>{&lt;REGRET M1 S1', c-BALD M1&gt;, M1}</td>
</tr>
</tbody>
</table>

113
Note that the super-interpreter removes the a-marker of \texttt{<BALD Ml>}, making it noncancelable just as the denotation, \texttt{<REGRET Ml S1'} , is not cancelable.
APPENDIX D: INTERPRETATION OF (7.20)

This appendix shows formally how the coreferentiality of the person who regrets and the person who is bald is established in the interpretation of (7.20) in Chapter 7, which is problematic in Karttunen and Peters (1979)’s system. The notational convention is the same as that in Appendixes B and C.

(7.20) A man regrets being bald.
(SOME (X) (AND (MAN X)
          (REGRET X (INT (BALD X))))
S3 S4

The model:
UDn-1={M1,M2}
M1 and M2 are men, M2 regrets being bald, and M1 and M2 are bald.

<table>
<thead>
<tr>
<th>step</th>
<th>subformula</th>
<th>X</th>
<th>UDn</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S0:(SOME (X) S1)</td>
<td></td>
<td>{}</td>
</tr>
<tr>
<td>2</td>
<td>S1:(AND S2 S3)</td>
<td>M1</td>
<td>{}</td>
</tr>
<tr>
<td>3</td>
<td>S2:(MAN X)=T</td>
<td>M1</td>
<td>{&lt;MAN M1&gt;}</td>
</tr>
</tbody>
</table>
| 4    | S3:(REGRET X S4)=F M1          | M1         | {<REGRET M1 S4'>, a-<BALD M1>,
          <MAN M1>}>             |
| 5    | S1=F                           | M1         | {}                         |
| 6    | S1:(AND S2 S3)                 | M2         | {}                         |
| 7    | S2:(MAN X)=T                   | M2         | {<MAN M2>}                 |
| 8    | S3:(REGRET X S4)=T M2          | M2         | {<REGRET M2 S4'> a-<BALD M2>,
          <MAN M2>}>             |
| 9    | S1=T                           | M2         | {M2, <REGRET M2 S4'> a-<BALD M2>,
          <MAN M2>}>             |
| 10   | S0=T                           |            | {M2, <REGRET M2 S4'>, <BALD M2>,
          <MAN M2>}>             |

Note that the implicature is <BALD M2>, i.e., the person M2 is bald, since the a-info which is the origin of this implicature is produced at the time when X is bound to M2. If, on the other hand (7.20) and (D.1) below are evaluated separately as the denotation and the implicature, as in Karttunen and Peters’s system, the evaluation of (D.1) can give M1 as the man who is bald, since both M1 and M2 satisfy the formula and there is no reason to choose one over the other.
Since the EIL system can produce an implicature in the middle of evaluation of quantification, coreferrentiality is automatically established without any additional (and ad hoc) mechanism.


Cooper, R. (ms. undated), "Temporal adverbs as generalized quantifiers," manuscript, University of Wisconsin.


