08/674610

ISSUE CLASSIFICATION	
Class	Subclass
704	9

[redacted]

May

5966686

5966686

UTILITY SERIAL NUMBER 08/674610	PATENT DATE OCT 12 1999	PATENT NUMBER

SERIAL NUMBER	FILING DATE	CLASS	SUBCLASS	GROUP ART UNIT	EXAMINER
		704	9	2741	THOMAS

APPLICANTS

NONE

Ø J.T.

NONE

Ø, J.T.

CPA

CERTIFICATE

JAN 2 2001

OF CORRECTION

Foreign priority claimed 35 USC 119 conditions met / Verified and Acknowledged	☐ yes ☑ no / ☐ yes ☑ no / Examiner's initials J.T.	AS FILED →	STATE OR COUNTRY	SHEETS DRWGS.	TOTAL CLAIMS	INDEP. CLAIMS	FILING FEE RECEIVED	ATTORNEY'S DOCKET NO.

ADDRESS

TITLE

U.S. DEPT. OF COMM./ PAT. & TM—PTO-436L (Rev.12-94)

PARTS OF APPLICATION FILED SEPARATELY	Applications Examiner

NOTICE OF ALLOWANCE MAILED		CLAIMS ALLOWED	
4-27-99	Assistant Examiner	Total Claims: 9	Print Claim: 1

ISSUE FEE 8/12			DRAWING		
Amount Due: 120.00	Date Paid: 7-23-99	JOSEPH THOMAS PRIMARY EXAMINER A.U. 2747 Primary Examiner	Sheets Drwg.: 69	Figs. Drwg.: 69	Print Fig.: 25
Label Area		PREPARED FOR ISSUE	ISSUE BATCH NUMBER: 542		

WARNING: The information disclosed herein may be restricted. Unauthorized disclosure may be prohibited by the United States Code Title 35, Sections 122, 181 and 368. Possession outside the U.S. Patent & Trademark Office is restricted to authorized employees and contractors only.

Form PTO-436A
(Rev. 8/92)

Formal Drawings (____ shts) set ____

BW

ISSUE FEE IN FILE

(FACE)

08/674610

PATENT APPLICATION

08674610

APPROVED FOR LICENSE
AUG 14 96
INITIALS ____

CONTENTS

Date Entered or Counted			Date Received or Mailed
	1.	Application ✓ papers.	
	2.	[illegible]	[illegible]
	3.	[illegible]	9-23-96
	4.	[illegible]	Dec 11, 1996
	5.	CFR	12-6-97
	6.	IDS	01-20-98
3/27	7.	Rej. (3 mos)	3/27/98 N.N. c/m
	8.	Ext. of time (1 month)	7-30-98 7-27
	9.	Amdt. A	7-30-98
9/14	10.	Final Rejection (3 months)	9/15/98
	11.	Amdt. B ~~(AF)~~	12-14-98
12-23	12.	Advisory Action	12-28-98
	13.	Ext. of [illegible] (2) + CPA REQUEST	2-16-99
	14.	Pre Amdt C	3-4-99
4/20	15.	Notice of Allowance Formal Drawings (shts) set	4-22-99
9-8-99	16.	Formal Drawings (1 shts) set 1	7-19-99
	17.	Req for C of C K322	4-19-00
	18.	Approval-In-Part	10-25-00
	19.		
	20.		
	21.		
	22.		
	23.		
	24.		
	25.		
	26.		
	27.		
	28.		
	29.		
	30.		
	31.		
	32.		

BAR CODE LABEL

U.S. PATENT APPLICATION

SERIAL NUMBER	FILING DATE	CLASS	GROUP ART UNIT
08/674,610	06/28/96	395	2412

APPLICANT

GEORGE HEIDORN, BELLEVUE, WA; KAREN JENSEN, BELLEVUE, WA.

CONTINUING DATA*******************
VERIFIED

FOREIGN/PCT APPLICATIONS**********
VERIFIED

FOREIGN FILING LICENSE GRANTED 11/20/96

STATE OR COUNTRY	SHEETS DRAWING	TOTAL CLAIMS	INDEPENDENT CLAIMS	FILING FEE RECEIVED	ATTORNEY DOCKET NO.
WA	69	36	7	$1,544.00	661005.447

ADDRESS

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

TITLE

METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

This is to certify that annexed hereto is a true copy from the records of the United States Patent and Trademark Office of the application which is identified above.

By authority of the
COMMISSIONER OF PATENTS AND TRADEMARKS

Date Certifying Officer

FORM PTO-1082

SEED and BERRY LLP
6300 Columbia Center
Seattle, Washington 98104-7092
Phone (206) 622-4900
Fax (206) 682-6031

08/674610

JUN 28 1996 PAT. & TRADEMARK OFF.

Express Mail Certificate No.: **EM417213544**
Docket No.: **661005.447**
Date: **June 28, 1996**

BOX PATENT APPLICATION
ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON DC 20231

Sir:

Transmitted herewith for filing is the patent application of:

Inventors: **George Heidorn and Karen Jensen**

For: **METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES**

Enclosed are:

[**X**] 69 sheets of informal drawings (Figs. 1-59).
[ ] An assignment of the invention to: Microsoft Corporation, a corporation of the State of Washington.
[ ] A Declaration and Power of Attorney.
[ ] A verified statement to establish small entity status under 37 C.F.R. 1.9 and 37 C.F.R. 1.27.
[ ] A certified copy of Application No. , filed , from which priority is claimed, .
[ ] This application claims the benefit of U.S. Provisional Application No. , filed . **[DELETE IF NOT APPLICABLE]**
[ ] The filing fee has been calculated as shown below.
[**X**] Filed without fee or formal papers.

			Small Entity		or	Other Than A Small Entity	
For:	**No. Filed**	**No. Extra**	**Rate**	**Fee**	**or**	**Rate**	**Fee**
Utility Fee				**$**	**or**		**$**
Total Claims			**x 11**	**$**	**or**	**x 22**	**$**
Independent Claims			**x 39**	**$**	**or**	**x 78**	**$**
( ) Multiple Dependent Claim Presented			**+ 125**	**$**	**or**	**+ 250**	**$**
ASSIGNMENT			**+ 40**	**$**	**or**	**+ 40**	**$**
			TOTAL	**$**	**or**	**TOTAL**	**$**

[ ] Please charge my Deposit Account No. 19-1090 in the amount of $_. A duplicate copy of this sheet is enclosed.
[ ] A check in the amount of $_is enclosed.
[ ] The Assistant Commissioner is hereby authorized to charge payment of the following fees associated with this communication or credit any overpayment to Deposit Account No. 19-1090. A duplicate copy of this sheet is enclosed.
[ ] Any additional filing fees required under 37 C.F.R. 1.16.
[ ] Any patent application processing fees under 37 C.F.R. 1.17.

Respectfully submitted,
SEED and BERRY LLP

Robert W. Bergstrom

Robert W. Bergstrom
Registration No. 39,906

08/674610

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Docket No. : 661005.447
Date : June 28, 1996

Box Patent Application
Assistant Commissioner for Patents
Washington, DC 20231

CERTIFICATE OF MAILING BY "EXPRESS MAIL"

Sir:

I hereby certify that the enclosures listed below are being deposited with the United States Postal Service "EXPRESS MAIL Post Office to Addressee" service under 37 C.F.R. § 1.10, Mailing Label Certificate No. EM417213544, on June 28, 1996, addressed to: Box Patent Application, Assistant Commissioner for Patents, Washington, DC 20231.

Respectfully submitted,

SEED and BERRY LLP

Jeanne L. Coward

Jeanne L. Coward

RWB:jlc

Enclosures:
- Postcard
- Form PTO-1082 (+ copy)
- Specification, claims, abstract
- Informal Drawings (Figures 1-59)

c:\rwb\0188

69

EXPRESS M/ NO. EM4172~~135~~ A No Fee

08/674610

Description

METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Technical Field

The present invention relates to the field of natural language processing ("NLP"), and more particularly, to a method and system for generating a logical form graph from a syntax tree.

Background of the Invention

Computer systems for automatic natural language processing use a variety of subsystems, roughly corresponding to the linguistic fields of morphological, syntactic, and semantic analysis to analyze input text and achieve a level of machine understanding of natural language. Having understood the input text to some level, a computer system can, for example, suggest grammatical and stylistic changes to the input text, answer questions posed in the input text, or effectively store information represented by the input text.

Morphological analysis identifies input words and provides information for each word that a human speaker of the natural language could determine by using a dictionary. Such information might include the syntactic roles that a word can play (*e.g.*, noun or verb) and ways that the word can be modified by adding prefixes or suffixes to generate different, related words. For example, in addition to the word "fish," the dictionary might also list a variety of words related to, and derived from, the word "fish," including "fishes," "fished,"

2

"fishing," "fisher," "fisherman," "fishable," "fishability," "fishbowl," "fisherwoman," "fishery," "fishhook," "fishnet," and "fishy."

Syntactic analysis analyzes each input sentence, using, as a starting point, the information provided by the morphological analysis of input words and the set of syntax rules that define the grammar of the language in which the input sentence was written. The following are sample syntax rules:

sentence	=	noun phrase + verb phrase
noun phrase	=	adjective + noun
verb phrase	=	adverb + verb

Syntactic analysis attempts to find an ordered subset of syntax rules that, when applied to the words of the input sentence, combine groups of words into phrases, and then combine phrases into a complete sentence. For example, consider the input sentence: "Big dogs fiercely bite." Using the three simple rules listed above, syntactic analysis would identify the words "Big" and "dogs" as an adjective and noun, respectively, and apply the second rule to generate the noun phrase "Big dogs." Syntactic analysis would identify the words "fiercely" and "bite" as an adverb and verb, respectively, and apply the third rule to generate the verb phrase "fiercely bite." Finally, syntactic analysis would apply the first rule to form a complete sentence from the previously generated noun phrase and verb phrase. The result of syntactic analysis, often represented as an acyclic downward branching tree with nodes representing input words, punctuation symbols, phrases, and a root node representing an entire sentence, is called a parse.

3

Some sentences, however, can have several different parses. A classic example sentence for such multiple parses is: "Time flies like an arrow." There are at least three possible parses corresponding to three possible meanings of this sentence. In the first parse, "time" is the subject of the sentence, "flies" is the verb, and "like an arrow" is a prepositional phrase modifying the verb "flies." However, there are at least two unexpected parses as well. In the second parse, "time" is an adjective modifying "flies," "like" is the verb, and "an arrow" is the object of the verb. This parse corresponds to the meaning that flies of a certain type, "time flies," like or are attracted to an arrow. In the third parse, "time" is an imperative verb, "flies" is the object, and "like an arrow" is a prepositional phrase modifying "time." This parse corresponds to a command to time flies as one would time an arrow, perhaps with a stopwatch.

Syntactic analysis is often accomplished by constructing one or more hierarchical trees called syntax parse trees. Each leaf node of the syntax parse tree generally represents one word or punctuation symbol of the input sentence. The application of a syntax rule generates an intermediate-level node linked from below to one, two, or occasionally more existing nodes. The existing nodes initially comprise only leaf nodes, but, as syntactic analysis applies syntax rules, the existing nodes comprise both leaf nodes as well as intermediate-level nodes. A single root node of a complete syntax parse tree represents an entire sentence.

Semantic analysis generates a logical form graph that describes the meaning of input text in a deeper way than can be described by a syntax parse tree alone. The logical form graph is a first attempt to understand the input text at a level analogous to that achieved by a human speaker of the language.

4

The logical form graph has nodes and links, but, unlike the syntax parse tree described above, is not hierarchically ordered. The links of the logical form graph are labeled to indicate the relationship between a pair of nodes. For example, semantic analysis may identify a certain noun in a sentence as the deep subject or deep object of a verb. The deep subject of a verb is the doer of the action and the deep object of a verb is the object of the action specified by the verb. The deep subject of an active voice verb may be the syntactic subject of the sentence, and the deep object of an active voice verb may be the syntactic object of the verb. However, the deep subject of a passive voice verb may be expressed in an agentive-by phrase, and the deep object of a passive voice verb may be the syntactic subject of the sentence. For example, consider the two sentences: (1) "Dogs bite people" and (2) "People are bitten by dogs." The first sentence has an active voice verb, and the second sentence has a passive voice verb. The syntactic subject of the first sentence is "Dogs" and the syntactic object of the verb "bite" is "people." By contrast, the syntactic subject of the second sentence is "People" and the verb phrase "are bitten" is modified by the agentive-by phrase "by dogs." For both sentences, "dogs" is the deep subject, and "people" is the deep object of the verb or verb phrase of the sentence. Although the syntax parse trees generated by syntactic analysis for sentences 1 and 2, above, will be different, the logical form graphs generated by semantic analysis will be the same, because the underlying meaning of the two sentences is the same.

Further semantic processing after generation of the logical form graph may draw on knowledge databases to relate analyzed text to real world concepts in order to achieve still deeper levels of understanding. An example

5

knowledge base would be an on-line encyclopedia, from which more elaborate definitions and contextual information for particular words can be obtained.

In the following, the three NLP subsystems -- morphological, syntactic, and semantic -- are described in the context of processing the sample input text: "The person whom I met was my friend." Figure 1 is a block diagram illustrating the flow of information between the NLP subsystems. The morphological subsystem 101 receives the input text and outputs an identification of the words and senses for each of the various parts of speech in which each word can be used. The syntactic subsystem 102 receives this information and generates a syntax parse tree by applying syntax rules. The semantic subsystem 103 receives the syntax parse tree and generates a logical form graph.

Figures 2-5 display the dictionary information stored on an electronic storage medium that is retrieved for the input words of the sample input text during morphological analysis. Figure 2 displays the dictionary entries for the input words "the" 201 and "person" 202. Entry 201 comprises the key "the" 203 and a list of attribute/value pairs. The first attribute "Adj" 204 has, as its value, the symbols contained within the braces 205 and 206. These symbols comprise two further attribute/value pairs: (1) "Lemma" / "the" and (2) "Bits" / "Sing Plur Wa6 Det Art B0 Def." A lemma is the basic, uninflected form of a word. The attribute "Lemma" therefore indicates that "the" is the basic, uninflected form of the word represented by this entry in the dictionary. The attribute "Bits" comprises a set of abbreviations representing certain morphological and syntactic information about a word. This information indicates that "the" is: (1) singular; (2) plural; (3) not inflectable; (4) a

6

determiner; (5) an article; (6) an ordinary adjective; and (7) definite. Attribute 204 indicates that the word "the" can serve as an adjective. Attribute 212 indicates that the word "the" can serve as an adverb. Attribute "Senses" 207 represents the various meanings of the word as separate definitions and examples, a portion of which are included in the list of attribute/value pairs between braces 208-209 and between braces 210-211. Additional meanings actually contained in the entry for "the" have been omitted in Figure 2, indicated by the parenthesized expression "(more sense records)" 213.

In the first step of natural language processing, the morphological subsystem recognizes each word and punctuation symbol of the input text as a separate token and constructs an attribute/value record for each part of speech of each token using the dictionary information. Attributes are fields within the records that can have one of various values defined for the particular attribute. These attribute/value records are then passed to the syntactic subsystem for further processing, where they are used as the leaf nodes of the syntax parse tree that the syntactic subsystem constructs. All of the nodes of the syntax parse tree and the logical form graph constructed by subsequent NLP subsystems are attribute/value records.

The syntactic subsystem applies syntax rules to the leaf nodes passed to the syntactic subsystem from the morphological subsystem to construct higher-level nodes of a possible syntax parse tree that represents the sample input text. A complete syntax parse tree includes a root node, intermediate-level nodes, and leaf nodes. The root node represents the syntactic construct (*e.g.*, declarative sentence) for the sample input text. The intermediate-level nodes

4

represent intermediate syntactic constructs (*e.g.*, verb, noun, or prepositional phrases). The leaf nodes represent the initial set of attribute/value records.

In some NLP systems, syntax rules are applied in a top-down manner. The syntactic subsystem of the NLP system herein described applies syntax rules to the leaf nodes in a bottom-up manner. That is, the syntactic subsystem attempts to apply syntax rules one-at-a-time to single leaf nodes to pairs of leaf nodes, and, occasionally, to larger groups of leaf nodes. If the syntactic rule requires two leaf nodes upon which to operate, and a pair of leaf nodes both contain attributes that match the requirements specified in the rule, then the rule is applied to them to create a higher-level syntactic construct. For example, the words "my friend" could represent an adjective and a noun, respectively, which can be combined into the higher-level syntactic construct of a noun phrase. A syntax rule corresponding to the grammar rule, "noun phrase = adjective + noun," would create an intermediate-level noun phrase node, and link the two leaf nodes representing "my" and "friend" to the newly created intermediate-level node. As each new intermediate-level node is created, it is linked to already-existing leaf nodes and intermediate-level nodes, and becomes part of the total set of nodes to which the syntax rules are applied. The process of applying syntax rules to the growing set of nodes continues until either a complete syntax parse tree is generated or until no more syntax rules can be applied. A complete syntax parse tree includes all of the words of the input sentence as leaf nodes and represents one possible parse of the sentence.

This bottom-up method of syntax parsing creates many intermediate-level nodes and sub-trees that may never be included in a final,

4

complete syntax parse tree. Moreover, this method of parsing can simultaneously generate more than one complete syntax parse tree.

The syntactic subsystem can conduct an exhaustive search for all possible complete syntax parse trees by continuously applying the rules until no additional rules can be applied. The syntactic subsystem can also try various heuristic approaches to first generate the most probable nodes. After one or a few complete syntax parse trees are generated, the syntactic subsystem typically can terminate the search because the syntax parse tree most likely to be chosen as best representing the input sentence is probably one of the first generated syntax parse trees. If no complete syntax parse trees are generated after a reasonable search, then a fitted parse can be achieved by combining the most promising sub-trees together into a single tree using a root node that is generated by the application of a special aggregation rule.

Figure 6 illustrates the initial leaf nodes created by the syntactic subsystem for the dictionary entries initially displayed in Figures 2-5. The leaf nodes include two special nodes, 601 and 614, that represent the beginning of the sentence and the period terminating the sentence, respectively. Each of the nodes 602-613 represent a single part of speech that an input word can represent in a sentence. These parts of speech are found as attribute/value pairs in the dictionary entries. For example, leaf nodes 602 and 603 represent the two possible parts of speech for the word "The," that are found as attributes 204 and 212 in Figure 2.

Figure 7-22 show the rule-by-rule construction of the final syntax parse tree by the syntactic subsystem. Each of the figures illustrates the application of a single syntax rule to generate an intermediate-level node that represents a syntactic structure. Only the rules that produce the intermediate-level nodes that comprise the final syntax tree are illustrated. The syntactic subsystem generates many intermediate-level nodes which do not end up included in the final syntax parse tree.

In Figures 7-14, the syntactic subsystem applies unary syntax rules that create intermediate-level nodes that represent simple verb, noun, and adjective phrases. Starting with Figure 15, the syntactic subsystem begins to apply binary syntax rules that combine simple verb, noun, and adjective phrases into multiple-word syntactic constructs. The syntactic subsystem orders the rules by their likelihood of successful application, and then attempts to apply them one-by-one until it finds a rule that can be successfully applied to the existing nodes. For example, as shown in Figure 15, the syntactic subsystem successfully applies a rule that creates a node representing a noun phrase from an adjective phrase and a noun phrase. The rule specifies the characteristics required of the adjective and noun phrases. In this example, the adjective phrase must be a determiner. By following the pointer from node 1501 back to node 1503, and then accessing morphological information included in node 1503, the syntactic subsystem determines that node 1501 does represent a determiner. Having located the two nodes 1501 and 1502 that meet the characteristics required by the rule, the syntactic subsystem then applies the rule to create from the two simple phrases 1501 and 1502 an intermediate-level node that represents the noun phrase "my friend." In Figure 22, the syntactic subsystem generates the final, complete syntax parse tree representing the input sentence by applying a trinary rule that combines the special Begin1 leaf node 2201, the verb phrase "The person whom I met was my friend" 2202, and the leaf node 2203 that represents the final terminating period to form node 2204 representing the declarative sentence.

The semantic subsystem generates a logical form graph from a complete syntax parse tree. In some NLP systems, the logical form graph is constructed from the nodes of a syntax parse tree, adding to them attributes and new bi-directional links. The logical form graph is a labeled, directed graph. It is a semantic representation of an input sentence. The information obtained for each word by the morphological subsystem is still available through references to the leaf nodes of the syntax parse tree from within nodes of the logical form graph. Both the directions and labels of the links of the logical form graph represent semantic information, including the functional roles for the nodes of the logical form graph. During its analysis, the semantic subsystem adds links and nodes to represent (1) omitted, but implied, words; (2) missing or unclear arguments and adjuncts for verb phrases; and (3) the objects to which prepositional phrases refer.

Figure 23 illustrates the complete logical form graph generated by the semantic subsystem for the example input sentence. Meaningful labels have been assigned to links 2301-2306 by the semantic subsystem as a product of the successful application of semantic rules. The six nodes 2307-2312, along with the links between them, represent the essential components of the semantic meaning of the sentence. In general, the logical form nodes roughly correspond to input words, but certain words that are unnecessary for conveying semantic meaning, such as "The" and "whom" do not appear in the logical form graph, and the input verbs "met" and "was" appear as their infinitive forms "meet" and "be." The nodes are represented in the computer system as records, and contain additional information not shown in Figure 23. The fact that the verbs were input in singular past tense form is indicated by additional information within the logical form nodes corresponding to the meaning of the verbs, 2307 and 2310.

The differences between the syntax parse tree and the logical form graph are readily apparent from a comparison of Figure 23 to Figure 22. The syntax parse tree displayed in Figure 22 includes 10 leaf nodes and 16 intermediate-level nodes linked together in a strict hierarchy, whereas the logical form graph displayed in Figure 23 contains only 6 nodes. Unlike the syntax parse tree, the logical form graph is not hierarchically ordered, obvious from the two links having opposite directions between nodes 2307 and 2308. In addition, as noted above, the nodes no longer represent the exact form of the input words, but instead represent their meanings.

Further natural language processing steps occur after semantic analysis. They involve combining the logical form graph with additional information obtained from knowledge bases, analyzing groups of sentences, and generally attempting to assemble around each logical form graph a rich contextual environment approximating that in which humans process natural language.

Prior art methods for generating logical form graphs involve computationally complex adjustments to, and manipulations of the syntax parse tree. As a result, it becomes increasingly difficult to add new semantic rules to a NLP system. Addition of a new rule involves new procedural logic that may conflict with the procedural logic already programmed into the semantic subsystem. Furthermore, because nodes of the syntax parse tree are extended and reused as nodes for the logical form graph, prior art semantic subsystems

12

produce large, cumbersome, and complicated data structures. The size and complexity of a logical form graph overlayed onto a syntax parse tree makes further use of the combined data structure error-prone and inefficient. Accordingly, it would be desirable to have a more easily extended and manageable semantic subsystem so that simple logical form graph data structures can be produced.

Summary of the Invention

The present invention is directed to a method and system for performing semantic analysis of an input sentence within a NLP system. The semantic analysis subsystem receives a syntax parse tree generated by the morphological and syntactic subsystems. The semantic analysis subsystem applies two sets of semantic rules to make adjustments to the received syntax parse tree. The semantic analysis subsystem then applies a third set of semantic rules to create a skeletal logical form graph from the syntax parse tree. The semantic analysis subsystem finally applies two additional sets of semantic rules to the skeletal logical form graph to provide semantically meaningful labels for the links of the logical form graph, to create additional logical form graph nodes for missing nodes, and to unify redundant logical form graph nodes. The final logical form graph generated by the semantic analysis subsystem represents the complete semantic analysis of an input sentence.

Brief Description of the Drawings

Figure 1 is a block diagram illustrating the flow of information between the subsystems of a NLP system.

Figures 2-5 display the dictionary information stored on an electronic storage medium that is retrieved for each word of the example input sentence: “The person whom I met was my friend.”

Figure 6 displays the leaf nodes generated by the syntactic subsystem as the first step in parsing the input sentence.

Figures 7-22 display the successive application of syntax rules by the syntactic subsystem to parse of the input sentence and produce a syntax parse tree.

Figure 23 illustrates the logical form graph generated by the semantic subsystem to represent the meaning of the input sentence.

Figure 24 shows a block diagram illustrating a preferred computer system for natural language processing.

Figure 25 illustrates the three phases of the preferred new semantical subsystem.

Figure 26 is a flow diagram for the new semantic subsystem (NSS).

Figure 27 displays the first set of semantic rules.

Figure 28A displays a detailed description of the semantic rule PrLF_You from the first set of semantic rules.

Figure 28B displays an example application of the semantic rule PrLF_You from the first set of semantic rules.

Figure 29 displays the second set of semantic rules.

Figures 30A-30B display a detailed description of the semantic rule TrLF_MoveProp from the second set of semantic rules.

14

Figure 30C displays an example application of the semantic rule TrLF_MoveProp from the second set of semantic rules.

Figure 31 displays a flow diagram for apply_rules.

Figure 32 displays a flow diagram for phase one of the NSS.

Figure 33 displays the third set of semantic rules.

J.T. 4/19/99

Figures 34A-~~C~~ 34C display a detailed description of the semantic rule SynToSem1 from the third set of semantic rules.

Figure 34D displays an example application of the semantic rule SynToSem1 from the third set of semantic rules.

Figure 35 displays a flow diagram for phase two of the NSS.

Figures 36-38 display the fourth set of semantic rules.

Figure 39A displays a detailed description of the semantic rule LF_Dobj2 from the fourth set of semantic rules.

Figure 39B displays an example application of the semantic rule LF_Dobj2 from the fourth set of semantic rules.

Figure 40 displays the fifth set of semantic rules.

J.T. 4/19/99

Figures 41A-~~C~~ 41C display a detailed description of the semantic rule PsLF_PronAnaphora from the fifth set of semantic rules.

Figure 41D displays an example application of the semantic rule PsLF_PronAnaphora from the fifth set of semantic rules.

Figure 42 displays a flow diagram for phase three of the NSS.

Figure 43 is a block diagram of a computer system for the NSS.

Figures 44-59 display each successful rule application by the NSS as it processes the syntax parse tree generated for the example input sentence.

15

Detailed Description of the Invention

The present invention provides a new semantic method and system for generating a logical form graph from a syntax tree. In a preferred embodiment, a new semantic subsystem (NSS) performs the semantic analysis in three phases: (1) filling in and adjusting the syntax parse tree, (2) generating an initial logical form graph, and (3) generating meaningful labels for links of the logical form graph and constructing a complete logical form graph. Each phase constitutes the application of one or two sets of rules to either a set of syntax tree nodes or to a set of logical form graph nodes.

The NSS addresses the recognized deficiencies in prior art semantic subsystems described above in the background section. Each phase of the NSS is a simple and extensible rule-based method. As additional linguistic phenomena are recognized, rules to handle them can be easily included into one of the rule sets employed by the NSS. In addition, the second phase of the NSS generates an entirely separate logical form graph, rather than overlaying the logical form graph onto an existing syntax parse tree. The logical form graph data structure generated by the NSS is therefore simple and space efficient by comparison with prior art logical form graph data structures.

Figure 24 is a block diagram illustrating a preferred computer system for a NLP system. The computer system 2401 contains a central processing unit, a memory, a storage device, and input and output devices. The NLP subsystems 2406-2409 are typically loaded into memory 2404 from a computer-readable storage device such as a disk. An application program 2405 that uses the services provided by the NLP system is also typically loaded into

16

memory. The electronic dictionary 2411 is stored on a storage device, such as a disk 2410, and entries are read into memory for use by the morphological subsystem. In one embodiment, a user typically responds to a prompt displayed on the output device 2403 by entering one or more natural language sentences on an input device 2404. The natural language sentences are received by the application, processed, and then passed to the NLP system by way of the morphological subsystem 2406. The morphological subsystem uses information from the electronic dictionary to construct records describing each input word, and passes those records to the syntactic subsystem 2407. The syntactic subsystem parses the input words to construct a syntax parse tree and passes the syntax parse tree to the semantic subsystem 2408. The semantic subsystem generates a logical form graph from the received syntax parse tree and passes that logical form graph to other NLP subsystems 2409. The application program then can send and receive information to the natural language subsystem 2409 in order to make use of the machine understanding of the input text achieved by the NLP system, and then finally output a response to the user on an output device 2403.

Figure 25 illustrates the three phases of the preferred new semantic subsystem. Phases 1-3 of the NSS are shown as 2502, 2504, and 2506, respectively. The states of the relevant data structures input and output from each phase of the NSS are displayed in Figure 25 as labels 2501, 2503, 2505, and 2507. The NSS receives a syntax parse tree 2501 generated by the syntactic subsystem. The first phase of the NSS 2502 completes the syntax parse tree using semantic rules, and passes the completed syntax parse tree 2503 to the second phase of the NSS 2504. The second phase of the NSS generates an initial logical form graph 2505 and passes that initial logical form graph to the third phase of the NSS 2506. The third phase of the NSS applies semantic rules to the initial logical form graph in order to add meaningful semantic labels to the links of the logical form graph, to add new links and nodes to fill out the semantic representation of the input sentence, and occasionally to remove redundant nodes. The complete logical form graph 2507 is then passed to other NLP subsystems for use in further interpreting the input sentence represented by the logical form graph or in answering questions or preparing data based on the input sentence.

A flow diagram for the NSS is displayed in Figure 26. The flow diagram shows successive invocation of the three phases of the NSS, 2601, 2602, and 2603. In the following, each phase of the NSS will be described in detail.

NSS Phase One - Completing Syntactic Roles of the Syntax Tree

In phase one of the NSS, the NSS modifies a syntax parse tree received from the syntactic subsystem by applying two different sets of semantic rules to the nodes of the syntax parse tree. These semantic rules can alter the linkage structure of the syntax tree or cause new nodes to be added.

The NSS applies a first set of semantic rules to resolve a variety of possible omissions and deficiencies that cannot be addressed by syntactical analysis. Application of these first set of semantic rules effect preliminary adjustments to the input syntax parse tree. The linguistic phenomena addressed by the first set of semantic rules include verbs omitted after the words "to" or "not," but understood to be implicit by a human listener, missing pronouns, such as "you" or "we" in imperative sentences, expansion of coordinate structures

A

involving the words "and" or "or," and missing objects or elided verb phrases. Figure 27 lists a preferred first set of semantic rules applied by the NSS in phase one. For each rule, the name of the rule followed by a concise description of the linguistic phenomenon that it addresses is shown.

The general format of each semantic rule is a set of conditions which are applied to a syntax parse tree node or logical form graph node and a list of actions that are applied to the syntax parse tree or logical form graph. For example, the NSS applies the conditions of each rule of the first set of semantic rules to the list of syntax records that represents the syntax parse tree and, for each rule for which all the conditions of that rule are satisfied, the NSS performs the list of actions contained in the rule, resulting in specific changes to the syntax parse tree. Of course, the actual form of each semantic rule depends on the details of the representation of the syntax parse tree and logical form graph, for which many different representations are possible. In the following figures, a semantic rule is described by a conditional expression preceded by the word "If" in bold type, followed by a list of actions preceded by the word "Then" in bold type. The "If" part of the semantic rule represents the conditions that must be applied to a syntax parse tree node or logical form graph node and found to be true in order for the rule, as a whole, to be applied to the node, and the "Then" expression represents a list of actions to be performed on the syntax parse tree or logical form graph. The displayed expression closely corresponds to the computer source code expression for the semantic rule.

Figure 28A displays an English-language representation of the semantic rule PrLF_You from the first set of semantic rules. As can be seen in Figure 28A, the "If" expression concerns the values of various attributes of the

19

syntax parse tree node to which the rule is applied, and the "Then" expression specifies the creation of a pronoun node for the lemma "you" and a noun phrase node parent for the pronoun node and the attachment of the created nodes to the syntax parse tree.

Figure 28B shows an example of the application of the semantic rule PrLF_You to the syntax parse tree 2801 generated by the syntactic subsystem for the sentence "Please close the door." Application of PrLF_You results in the modified syntax parse tree 2802, with two new nodes 2803 and 2804 connected to the root node for the sentence. This semantic rule has the purpose of explicitly placing an understood "you" of an imperative sentence into the syntax parse tree.

After all semantic rules of the first set of semantic rule that can be applied to the input syntax parse tree have been applied, the NSS makes main adjustments to the preliminarily-adjusted syntax parse tree by applying to the nodes of the preliminarily-adjusted syntax parse tree a second set of semantic rules. This second set of rules include rules that serve to identify and resolve long-distance attachment phenomena, to transform verbal phrases into verbs with prepositional phrase objects, and to replace, in certain cases, the word "it" with an infinitive clause.

Figure 29 lists a preferred second set of semantic rules applied by the NSS in phase one. For each rule, the name of the rule followed by a concise description of the linguistic phenomenon that it addresses is shown. Figures 30A-30B display an English-language representation of the semantic rule TrLF_MoveProp from the second set of semantic rules. As can be seen in Figures 30A-30B, the "If" expression concerns the values of various attributes of the syntax parse tree node to which the rule is applied and various related syntax parse tree nodes, and the "Then" expression specifies a rather complex rearrangement of the syntax parse tree.

Figure 30C shows an example of the application of the semantic rule TrLF_MoveProp to the syntax parse tree 3001 generated by the syntactic subsystem for the sentence "I have no desire to see the man." Application of TrLF_MoveProp results in the modified syntax parse tree 3002. The infinitive clause represented by node 3003 in the original syntax parse tree has been moved from its position as a child of node 3004 to being a child 3005 of the root DECL1 node 3006 of the modified syntax parse tree. This semantic rule has the purpose of moving clauses like the infinitive clause 3003 from a lower level to a higher level in the syntax tree to facilitate the subsequent transition from the syntax parse tree to a logical form graph.

In the preferred embodiment of the present invention, semantic rules are statements in a programming language that, when executed, create a new tree or graph node from one, two, or occasionally more existing tree or graph nodes and create appropriate links between the newly created node and the existing tree or graph nodes. In the preferred embodiment, the left-hand portion of a semantic rule specifies characteristics that the existing node or nodes must have in order for the rule to be applied. The right-hand portion of the semantic rule specifies the type of new node to be created, and the values for the new node's attributes. The rules described in Figure 28 and in Figure 30 exemplify this form.

In the preferred embodiment of the present invention, each syntax parse tree and each logical form graph is represented as a list of nodes, with the

ᘔ\|

links between the nodes represented by attribute values within the nodes. Each set of rules is also represented as a list. Application of set of rules to a syntax parse tree involves selecting successive nodes from the list of nodes and attempting to apply to each selected node each rule from the list of rules representing the set of rules. A particular rule can be successfully applied to a node if that node has the characteristics specified in the left-hand portion of the rule. Occasionally, a new node may be created as a result of a successful rule application, or an existing node may be marked for deletion.

A flow diagram for the subroutine "apply_rules" which applies a set of rules to a list of nodes representing a syntax parse tree or logical form graph is displayed in Figure 31. The subroutine "apply_rules" is called by the NSS to apply each set of rules during each of the three phases of the NSS. In step 3101, apply_rules receives a list of nodes as its first argument and a list of rules as its second argument. Steps 3102 through 3107 represent an outer loop, each iteration of which attempts to apply all of the input rules from the input list of rules to successive nodes selected from the input list. Steps 3103 through 3106 represent an inner loop, each iteration of which attempts to apply a rule selected from the list of input rules to a node selected from the input list of nodes. In step 3102, apply_rules selects the next node from the input list of nodes, starting with the first. In step 3103, apply_rules selects the next rule from the input list of rules, starting with the first. In step 3104, apply_rules determines whether the selected node has the characteristics specified in the left-hand part of the selected rule. If the node has the specified characteristics, then apply_rules applies in step 3105 the selected rule to the selected of node. If apply_rules determines in step 3106 that there are more rules to attempt to apply

22

to the selected node, apply_rules returns to step 3103 to select the next rule. If apply_rules determines in step 3107 that there are more nodes to attempt to apply the rules of the input rule list, apply_rules returns to step 3102 to select the next node.

A flow diagram for the processing done in the first phase of the NSS is displayed in Figure 32. In step 3201, the variable "parameter1" is assigned to be the list of syntax parse tree nodes that comprise the syntax parse tree generated by the syntactic subsystem and input into the NSS. In step 3202, the variable "parameter2" is assigned to be a list of the first set of semantic rules displayed in Figure 27. In step 3203, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the first set of semantic rules to the syntax parse tree to effect preliminary adjustments. In step 3204, the variable "parameter1" is assigned to be the list of syntax parse tree nodes that comprise the preliminarily-adjusted syntax parse tree. In step 3205, the variable "parameter2" is assigned to be a list of the second set of semantic rules displayed in Figure 29. In step 3206, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the second set of semantic rules to the syntax parse tree to effect main adjustments.

NSS Phase Two - Generating an Initial Logical Form Graph

In phase two of the NSS, the NSS applies a third set of semantic rules to the nodes of the adjusted syntax tree. Each successful rule application in phase two creates a new logical form graph node. By applying this third set of

23

rules, the NSS creates a new logical form graph. The nodes of the logical form graph consist of only semantically meaningful attributes and a pointer back to the corresponding syntax tree node. Unlike in prior art semantic subsystems, the logical form graph nodes created by the NSS in phase two are completely separate and distinct from the syntax parse tree nodes. The NSS constructs a skeleton of the logical form graph that comprises links, stored as attributes within the nodes, that interconnect the nodes of the logical form graph.

In Figure 33, a list of the third set of semantic rules applied by the NSS in phase two is displayed. For each rule, Figure 33 displays the name of the rule followed by a concise description of the linguistic phenomenon that it addresses. There are only three rules in this third set of rules, and only the first rule, SynToSem1, is commonly used. The second and third rules apply only to special situations when a fitted parse was generated by the syntactic subsystem, and the adjusted syntax parse tree therefore contains a fitted parse node.

Figures 34A-34C display an English-language representation of the semantic rule SynToSem1 from the third set of semantic rules. As can be seen in Figures 34A-34C, the "If" expression concerns the values of various attributes for the syntax parse tree node to which the rule is applied and various related syntax parse tree nodes, and the "Then" expression specifies the creation of a logical form graph node and placement of the new node within the incipient logical form graph.

Figure 34D shows an example of the application of the semantic rule SynToSem1 to the syntax parse tree 3401 generated by the syntactic subsystem for the sentence "The book was written by John." Application of SynToSem1 results in the skeletal logical form graph 3402. The skeletal logical

24

form graph has three nodes with temporary modifiers labeling the links. Attributes have been assigned to the new nodes, based on the syntactic attributes of the syntax parse tree nodes from which they were created. There are far fewer nodes in the logical form graph than in the corresponding syntax parse tree, because the logical form graph represents the semantic meaning of the sentence. The linguistic significance of the words "the," "was," and "by" in the original sentence is or will be incorporated into the attributes and labels of the logical form graph, and the complex node hierarchies which emanated from their presence as leaf nodes in the syntax parse tree are not necessary in the logical form graph.

Figure 35 displays a flow diagram for phase two of the NSS. In step 3501, the variable "parameter1" is assigned the list of nodes representing the adjusted syntax parse tree. In step 3502, the variable "parameter2" is assigned to be a list of the third set of semantic rules displayed in Figure 33. In step 3503, the NSS invokes subroutine "apply_rules" to apply the third set of semantic rules to the nodes of the adjusted syntax parse tree, thereby creating a new logical form graph corresponding to the adjusted syntax parse tree.

NSS Phase Three - Completing the Logical Form Graph

In phase three of the NSS, the NSS applies a fourth set of semantic rules to the skeletal logical form graph to add semantically meaningful labels to the links of the logical form graph. These new labels include "deep subject" ("Dsub"), "deep object" ("Dobj"), "deep indirect object" ("Dind"), "deep predicate nominative" ("Dnom"), "deep complement" ("Dcmp"), and "deep predicate adjective" ("Dadj"). In Figures 36-38, a list of the fourth set of semantic rules applied by the NSS in phase three is displayed. For each rule, Figures 36-38 display the name of the rule followed by a concise description of the linguistic phenomenon that it addresses.

Figure 39A displays an English-language representation of the semantic rule LF_Dobj2 from the fourth set of semantic rules. As can be seen in Figure 39A, the "If" expression concerns the values of various attributes of the logical form graph node to which the rule is applied, and the "Then" expression specifies the labeling of a link in the logical form graph.

Figure 39B shows an example of the application of the semantic rule LF_Dobj2 to the logical form graph 3901 generated by the NSS for the sentence "The book was written by John." Application of LF_Dobj2 to a logical form graph containing a passive clause identifies the syntactic subject as the deep object of the action. This is accomplished, in Figure 39B, by relabeling link 3903 from a temporary modifier to the label 3904 indicating a deep object relationship.

As the final step in phase three, the NSS makes final adjustments to the logical form graph by applying a fifth set of semantic rules. This set of rules include rules that serve to unify a relative pronoun with its antecedent, find and explicitly include missing pronouns, resolve number ellipsis, provide missing deep subjects, unify redundant instances of personal pronouns, and contract coordinate structures expanded in the first sub-step of semantic analysis. These rules also deal with the problem of taking a pronoun (or "proform") and identifying the noun phrase to which it refers. In many cases, it is not possible to identify the correct noun phrase referent with the level of information that the logical form graph provides. In these cases, a list of the most likely candidates is created, and further processing is postponed until later steps of the NLP system that employ more global information. In Figure 40, a list of the fifth set of semantic rules applied by the NSS in phase three is displayed. For each rule, Figure 40 displays the name of the rule followed by a concise description of the linguistic phenomenon that it addresses.

Figures 41A-41C display an English-language representation of the semantic rule PsLF_PronAnaphora from the fifth set of semantic rules. As can be seen in Figures 41A-41C, the "If" expression concerns the values of various attributes of the logical form graph node to which the rule is applied, and of related logical form graph nodes, and the "Then" expression specifies the addition of a logical form graph node representing an omitted referent of a pronoun.

Figure 41D shows an example of the application of the semantic rule PsLF_PronAnaphora to the logical form graph 4101 generated by the NSS for the sentence "Mary likes the man who came to dinner, and Joan likes him too." Application of PsLF_PronAnaphora to a logical form graph containing a pronoun node with a referent in a different part of the logical form graph adds a new node to which the pronoun node is directly linked. In Figure 41D, the new node 4103 has been added by application of PsLF_PronAnaphora to indicate that the node "he1" refers to "man."

A flow diagram for the processing done in phase three of the NSS is displayed in Figure 42. In step 4201, the variable "parameter1" is assigned to be the list of logical form graph nodes that comprise the logical form graph generated during phase two of the NSS. In step 4202, the variable "parameter2" is assigned to be a list of the fourth set of semantic rules displayed in Figures 36-

27

38. In step 4203, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the fourth set of semantic rules to the logical form graph to add semantically meaningful labels to the links of the logical form graph. In step 4204, the variable "parameter1" is assigned to be the list of the logical form graph nodes that comprise the meaningfully-labeled logical form graph generated in step 4203. In step 4205, the variable "parameter2" is assigned to be a list of the fifth set of semantic rules displayed in Figure 40. In step 4206, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the fifth set of semantic rules to the logical form graph to effect final adjustments.

Figure 43 is a block diagram of a computer system for the NSS. The computer 4300 contains memory with the semantic rules 4304-4308 and rule application engine 4303. The rule application engine, under control of a central processing unit, applies the five sets of rules to the syntax parse tree 4301 to generate a corresponding logical form graph 4302. The syntax parse tree is preferably generated by the morphological and syntactic subsystems, which are not shown. The syntax tree and logical form graph can also be used to accomplish a subsequent task requiring information analogous to that which a human reader would obtain from the input sentences. For example, a grammar checker program might suggest a new phrasing for the input sentence that more accurately or concisely states what was stated in the input sentence. As another example, a computer operating system might perform computational tasks described by the input sentence. As still another example, information contained

VA

in the input sentence might be categorized and stored away for later retrieval by a database management system.

Semantic Processing of the Example Input Sentence

The following discussion and Figures 44-59 describe the complete NSS processing of the example sentence "The person whom I met was my friend." Each semantic rule that is applied by the NSS will be described, along with a representation of the results of the rule application.

No preliminary adjustment rules from the first set of semantic rules are successfully applied to the syntax parse tree input into the NSS from the syntactic subsystem during phase one. One main adjustment rule from the second set of semantic rules is applied to the input syntax parse tree. Figure 44 displays the syntax parse tree 4400 in the form it is input. Note that it is represented in Figure 44 slightly more simply than in Figure 22. The NSS successfully applies the semantic rule TrLF_LongDist1, displayed in Figure 29 as rule 1, to the relative clause node RELCL1, 4401, of the syntax parse tree 4400 to generate the adjusted syntax parse tree 4402. The effect of applying rule TrLF_LongDist1 is the introduction of a direct object attribute in the noun phrase node 4403 to indicate that the word "whom" is the direct object of the phrase "I met." Normally, in English, the direct object of a verb follows the verb. Because "whom" does not follow "I met" in the sentence that was parsed to produce the syntax tree 4400, the fact that "whom" is the direct object of "I met" was not identified by the application of syntactic rules.

Seven rules from the third set of semantic rules are successfully applied in phase two of the NSS. In Figure 45, the NSS successfully applies the

24

semantic rule SynToSem1, displayed in Figure 33 as rule 1, to the determinate pronoun node DETP2, 4501, of the syntax parse tree to generate the logical form graph node "my" 4502. In Figure 46, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP4, 4601, of the syntax parse tree to generate the logical form graph node "friend" 4602 and the link 4603 with the temporary semantic label "Tmods" 4604. In Figure 47, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP3, 4701, of the syntax parse tree to generate the logical form graph node "I" 4702. In Figure 48, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP2, 4801, of the syntax parse tree to generate the logical form graph node "whom" 4802. In Figure 49, the NSS successfully applies the semantic rule SynToSem1 to the relative clause node RELCL1, 4901, of the syntax parse tree to generate the logical form graph node "meet" 4902 and the link 4903 with the temporary semantic label "Tmods" 4904. In Figure 50, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP1, 5001, of the syntax parse tree to generate the logical form graph node "person" 5002 and the link 5003 with the temporary semantic label "Tmods" 5004. In Figure 51, the NSS successfully applies the semantic rule SynToSem1 to the declarative sentence node DECL1, 5101, of the syntax parse tree to generate the logical form graph node "be" 5102 and the link 5103 with the temporary semantic label "Tmods" 5104. Thus, with the completion of phase two of the NSS, a skeletal logical form graph has been created.

Six rules from the fourth set of semantic rules are successfully applied in phase three of the NSS. In Figure 52, the NSS successfully applies the semantic rule LF_Dsub1, displayed in Figure 36 as rule 1, to the logical form

30

graph node "be" 5201 to generate the link label "Dsub" 5202 and the link 5203 with the temporary semantic label "Tmods" 5204. In Figure 53, the NSS successfully applies the semantic rule LF_Dnom, displayed in Figure 36 as rule 10, to the logical form graph node "be" 5301 to generate the link label "Dnom" 5302. In Figure 54, the NSS successfully applies the semantic rule LF_Props, displayed in Figure 38 as rule 21, to the logical form graph node "person" 5401 to generate the link label "Props" 5402. In Figure 55, the NSS successfully applies the semantic rule LF_Dsub1, displayed in Figure 36 as rule 1, to the logical form graph node "meet" 5501 to generate the link label "Dsub" 5502. In Figure 56, the NSS successfully applies the semantic rule LF_Dobj1, displayed in Figure 36 as rule 3, to the logical form graph node "meet" 5601 to generate the link labeled "Dobj" 5603 to link the node "meet" to the node "whom" 5602. In Figure 57, the NSS successfully applies the semantic rule LF_Ops, displayed in Figure 38 as rule 22, to the logical form graph node "friend" 5701 to generate the link label "PossBy" 5702.

One rule from the fifth set of semantic rules is successfully applied in phase three of the NSS. In Figure 58, the NSS successfully applies the semantic rule PsLF_RelPro, displayed in Figure 40 as rule 1, to the logical form graph node "whom," displayed as 5602 in Figure 56, to generate the link labeled "Dobj" 5801 and to remove the node "whom." In Figure 59, the NSS successfully applies the semantic rule PsLF_UnifyProns, displayed in Figure 40 as rule 10, to the logical form graph to consolidate the nodes "I" and "my" into a single node. This is the last rule applied successfully by the NSS. Figure 59 thus displays the final, complete logical form graph generated by the NSS for the input sentence "The person whom I met was my friend."

31

Although the present invention has been described in terms of a preferred embodiment, it is not intended that the invention be limited to this embodiment. Modifications within the spirit of the invention will be apparent to those skilled in the art. The scope of the present invention is defined by the claims that follow.

32

Claims

1. A method in a computer system for generating a logical form graph for a sentence in a natural language, the sentence being represented by a syntax parse tree having nodes representing syntactic constructs of the sentence, the syntax parse tree being represented in a data structure, the method comprising:

adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence;

adjusting the syntax parse tree with the added syntactic roles to represent a complete syntactic analysis of the sentence;

generating a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is separate from the data structure of the syntax parse tree;

adding semantic labels to the generated skeletal logical form graph; and

adjusting the logical form graph with semantic labels to add semantic constructs to complete the logical form graph.

2. The method of claim 1 wherein when the sentence omits a verb after a predefined word, the step of adding syntactic roles adds a syntactic construct for the omitted verb.

3. The method of claim 2 wherein the predefined word is the word "to."

4. The method of claim 2 wherein the predefined word is the word "not."

5. The method of claim 1 wherein when the sentence is missing a pronoun, the step of adding syntactic roles adds a syntactic construct for the missing pronoun.

6. The method of claim 5 wherein the missing pronoun is the word "you" in an imperative sentence.

7. The method of claim 1 wherein when the sentence includes coordinate structures, the step of adding syntactic roles adds a syntactic construct to expand the coordinate structure.

8. The method of claim 7 wherein the coordinate structures include the word "and."

9. The method of claim 7 wherein the coordinate structures include the word "or."

10. The method of claim 1 wherein the step of adjusting the syntax parse tree includes resolving long-distance attachment phenomena.

11. The method of claim 1 wherein the step of adjusting the syntax parse tree includes transforming verbal phrases into verbs with prepositional phrase objects.

12. The method of claim 1 wherein the step of adjusting the syntax parse tree includes replacing the word "it" with and infinitive clause.

13. The method of claim 1 wherein the step of generating the skeletal logical form graph includes assigning attributes to nodes of the skeletal logical form graph based on the attributes of the adjusted syntax parse tree.

14. The method of claim 1 wherein the step of adding semantic labels includes adding semantic labels indicating deep parts of speech.

15. The method of claim 14 wherein the deep part of speech is a subject.

16. The method of claim 14 wherein the deep part of speech is an object.

17. The method of claim 14 wherein the deep part of speech is an indirect object.

18. The method of claim 14 wherein the deep part of speech is a predicate nominative.

19. The method of claim 14 wherein the deep part of speech is a complement.

20. The method of claim 14 wherein the deep part of speech is a predicate adjective.

21. A method in a computer system for generating a logical form graph for a phrase of words specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the method comprising:

generating an initial syntax parse tree of the phrase based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of the words of the phrase;

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the phrase;

generating a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph.

22. The method of claim 21 wherein the step of adjusting the initial syntax parse tree includes adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the phrase.

23. The method of claim 21 wherein the step of adjusting the skeletal logical form graph includes adding semantic labels to the generated skeletal logical form graph.

24. A computer-readable medium containing instructions for causing a computer system to generate a logical form graph for a sentence specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the computer system having an initial syntax parse tree of the sentence that represents a parse of the sentence based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of words of the sentence, by:

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the sentence;

generating a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph for the sentence.

25. The computer-readable medium of claim 24 wherein the adjusting of the initial syntax parse tree includes adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence.

26. The computer-readable medium of claim 24 wherein adjusting of the skeletal logical form graph includes adding semantic ~~labels to the~~ generated skeletal logical form graph.

27. A computer system for generating a logical form graph for a sentence in a natural language, the sentence being represented by a syntax parse tree having nodes representing syntactic constructs of the sentence, the syntax parse tree being represented in a data structure, the method comprising:

a phase one component for adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence and for adjusting the syntax parse tree with the added syntactic roles to represent a complete syntactic analysis of the sentence;

a phase two component for generating a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is separate from the data structure of the syntax parse tree, the logical form graph having nodes and links, the nodes corresponding to semantic constructs and the links corresponding to relationships between semantic constructs; and a phase three component for adding semantic labels to the generated skeletal logical form graph and for adjusting the logical form graph with semantic labels to add semantic constructs to complete the logical form graph.

28. A method in a computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the method comprising the steps of:

generating a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

generating a separate logical form graph to represent a semantic analysis of the input text.

29. A computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the system comprising:

a component that generates a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

a component that generates a separate logical form graph to represent a semantic analysis of the input text, wherein the logical form graph comprises nodes and directional links.

30. The system of claim 29 wherein the component that generates a separate logical form graph comprises the following sub-components:

a first sub-component that generates an initial skeletal logical form graph; and

a second sub-component that identifies semantic roles for the nodes of the skeletal logical form graph and labels the directed links of the skeletal logical form graph to produce a final, complete logical form graph.

31. A computer system for processing a syntax parse tree representing a syntactic analysis of input text constituting a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, wherein the syntax parse tree comprises a set of nodes and directed edges linking the nodes, the system comprising:

a rule processing engine for applying semantic rules to generate a separate logical form graph from the syntax parse tree, wherein the logical form graph comprises nodes and directed links; and

a set of semantic rules.

32. The computer system of claim 31, wherein the set of semantic rules include a sub-set of semantic rules that add syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence.

33. The computer system of claim 32, wherein the set of semantic rules include a sub-set of semantic rules that adjust the syntax parse tree with the added syntactic roles to represent a complete syntactic analysis of the sentence.

34. The computer system of claim 31, wherein the set of semantic rules include a sub-set of semantic rules that generate a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is separate from the data structure of the syntax parse tree.

35. The computer system of claim 34, wherein the set of semantic rules include a sub-set of semantic rules that add semantic labels to the generated skeletal logical form graph.

36. The computer system of claim 35, wherein the set of semantic rules include a sub-set of semantic rules that adjust the logical form graph with semantic labels to add semantic constructs to complete the logical form graph.

add A4

add C1

674610

METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Abstract of the Disclosure

Methods and computer systems for semantically analyzing natural language sentences. The natural language processing subsystems for morphological and syntactic analysis transform an input sentence into a syntax parse tree. Semantic analysis applies three sets of semantic rules to create a skeletal logical form graph from a syntax parse tree. Semantic analysis then applies two additional sets of semantic rules to provide semantically meaningful labels for the links of the logical form graph, to create additional logical form graph nodes for missing elements, and to unify redundant elements. The final logical form graph represents the complete semantic analysis of an input sentence.

WPN/RWB/661005/447-AP/V9

08/674610

704/9
Thomas
2747

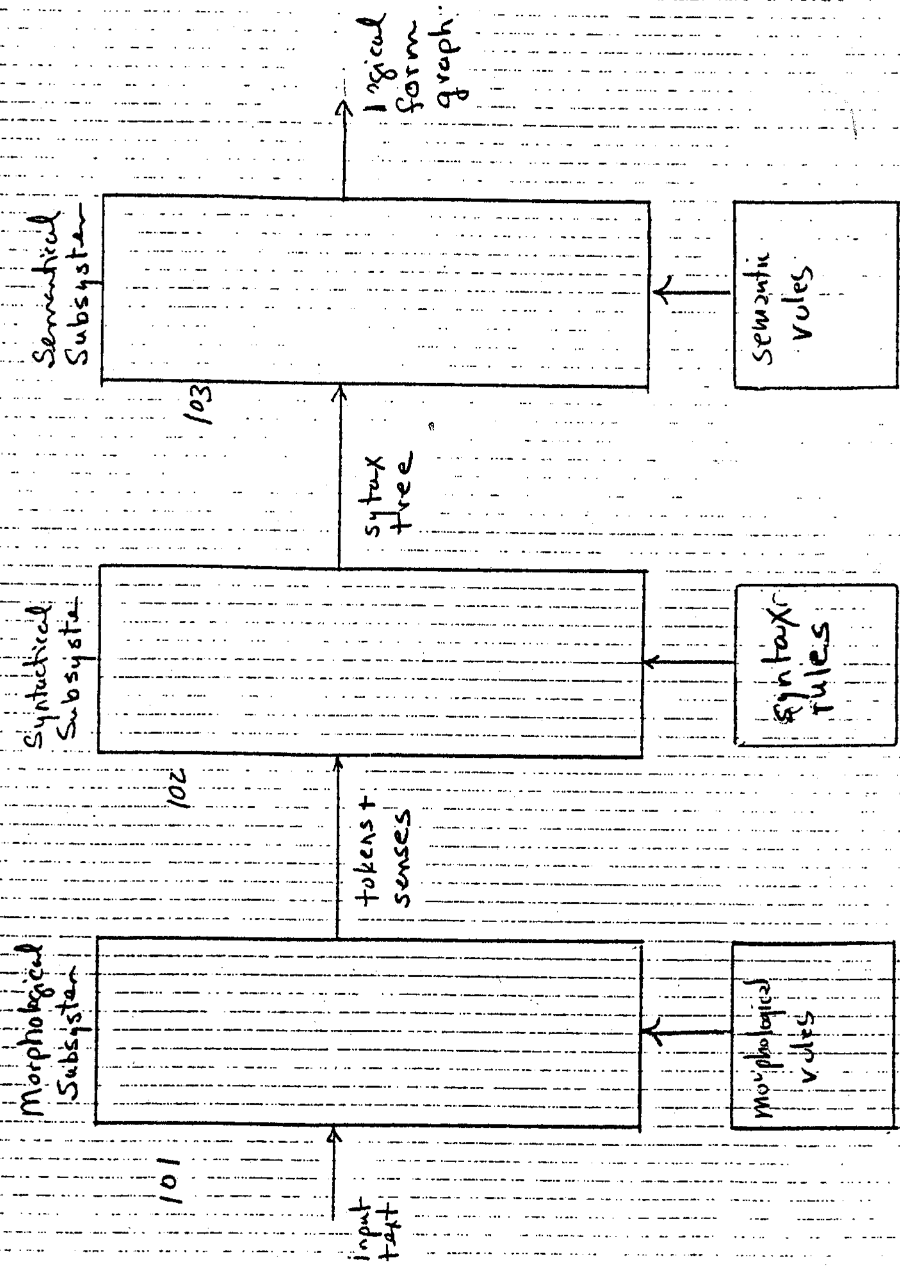


Figure 1

69 figs

Figure 2

08/674610

203

```
the
{        204
Adj          205
       {Lemma        "the"
        Bits         Sing Plur Wa6 Det Art
          212        B0 Def}   206
Adv
       {Lemma        "the"
        Bits         Wa5}
Senses   208
       {Bits         Sing Plur Wa6 Closed
                     Det Art Def
207     Lemma        "the"
        Cat          Adj
        Infl         Adj-nil
        Defin        "used when it is clearly understood who or what is meant"
        Exs          "We have a cat and a dog. The cat (= our cat) is black and the dog (=
our dog) white."
                     "the history of China (= Chinese history)"
                     "The Danes that I know work very hard."
                     "Take these letters to the post office (it is understood that you know
which post office and where it is)"}  209
    210 {Lemma       "the"
        Cat          Adv
        Defin        "To that extent; by that much"
        Exs          "the sooner the better."}  211

      (more sense records)  213
}
```

202

```
person
{
Noun
       {Lemma        "person"
        Bits         Pers3 Sing Humn Mass
                     Anim Count Conc C9
                     Humn_sr
        Infl         Noun-default }
 Senses
       {Lemma        "person"
        Cat          Noun
        Defin        "A living human being."
        Exs          "chairperson"
                     "spokesperson"
                     "salesperson."}
      (more sense records)

}
```

Figure 3


```
whom
{Pron
        {Lemma      "who"
         Bits       Pers3 Sing Plur Rel Wh
                    Humn Obj Anim}
Senses
        {Lemma      "who"
         Bits       Pers3 Sing Plur Rel Wh
                    Closed Humn Obj Anim
         Cat        Pron
         Defin      "(the object form of who, used esp. in writing and careful speech)"
         Exs        "With whom?"
                    "The man with whom he talked."
                    "You saw whom?"
                    "Whom did they see?"
                    "the man (whom) they saw arriving"
                    "a man (whom) you may know of"}
        (more sense records)

}
```

```
i
{
 Noun
        {Lemma      "i"
         Bits       Pers3 Sing TakesAn
         Infl       Noun-irreg}
Pron
        {Lemma      "I"
         Bits       Sing Nom TakesAn Pers1
                    Humn Anim LexCap}
Senses
        {Lemma      "i"
         Cat        Noun
         Infl       Noun-irreg
         Defin      "The ninth letter of the modern English alphabet."}

        {Lemma      "I"
         Cat        Pron
         Defin      "Used to refer to oneself as speaker or writer."}
        (more sense records)
}
```

```
met
{
 Verb
        {Lemma      "meet"
         Bits       Sing Plur Past
                    Pastpart
         Infl       Verb-meet }
Senses
        {Lemma      "meet"
         Bits       Past Pastpart
         Cat        Verb}

  }
```

Figure 4

08/674610

```
was
{
 Verb
       {Lemma       "be"
        Bits        Pers3 Sing Past Pers1
        Infl        Verb-be }}
 Senses
       {Lemma       "be"
        Bits        Past Pastpart
        Cat         Verb}
      (more sense records)
}
```

```
my
{
 Adj
       {Lemma       "I"
        Bits        Wa5 Det Poss Pers1 Def
                    Gen A0
        Infl        Adj-none }
 Ij
       {Lemma       "my" } }
 Senses
       {Lemma       "I"
        Bits        Wa5 Closed Det Poss
                    Pers1 Def Gen A0
        Cat         Adj
        Infl        Adj-none
        Defin       "belonging to me"
        Exs         "my car"
                    "my mother"}

       {Cat         Ij
        Defin       "Used as an exclamation of surprise, pleasure, or dismay"
        Exs         "Oh, my! What a tiring day!"}

      (more sense records)

}
```

Figure 5

09/674610

```
friend
{
 Noun
       {Lemma       "friend"
        Bits        Pers3 Sing Humn Anim
                    Count Conc Humn_sr NO
                    Wrdy
        Infl        Noun-default
        Vprp        (of to)
        Bitrecs
              {Bits       Humn Count Conc
               Vprp       (of) }

              {Bits       Humn Count Conc
               Vprp       (to) } }
 Verb
       {Lemma       "friend"
        Bits        Inf Plur Pres T1
        Infl        Verb-default } }
Senses
       {Lemma       "friend"
        Bits        Humn Conc
        Cat         Noun
        Defin       "A person whom one knows, likes, and trusts."}

       {Bits        T1
        Lemma       "friend"
        Cat         Verb
        Infl        Verb-default
        Defin       "To befriend."}

     (more sense records)

}
```

Figure 6

09/574610

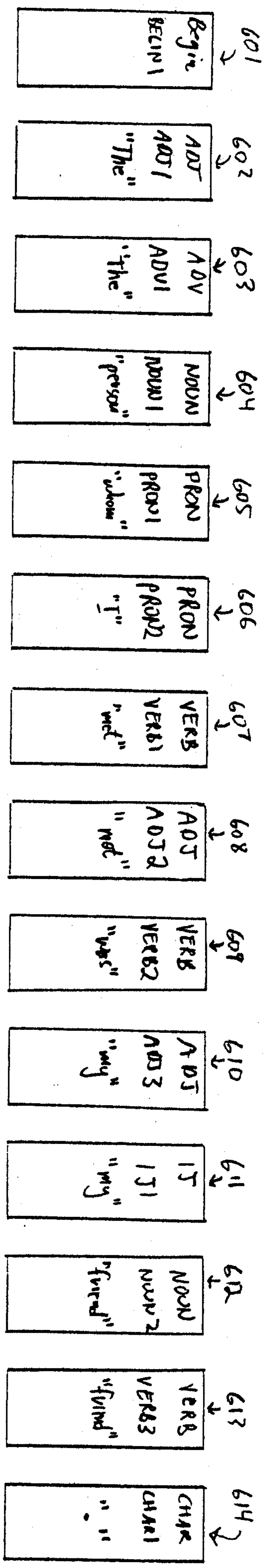
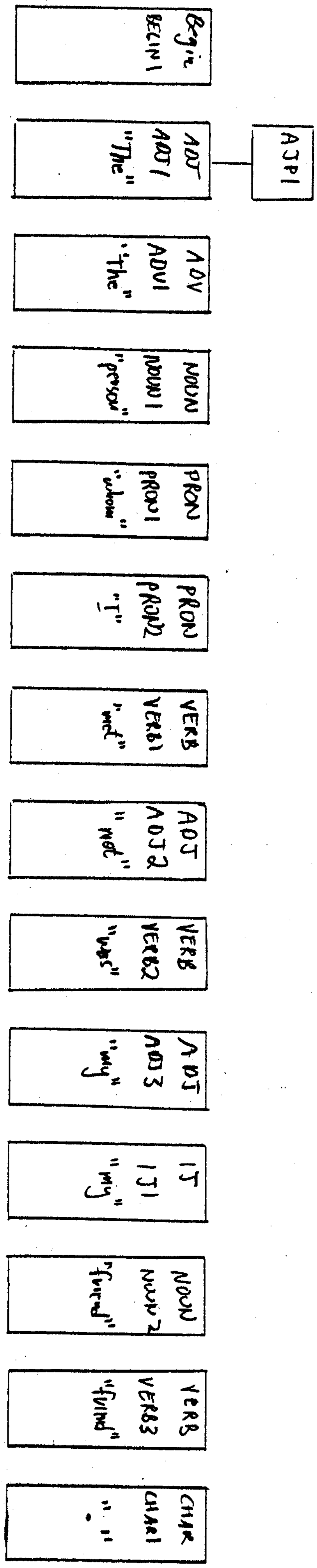


Figure 7

Rule: Adjective to Adjective Phrase

ADJ1 $\longrightarrow$ AJP1

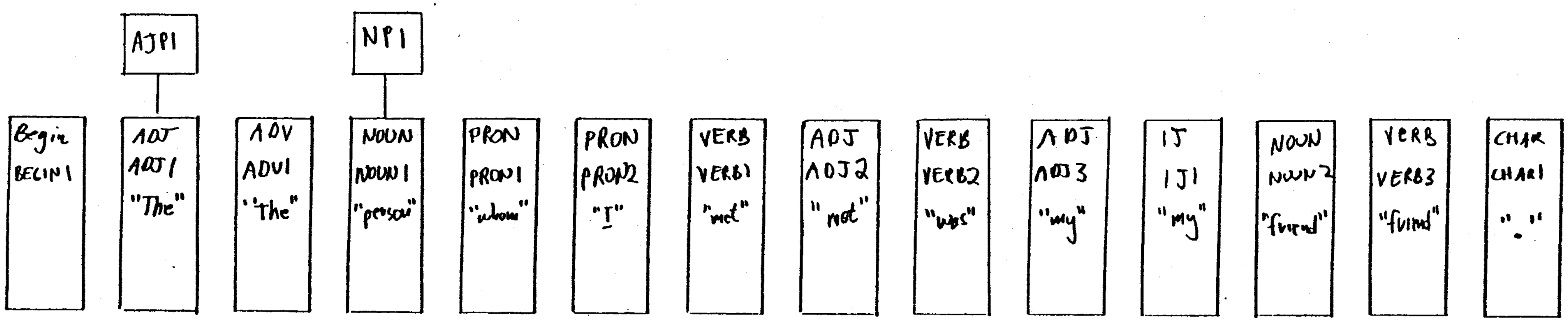


08/674610

Figure 8

Rule: Noun to Noun Phrase

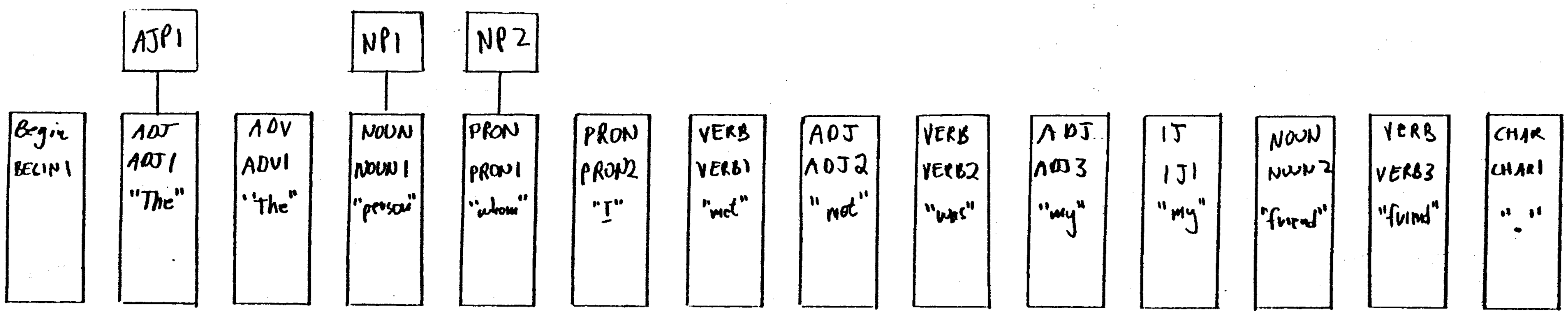
NOUN1 → NP1



08/674610

Figure 9

Rule: Pronoun to Noun Phrase
PRON1 → NP2

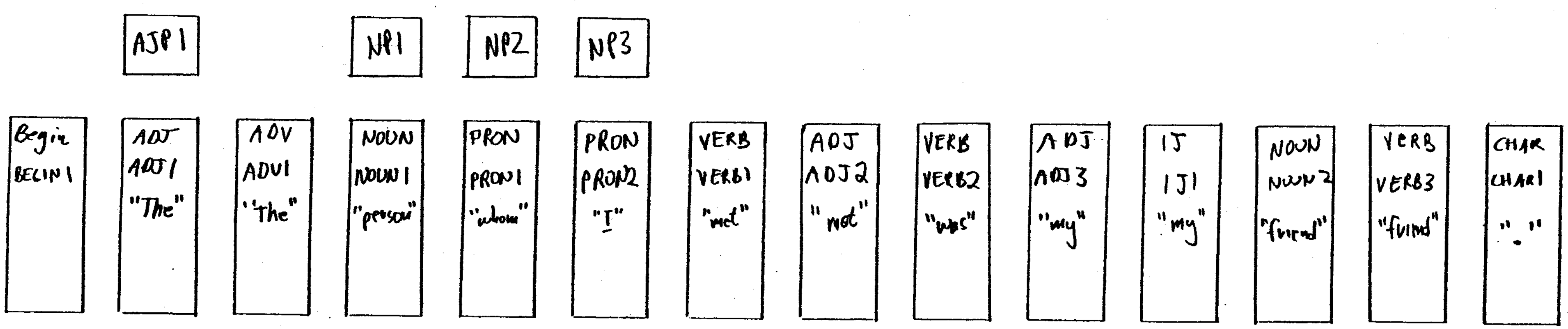


08/674610

Figure 10

Rule: Pronoun to Noun Phrase

PRON2 → NP3

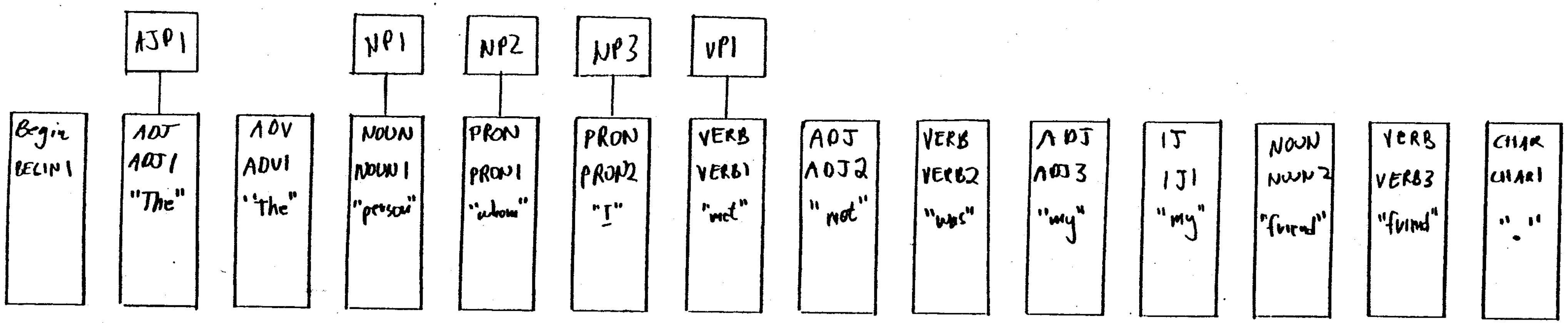


08/674610

Figure 11

Rule: Verb to Verb Phrase

VERB1 → VP1

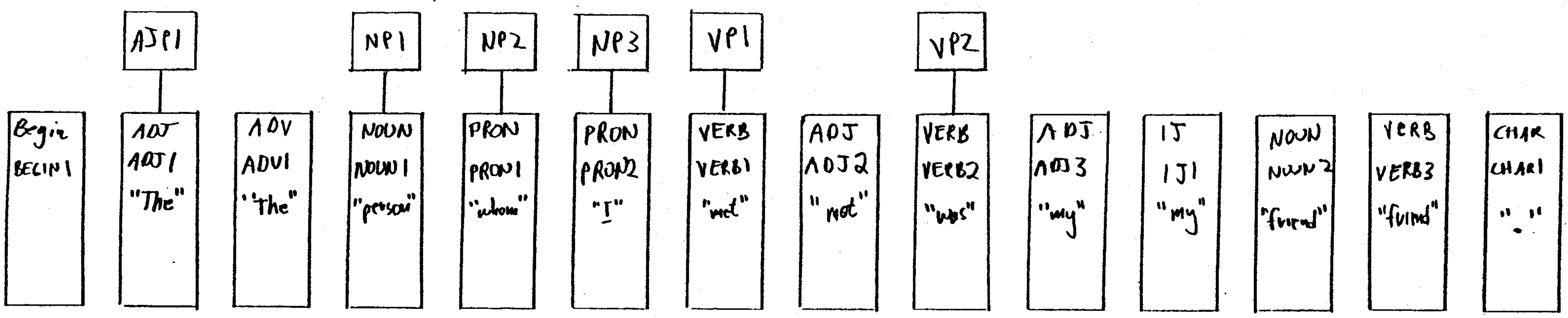


08/674610

Figure 12

Rule: Verb to Verb Phrase

VERB2 → VP2

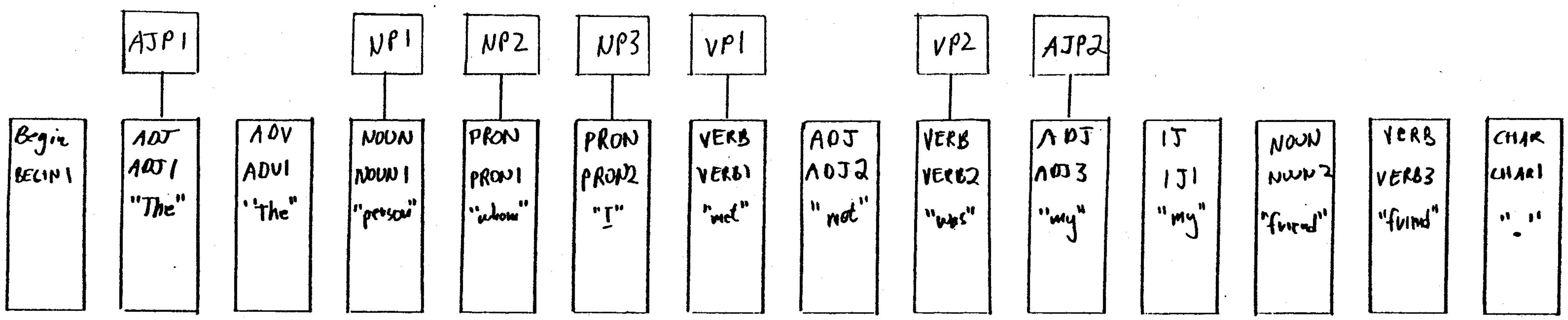


08/674610

Figure 13

Rule: ADJECTIVE to ADJECTIVE PHRASE

ADJ3 $\rightarrow$ AJP2



08/674610

Figure 14

Rule: Noun to Noun phrase

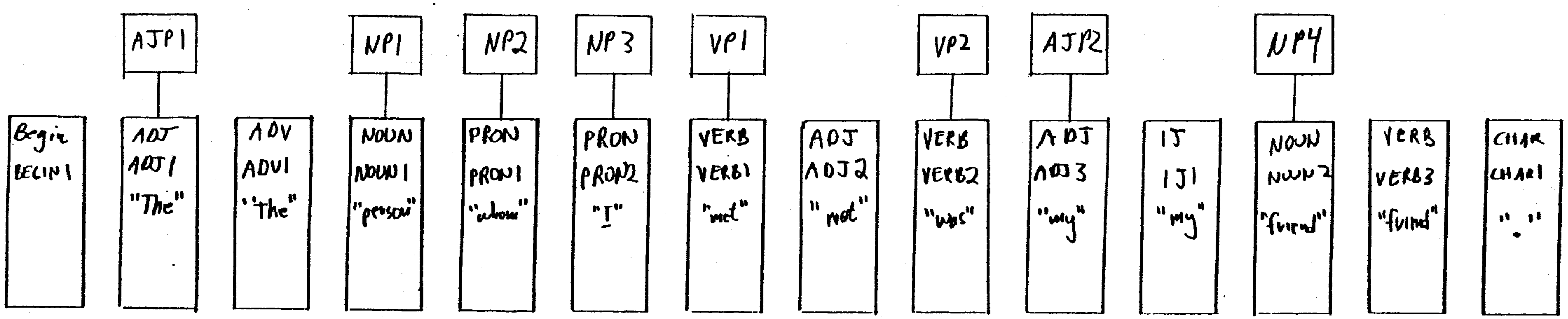


Figure 15

08/674610

Rule: Noun Phrase with Determiner

AJP2 , NP4 → NP5

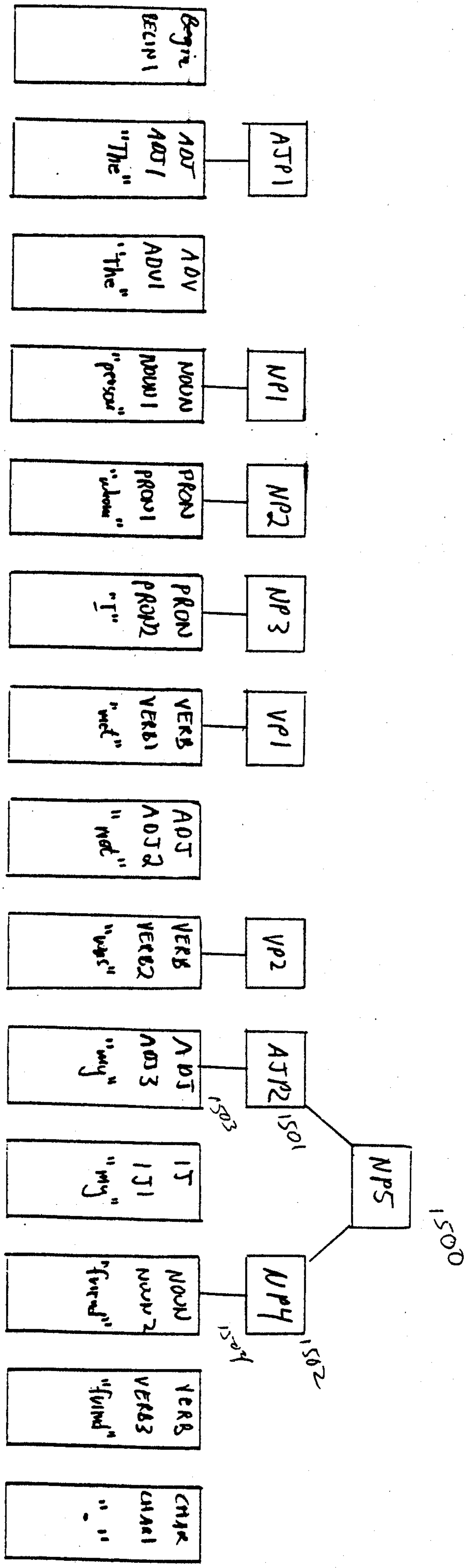
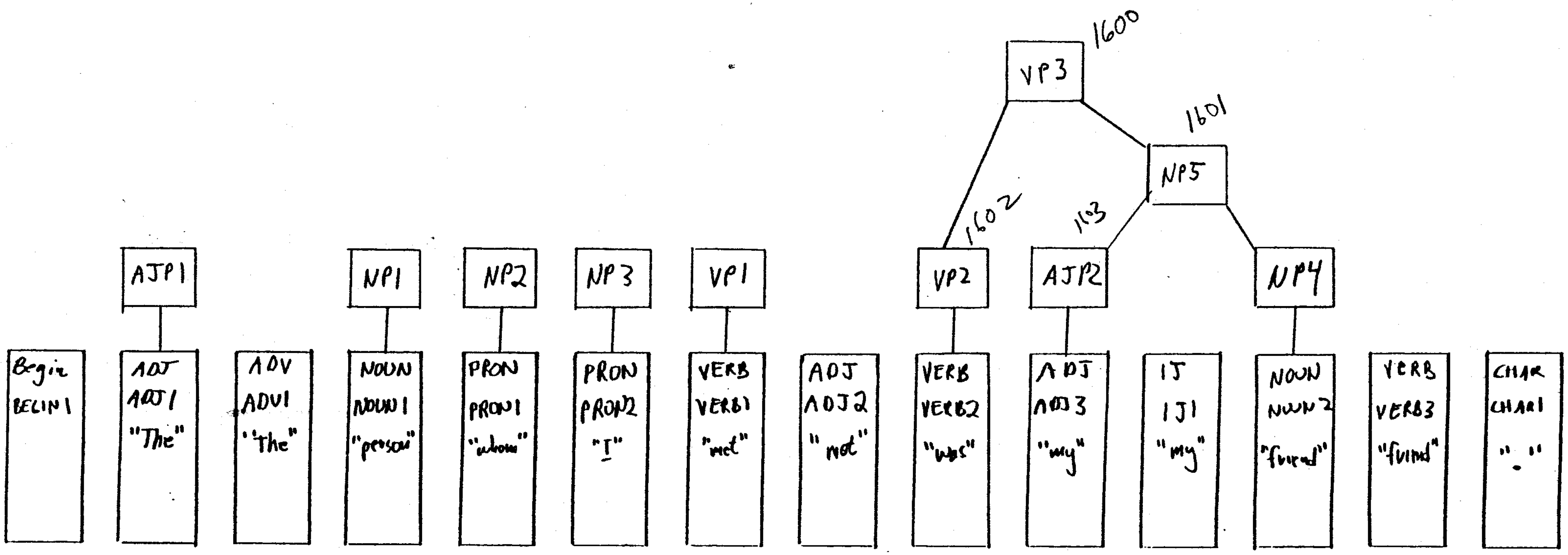


Figure 16

Rule: Verb Phrase with Noun Phrase as Object of Transitive Verb

VP2, NP5 → VP3



08/674610

Figure 17

Rule: Verb Phrase with Noun Phrase as Subject

NP3; VP1 $\rightarrow$ VP4

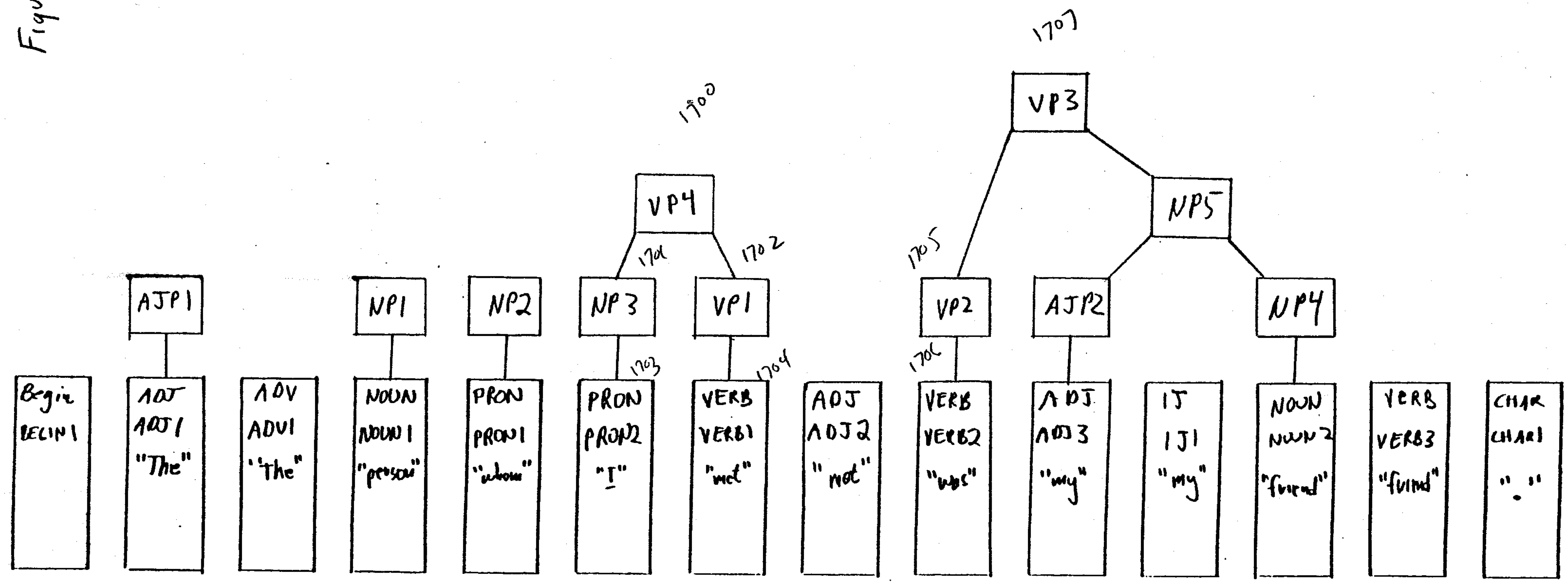
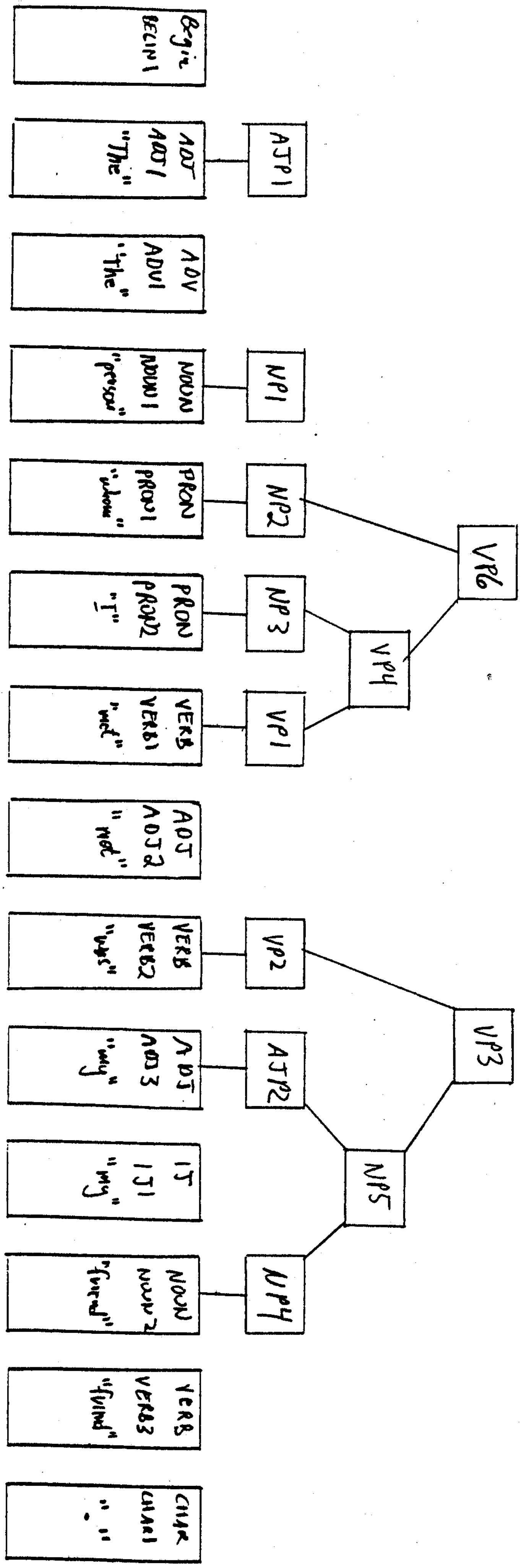


Figure 18

08/674610



Rule: Topicalization
NP2, VP4 → VP6

Rule: Noun Phrase with Relative Clause

NP1, VP6 → NP9

Figure 19

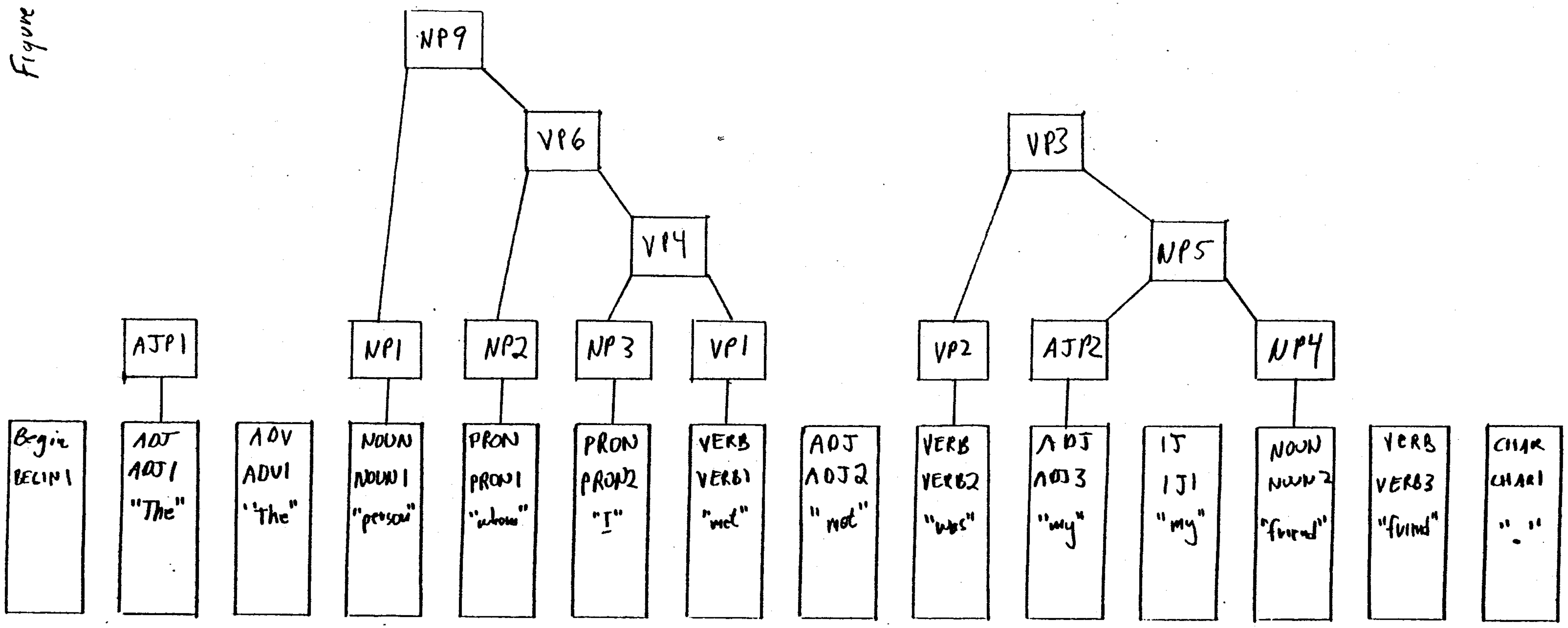


Figure 20

Rule: Noun Phrase with Determinate Quantifier

AJP1, NP9 → NP11

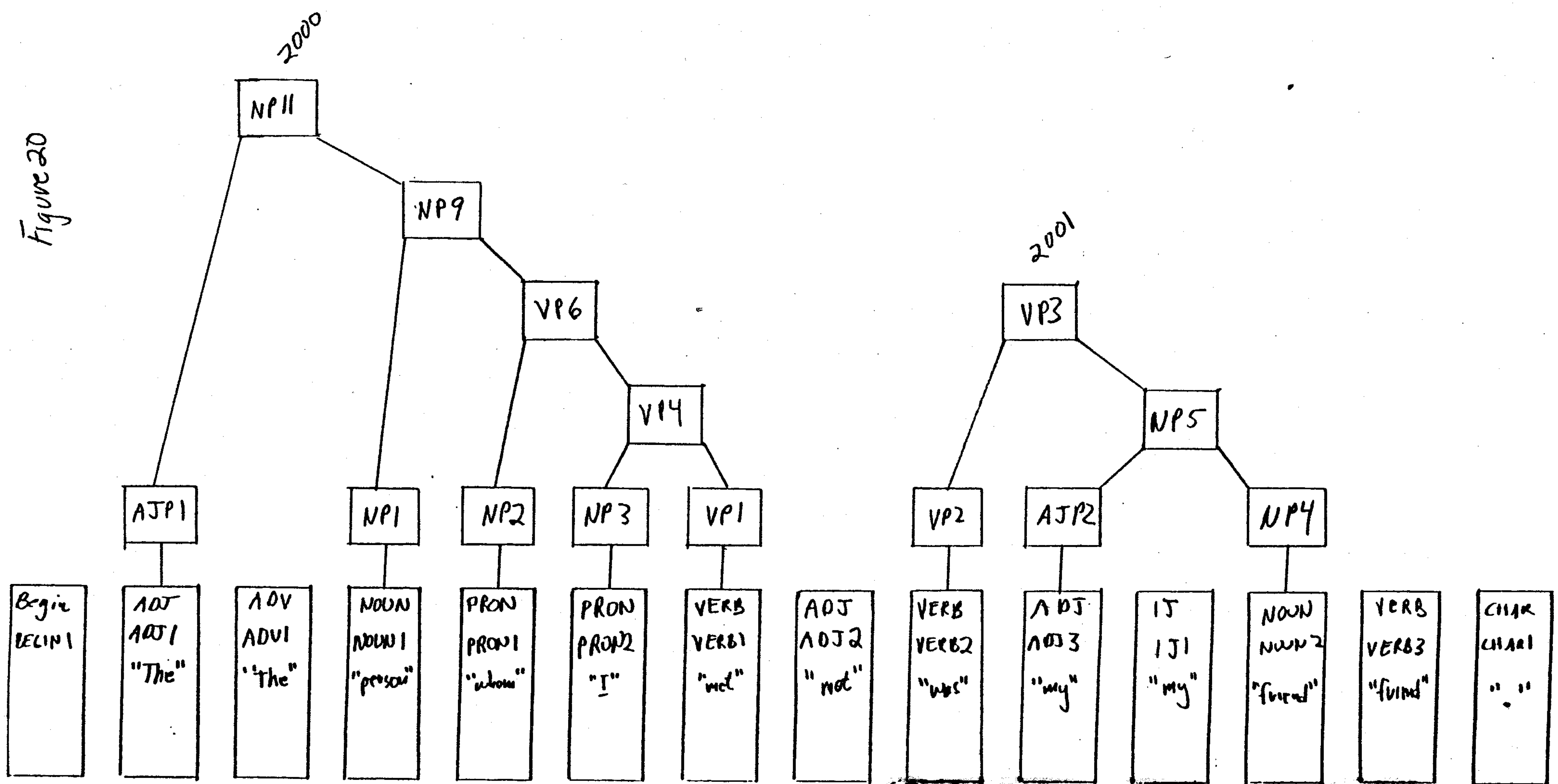


Figure 21

Rule: Verb Phrase with Noun Phrase Subject
NP11, VP3 → VP9

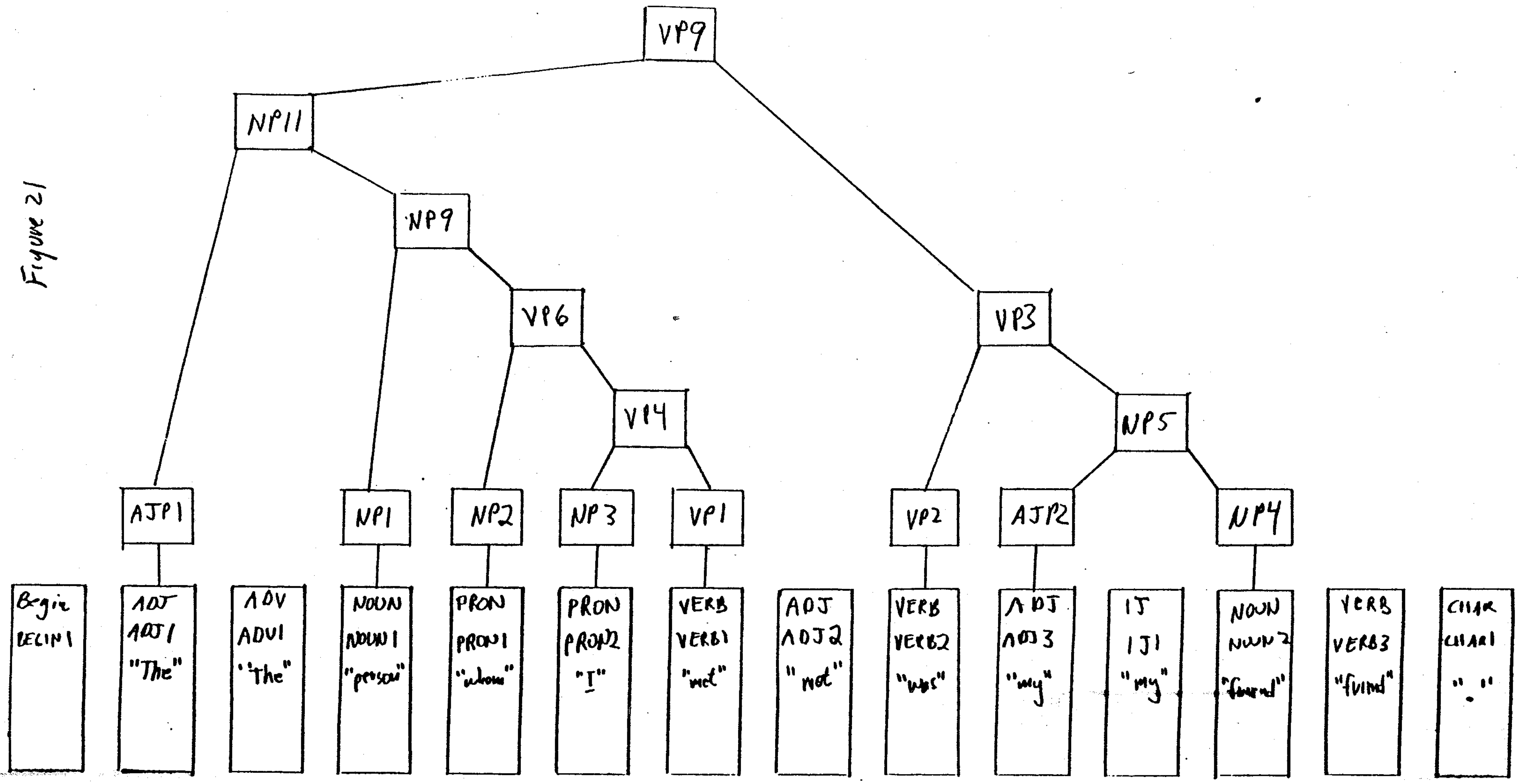


Figure 22

08/674610

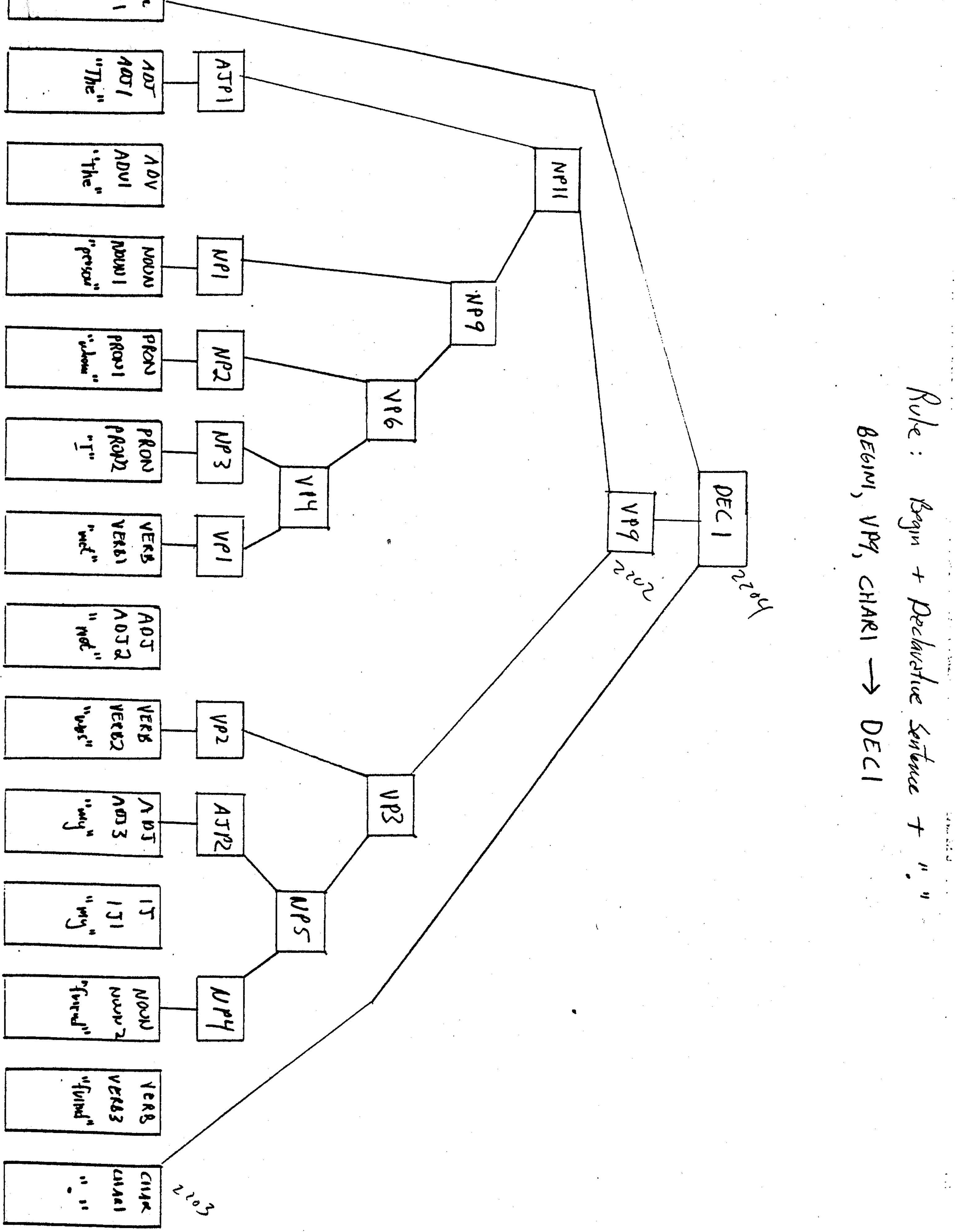
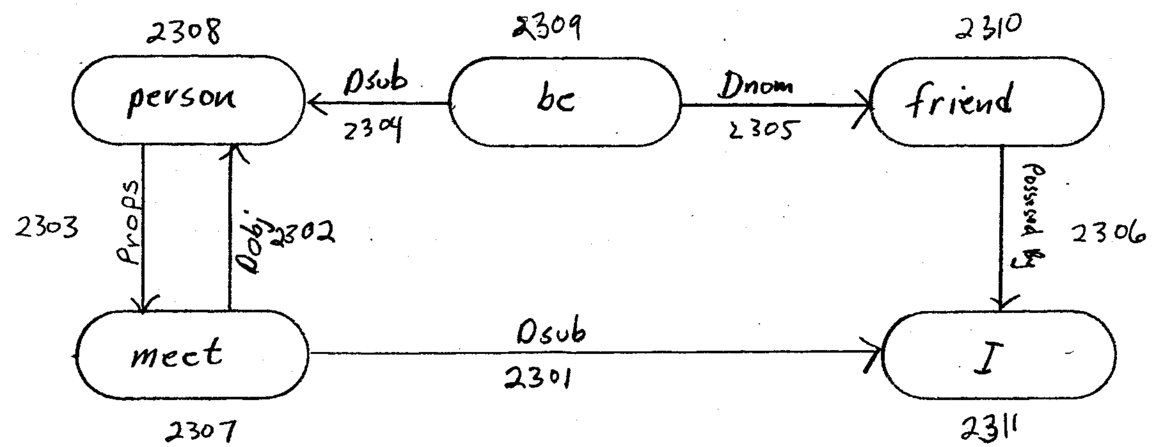


Figure 23



2308
person
2309
bc
2310
friend
Dsub
2304
Dnom
2305
2303
Props
Dobj
2302
Possessed by
2306
meet
Dsub
2301
I
2307
2311

Figure 24

08/874610

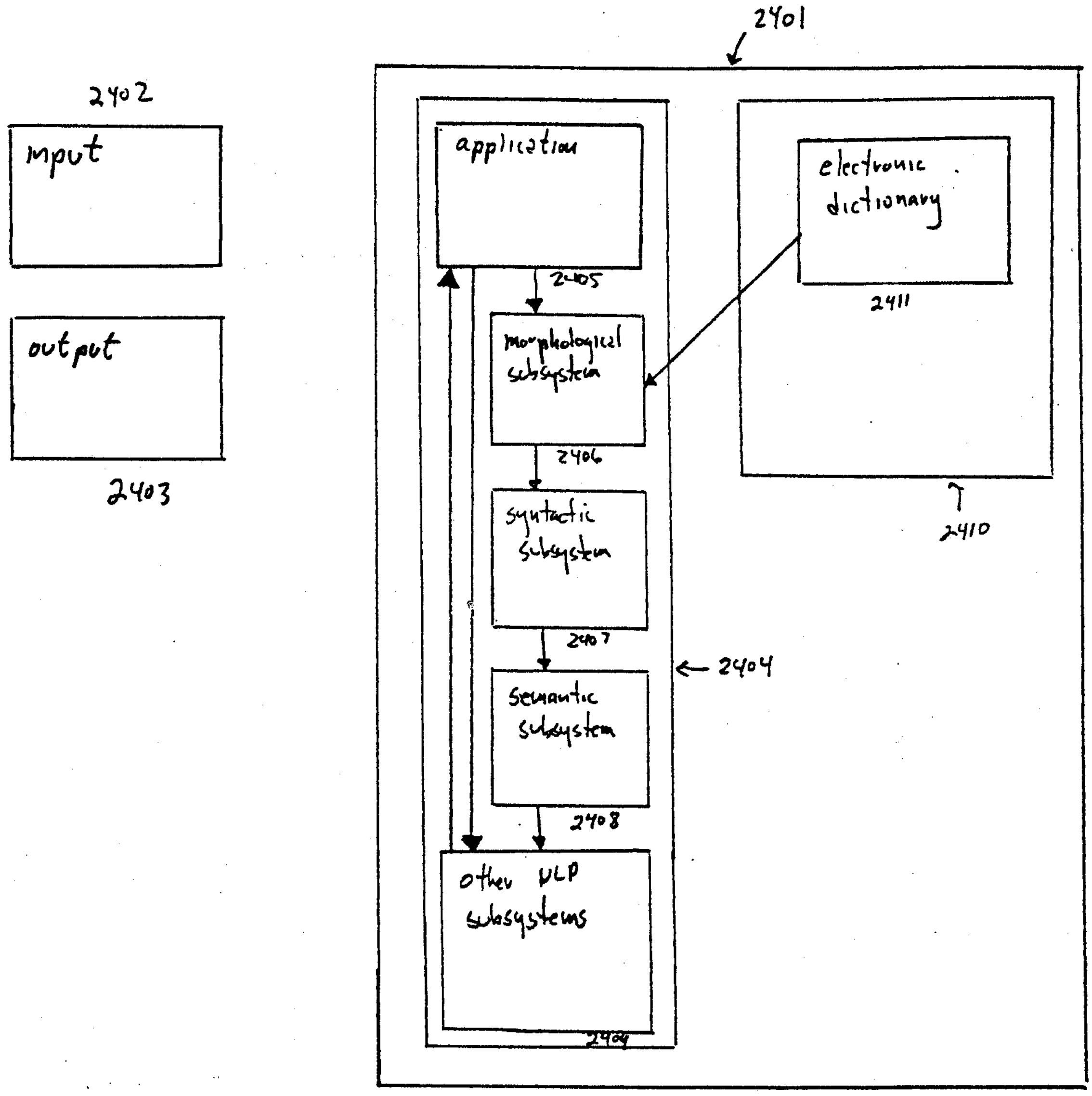
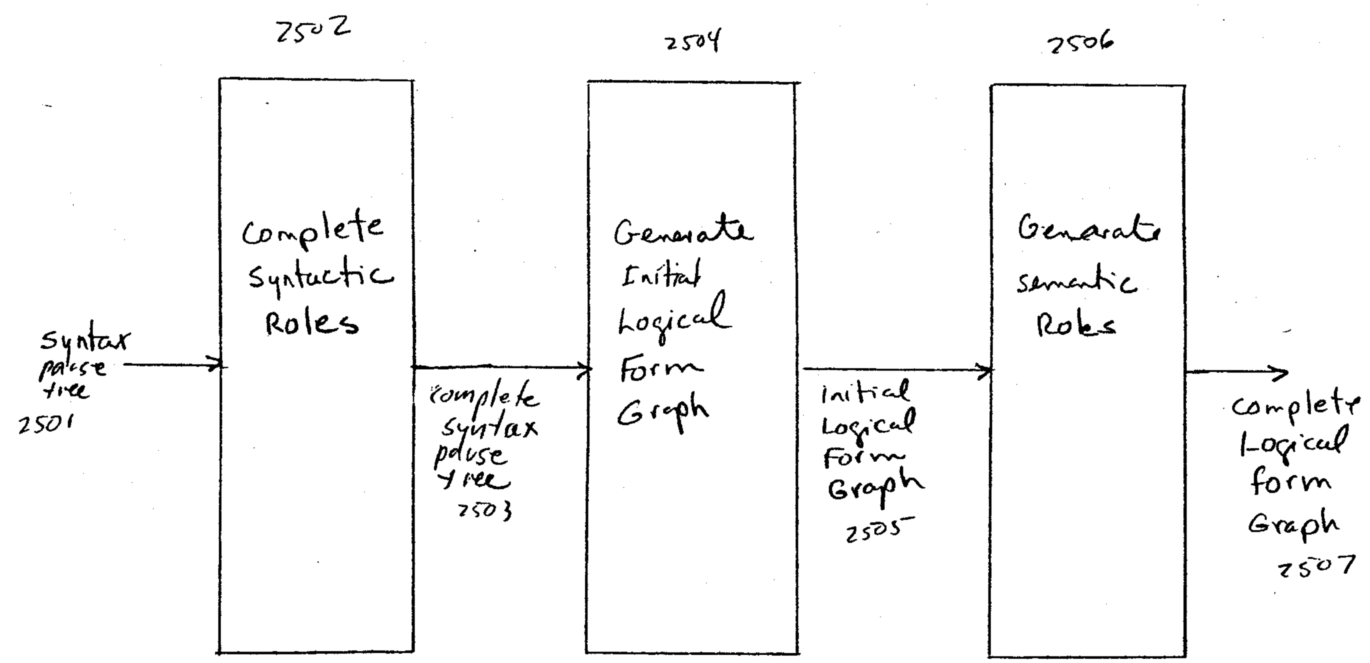


Figure 25

The New Semantic Subsystem



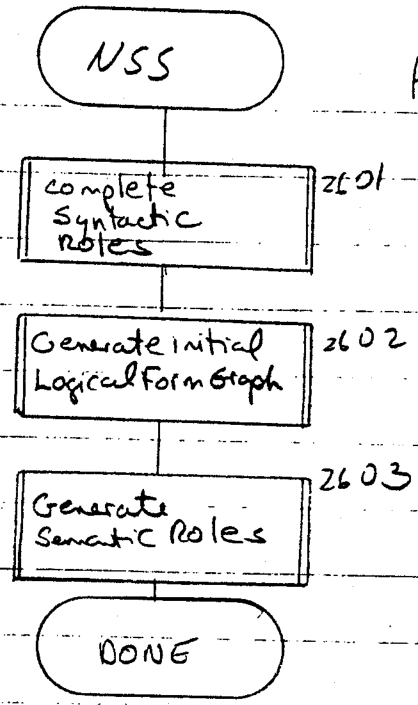

NSS
complete Syntactic Roles
2601
Generate initial Logical Form Graph
2602
2603
Generate Semantic Roles
DONE

Figure 26

08/674610

Figure 27

1. PrLF_NPQuantOf: for NPs like "a number of books," makes "books" the head and "a number of" the modifier

2. PrLF_PPQuantOf: same but for PPs, like "with a number of books"

3. PrLF_notAnaphora: prepares to fill VP anaphora like "John thought he would go but Jim thought not _________"

4. PrLF_soAnaphora: prepares to fill VP anaphora like "Mary wondered if it was true but Jane knew so _________"

5. PrLF_toAnaphora: prepares to fill VP anaphora like "Chris wanted to go but Pat didn't want to ____"

6. PrLF_You: supplies the understood "you" in commands like "(You) please close the door"

7. PrLF_HowAbout: supplies the understood "you" in constructions like "How about (you) closing the door"

8. PrLF_We: supplies the understood "we/us" in constructions like "Let's (us) go to the movies"

9. PrLF_I: supplies the understood "I" in, for example, "(I) Thank you" or "(I) Have not yet received your letter"

10. PrLF_SubjectMods: connects "we" and "all" in, e.g., "We are all reading the book"; connects "he" and "hungry" in, e.g., "He arrived hungry"

11. PrLF_RightShift: connects "the man" and "who was my friend" in, e.g., "The man arrived who was my friend"

12. PrLF_InfclPP: prepares for correct interpretation in constructions like "a person on whom to rely"

13. PrLF_QuantifierEllipsis: having to do with the resolution of pronoun references

14. PrLF_PossessivePronHead: having to do with the resolution of pronoun references

15. PrLF_PossibleCorefsOfProns: having to do with the resolution of pronoun references

16. PrLF_VPAnaphora: identifies and fills missing arguments in all cases of VP anaphora, e.g., "Sarah likes basketball and I do too"

17. PrLF_DistCoords: distributes elements across coordinated structures, like "They washed ____ and dried the dishes"

08/674610

Figure 28A - PrLF_You

If the Syntax Record
has the attribute "Infinitive"
and does not have the attribute "Subject"
or has the attribute "Verb Phrase Invert" and does not have any of the attributes "Object2," "Yes/No/Question," or "Old Subordinate Clause"
and does not meet the "There Subject Test"
and does not have the "Coordinate Constructions" attribute
and does not have any premodifiers with the node type "Auxiliary Phrase" or the attribute "Modal Verb"
and does not have any premodifiers with the lemma "let" or the node type "Adverbial Phrase,"
and does not have the node type "Abbreviated Clause," "Auxiliary Phrase," "Complement Clause," "Infinitive Clause," "Noun Relative," "Past Participle Clause," or "Relative Clause"
and does not have a parent with the node type "Past Participle Clause"
and if the head of the parent has node type "Conjunction,"
then the parent does not have a "Subject" attribute and does not have the node type "Auxiliary Phrase," "Complement Clause," "Infinitive," "Noun Relative," or "Relative Clause"
and if there is an Auxiliary Attribute on its Head
then for all its Premodifiers their Lemma must not be "neither" nor "so,"
and if it has a Do Modifier,
then it must have an Infinitive attribute and either there must not be a Modal on the First Verb attribute, or the Lemma of its First Verb must be either "dare" or "need,"
and if it has a Perfective attribute,
then its Lemma must be do,
and if it has a Verb Phrase Invert attribute,
then either there must not be a L9 attribute
or there must not be a Comma attribute and for all of its Premodifiers their node type must not be equal to "Prepositional Phrase" and for all of its Premodifiers their node type must either not be "Adverbial Phrase" or there must be a Comma attribute or the node type of their Head must be an Interjection,
and has neither "ect" nor "ect." as its Lemma,
and if its Lemma is "suffice,"
then the Lemma of its Object1 cannot be "it,"
and if its Lemma is "thank,"
then the Lemma of its Object 1 cannot be "you,"

Then

create a pronoun record for the lemma "you";

make the Subject attribute of the syntax record be a copy of the pronoun record and set the Segtype to be "NP," set the node type to be Segtype, and set the head attribute to be the pronoun record;

and set the premodifiers of the syntax record to be the value of the subject attribute plus all of the original premodifiers and set the Undersubject attribute flag.

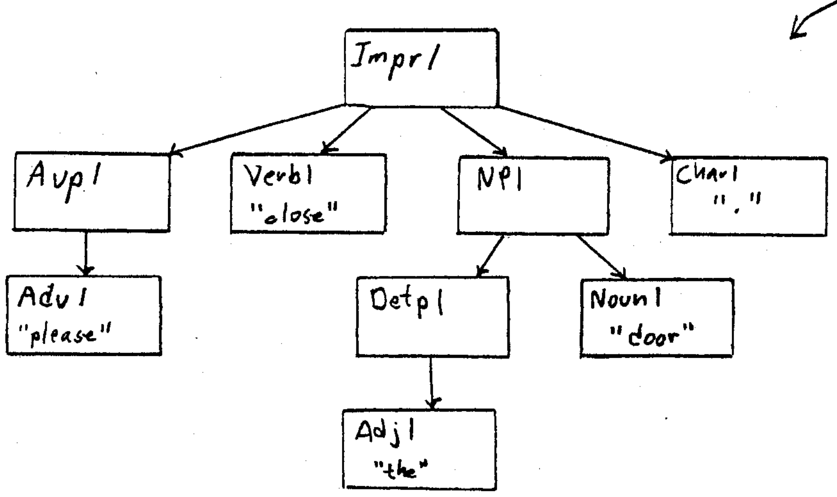
08/674610

igure 28B

sentence represented by parse tree: "Please close the door."

syntax parse tree generated by syntactic subsystem:

2801



Rule PrLF_You

2802

2803

2804

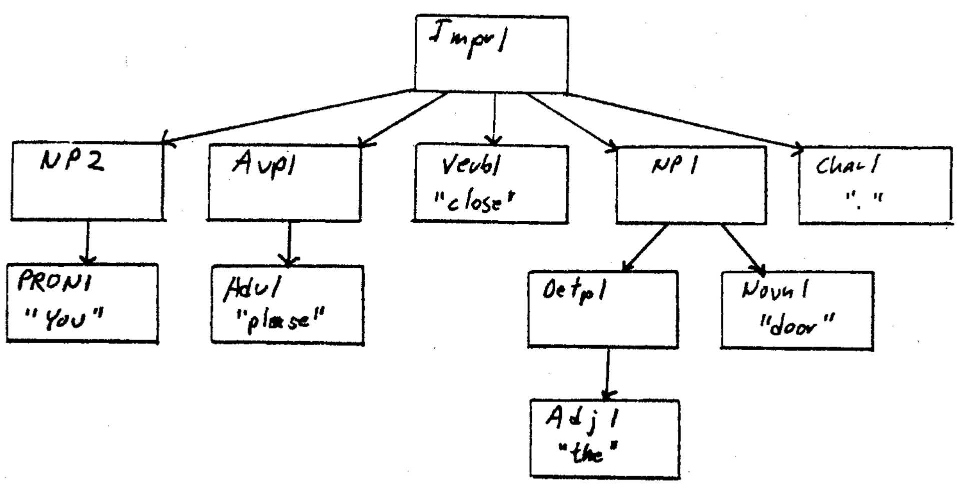


Figure 29

1. TrLF_LongDist1: locates NPs that are removed from their semantic heads and reattaches them, e.g., "Who did John say that Mary likes _(who)_?"

2. TrLF_LongDist2: performs the same kind of long-distance attachment for AJPs, INFCLs, PPs, PRPRTCLs, PTPRTCLs, SUBCLs

3. TrLF_PhrasalVerb: defines semantic objects of certain verbs when they appear hidden inside PPs: "his hat" is really the semantic object of "took off" in "He took off his hat"

4. TrLF_ControlwNP: e.g., in "Chris told Pat what to eat," "Pat" is really the subject of "eat" and "what" is its object

5. TrLF_ControlwAJP: e.g., in "I find this difficult to believe," "this" is really the object of "believe"

6. TrLF_ForInfcl: used in "for-to" constructions, e.g., in "For Mary to talk to John is easy," "Mary is really the subject of "talk"

7. TrLF_ForInfclCoords: used in "for-to" constructions that have coordinated PPs

8. TrLF_MoveProp: given our strategy for attachment, it is sometimes necessary to move clauses from a lower to a higher level so that the proper argument structure can be assigned

9. TrLF_ControlatVP: e.g., in "Farmers grow food by using salt water," "farmers" is really the subject of "use salt water"

10. TrLF_PropsAsArgs: some clauses (propositions) can be arguments, e.g., in "Has he to answer the letter?" the object of "has" is "to answer the letter"

11. TrLF_Extraposition: e.g., in "It makes me happy to meet you," the real subject of "makes" is "to meet you" -- "it" is an empty word and must drop out

12. TrLF_FillCoords: fills in missing arguments in coordinated structures

13. TrLF_RedefineSubject: e.g., in "What is John's address?" we interpret "John's address" as the logical subject even though it is not in canonical subject postition

Figure 30A - TrLF MoveProp

If the Syntax Record

has either a node type of Abbreviated Clause, Infinitive Clause, Present Participle Clause, Past Participle Clause

or if it has a Gerund attribute and an Object of a Prepositional Phrase and

if it has Premodifiers,

then the node type of all Premodifiers must be either Auxilary Phrase, Adverbial Phrase, or Prepositional Phrase,

and the node type of the Head attribute of the Parent is not "verb"

and this syntax record is the last of the post modifiers of its parent

and this syntax record is not in the coordinates attribute of its parent

and among the ancestors of the parent there is a record whose node type of the Head is "Verb" but none of those ancestors can have a Coordinates attribute (this record will later be referred to as "same ancestor")

and there should be no For To Prepositional Phrase attribute on the parent,

and if the node type equals Infinitive Clause,

then there must be either no WH attribute on PP obj of the parent or the syntax record is not equal to the Nominal Relative of the parent,

and if the node type is either Present Participle or Past Participle,

then its Parent does not have an Object of a Prepositional Phrase,

and if the node type is a Present Participle Clause,

then there must be an 'ING' Complement on the same ancestor

and if the node type is a Past Participle Clause,

then there must be a V8 (code from Longman's dictionary) attribute on the same ancestor and if there is an X1 attribute on the syntax record then there must not be an Object 1

and there is no B3 attribute on its parent,

and this syntax record must follow the head of the same ancestor or there is a passive attribute on the same ancestor

and if the Lemma of the Parent is 'certain'

then the node type of the parent must not be an Adjective Phrase

and if the Lemma of the Preposition is either "as" or "of,"

then there must be a To Noun attribute of its Parent

and if the Lemma of the same ancestor is either "be" or "become"

then either the node type of the Parent must be an Adjective Phrase

or there must be a WH attribute on the Parent

or there must be both a To Noun attribute on parent and no There Subject Test on the same ancestor

or the Lemma of the Parent must be one of the following: "delight," "horror," "joy," "pleasure," "riot," "shame," "surprise," "terror,"

Figure 30B - TrLF MoveProp

Then

the syntax record whose attributes will be changed is the same ancestor syntax record (see above);

if the Parent of the syntax record has the Subject attribute and the Parent of syntax record also has the Object attribute,

then delete the object attribute from the ancestor;

if the Parent of the syntax record has the Subject attribute and the Parent of the Syntax Record does not also have the Object 1 attribute,

then set the Subject attribute of same ancestor to be the syntax record;

if the same ancestor has

the DI (Longman code) attribute and there is an Object Complement attribute and no Indirect Object attribute and there is a To Infinitive on the syntax record and the Parent of syntax record is the Object

and there is no WH attribute on the Parent of Syntax Record

and either there is an Animate attribute on Parent of syntax record

or there is a Case attribute on Parent of Syntax Record and the Lemma of the Parent of the syntax record is not "it"

or there is a Human attribute on the Parent of Syntax Record

or there is a Proper Name attribute on Parent of syntax record,

then make the Indirect Object Attribute on same ancestor equal to that of the Parent of syntax record;

if there is an To Infinitive attribute on the syntax record and no Passive attribute on same ancestor,

then make the Predicate Complement attribute equal to the syntax record;

if the Parent of syntax record is in the Propositions attribute of same ancestor,

then take that Propositions list and replace the Parent of the syntax record with the syntax record itself in the propositions list;

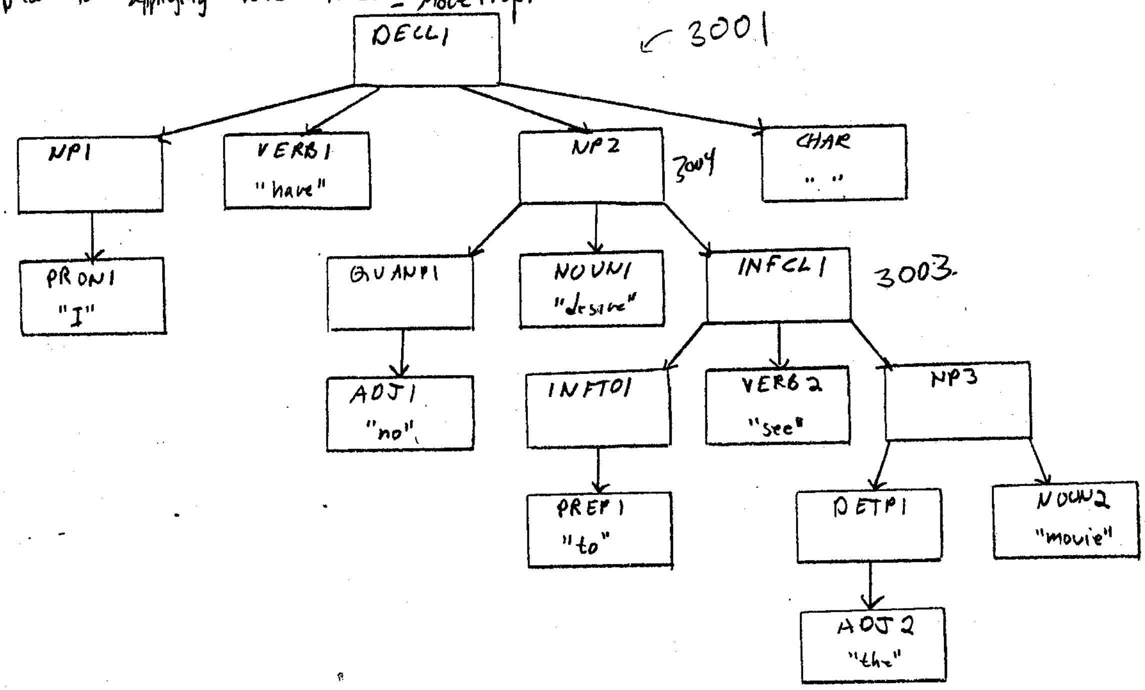
delete the Infinitive attribute of the Parent of the syntax record;

delete the Alternatives attribute on the syntax record;

reattach the syntax record to the same ancestor.

Figure 30.C

sentence represented by parse tree: "I have no desire to see the movie."
syntax parse tree prior to applying rule TrLF_MoveProp:



Rule TrLF_MoveProp:

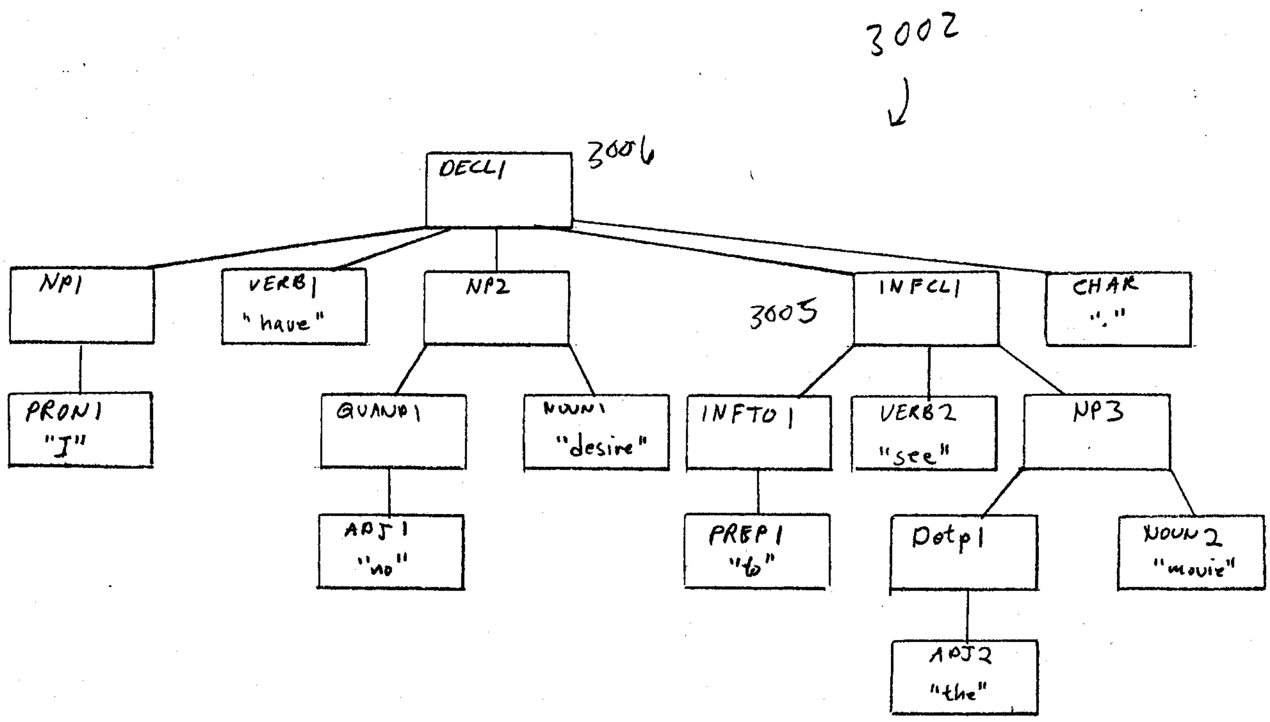


Figure 31

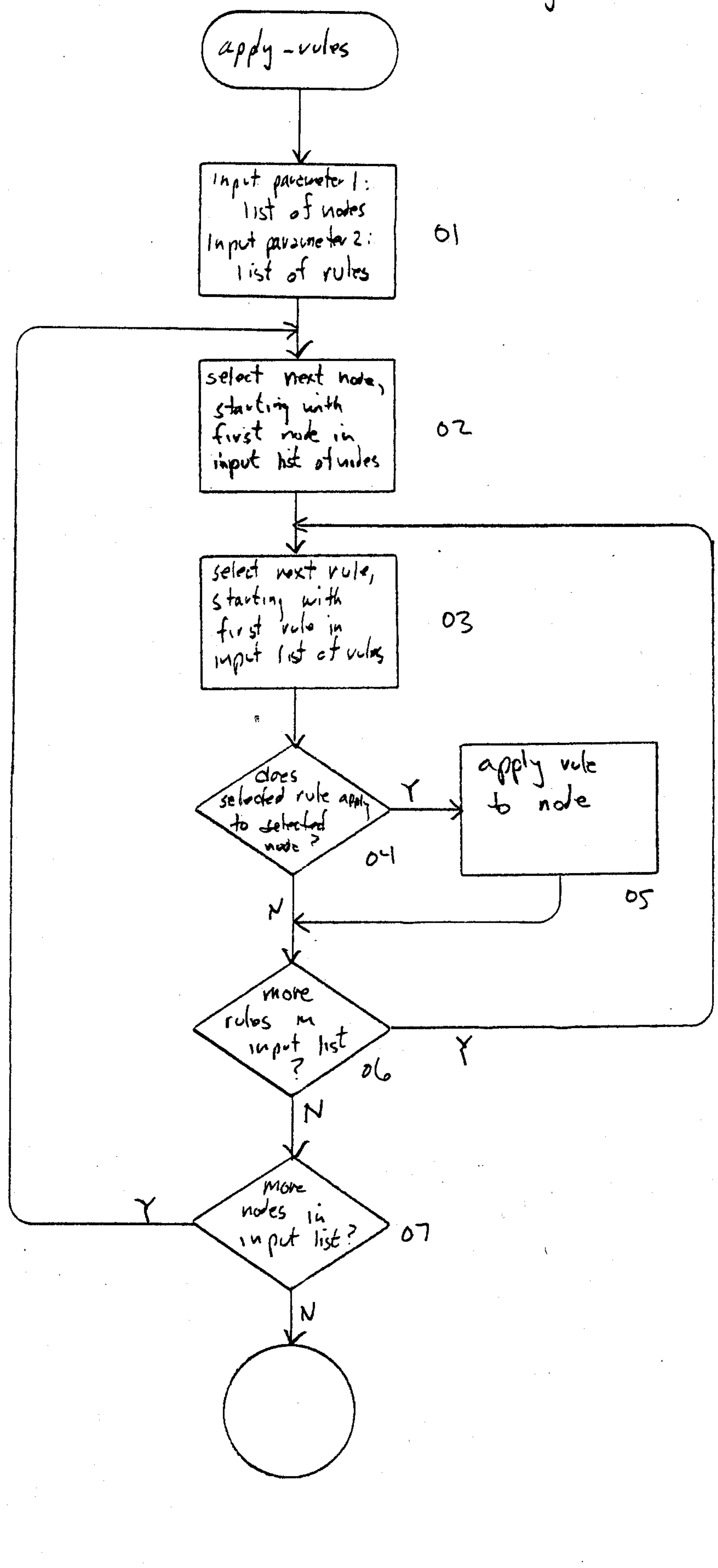

apply-rules
Input parameter 1: list of nodes
Input parameter 2: list of rules
01
select next node, starting with first node in input list of nodes
02
select next rule, starting with first rule in input list of rules
03
does selected rule apply to selected node?
04
Y
apply rule to node
05
N
more rules in input list?
06
Y
N
more nodes in input list?
07
Y
N

Figure 32

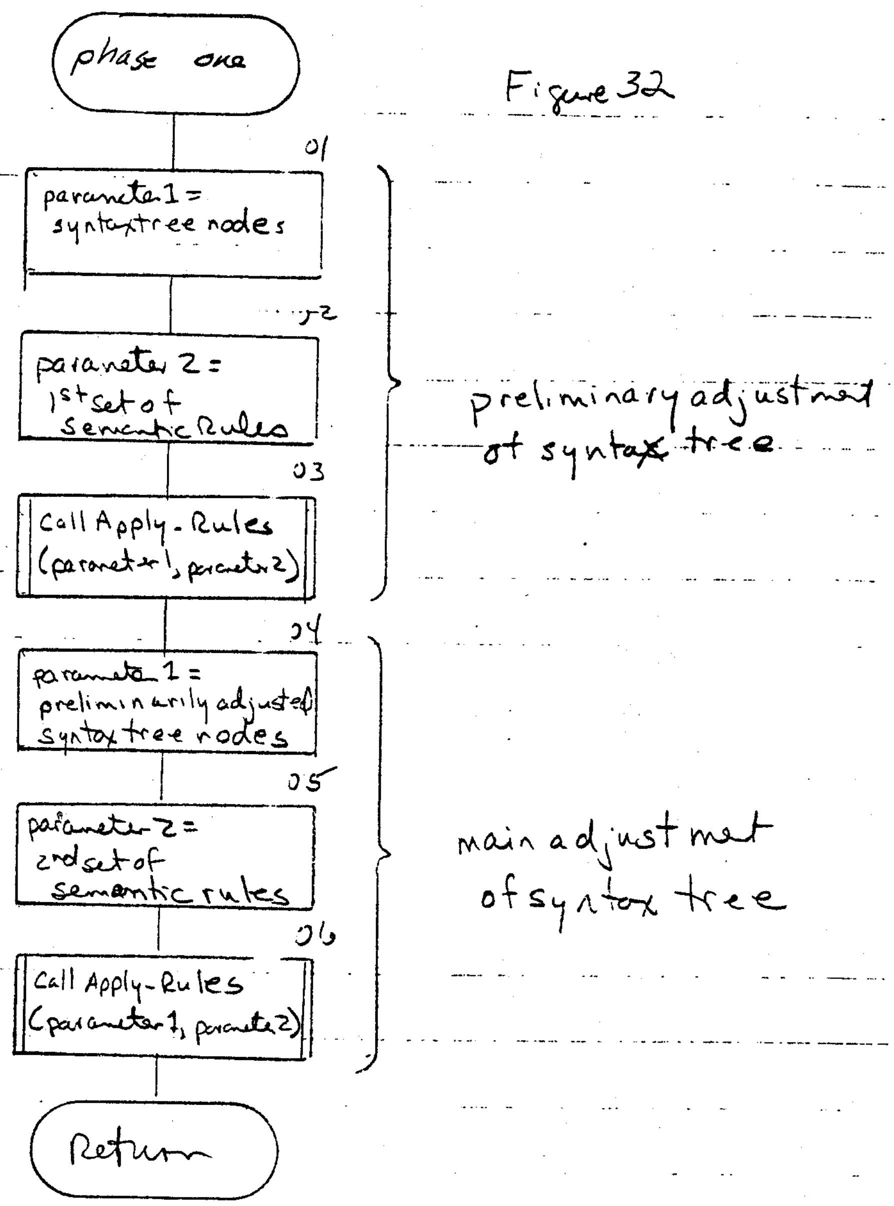

phase one
01
parameter 1 = syntaxtree nodes
02
parameter 2 = 1st set of semantic Rules
03
Call Apply-Rules (parameter 1, parameter 2)
preliminary adjustment of syntax tree
04
parameter 1 = preliminarily adjusted syntax tree nodes
05
parameter 2 = 2nd set of semantic rules
06
Call Apply-Rules (parameter 1, parameter 2)
main adjustment of syntax tree
Return

Figure 33

1. SynToSem1: creates semantic nodes and a basic semantic graph in es

2. SynToSem2: creates the top-level semantic node and graph for fitted parses

3. SynToSem3: creates semantic nodes for a special subclass of elements in fitted parses

Figure 34 A - Rule SynToSem1

If

the Syntax Record
has a Head and
there is no Subordinate Conjunction and
there is no Correlative and
there is no "It subject" and
there is no "There subject" and
there is no Ancestor of the Head for which it is true that that node
is the Emphatic of its Parent and is not a fraction and the head node
is not a verb and
if the segment is the Relative Pronoun of its Parent,
then there must not be a Nominal Relative on the Object of its Parent
and for all of its Parents last records there must not be a VPDone attribute and
if the lemma equals 'that'
then there must not be an Extra Position on the Parent of the Parent and
the node type is not "Auxiliary Phrase," "To Infinitive," "Determiner Phrase,"
or "Tag" or
there is a Possessive attribute or
there is an EVR attribute or
the Lemma equals "other" or
there are Coordinates and for all of those coordinates there is either a
Possessive attribute or an EVR attribute or the lemma is "other" and
if the node type is "Adverb Phrase"
then if the node type of Parent equals Prepositional Phrase
then the segment must not be the first of the Premodifiers of its Parent
and
either the Lemma must not be equal to 'well' or there must not be any Degree
attribute or there must not be any Weak Obligation on the Parent and
If the node type of the Head is a Conjunction or a Preposition,
then the segment node must not be a Conjunction of the Parent and the
segment node must not be a Preposition of the Parent and
If the node type is a Conjunctive Phrase
then there must not be any Coordinates of the Parent or there must not be a
Coordinate Conjunction attribute and
If the node type is a Quantifier Phrase,
then the Lemma of the Head must not be "no" and
If the word could have been an Interjection
then the node type must not be an Adverb Phrase or
there must be Premodifiers or
there must be no comma or
the segment must be the Post Adverbial of the Parent or
the number of Post Modifiers must be greater than one and
If there is an Intensifier attribute
then either the node type of Head of Parent is a "verb" or
the node type of Parent equals "fitted" or
there is an Adverbial Phrase attribute or
there is a WH marker and a Nominal Relative on the Parent and
If there is a Preposition attribute,
then there must be an Object of the Prepositional Phrase or
there is a Particle attribute on the Parent or
the word also could have been an Adverb and

Figure 34 B - Rule SynToSem1

If the Lemma is "also", "so," or "too,"
then there must not be a VPDone attribute on the Parent and
If the Lemma is "as" or "than"
then there must not be a Comparative on the Parent and
If the Lemma equals "for"
then there must not be a "for to" Preposition on the Parent and
If the Lemma equals "it"
then if there is a Topic Clause on the Parent
then the segment must be equal to the Subject of the Parent or
the segment must be equal to the Object of the Parent and
If the Lemma equals "it"
then the segment must not be in the Premodifiers of the Parent or
If there is an Extra Position on the Predicate Adjective of the Parent
then there must not be a Right Shift attribute on the Parent and
if there is a WH Question attribute on the Parent
then there is no "To Infinitive" attribute on the Predicate Compliment of the Parent and it's not the case that for any of the Post Modifiers of the Parent that there is a "For to" prepositional phrase on the first of the Premodifiers and
If the Lemma equals "let"
then the node type is not equal to "Adverb Phrase" and
If the Lemma equals "not"
then there must be a Coordinate Conjunction on the Parent and
If the Lemma equals "there"
then there must not be any Skipover attribute and
either there must not be any "Yes No" question on the parent or
there must not be a Copulative on the Parent or
there must be a T1 attribute on the Parent or
the first token integer must be greater than the first token integer of the Subject of the Parent and
If the Lemma is "whether" or "whether or not"
then the node type of the Nominative Relative must not be an "Infinitive Clause" and
the Lemma must not be "etc," "etc.," "the," "hm," "mm," "uh," or "um"

THEN

(If syntax node was kept, then create a corresponding semantic node.)
If the node type of the syntax node is a Noun Phrase and
there are Bases on the syntax node and
there is a Subject or an Object on the syntax node,
then make the Predicate equal to the Lemma of the first Basis of the syntax node
Else if there is a Proper Noun attribute on the syntax node and
if there is a dictionary entry for that word,
then make the Predicate equal to that dictionary entry
Else set the Predicate equal to the Lemma of the syntax node
If the word could have been a Verb and has a Present Participle attribute and
if for any of the Premodifiers of the syntax node there is a Possessive or
if the Lemma of the Preposition of the first of the Postmodifiers of the syntax node is "by," "for," "of," or "to"

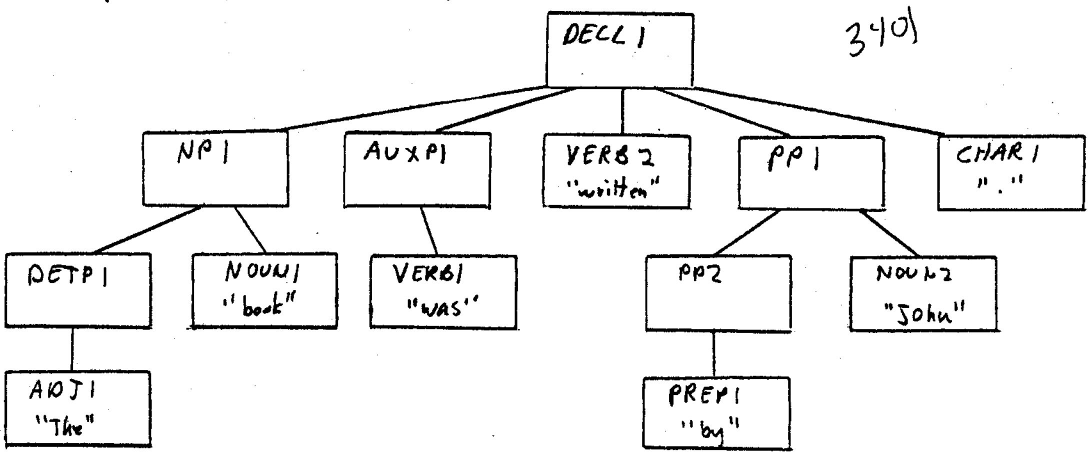
Figure 34 C - Rule SynToSem1

then make the Predicate equal to the Lemma of the Verb entry of the Part of Speech
Record
Copy the appropriate fields from syntax node to the semantic node.
Go through each of the Premodifiers of syntax record and examine each Premodifier
For each record of Premodifiers of the syntax record
if there is a semantic node on the record and
if the semantic node of the record is not in the temporary modifiers attributes of this semantic record and there is no Skipover attribute on the record and the record is not equal to the Preposition of the Parent of the record and the record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record, or Coordinate Subordinate Clauses
then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record
For each record of the Postmodifiers of the syntax record
if there is a semantic node on record and
if the semantic node of record is not in the Temporary Modifiers attributes of this semantic record and there is no Skipover attribute on record and record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record or Coordinate Subordinate Clauses
then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record
If there are Coordinates of the syntax record and no Coordinates of the Prepositional Phrase on that syntax record and no Coordinate Subordinate Clauses
then
for each of the Coordinates of syntax record
if there is a Semantic node on record,
then add that Semantic node to Coordinates attribute on this new Semantic record.

Figure 34D

Sentence represented by syntax parse tree: "The book was written by John."
syntax tree prior to application of rule SynToSem1;

3401



Rule SynToSem1:

3402

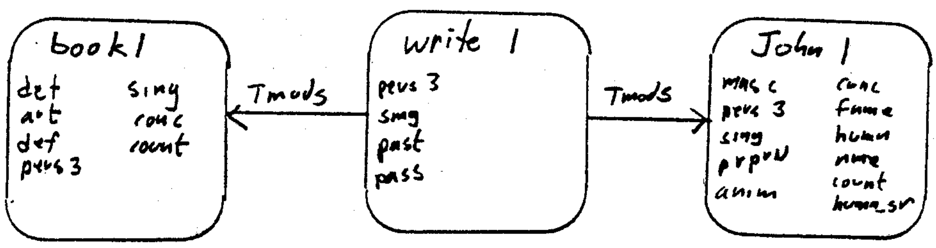


Figure 35

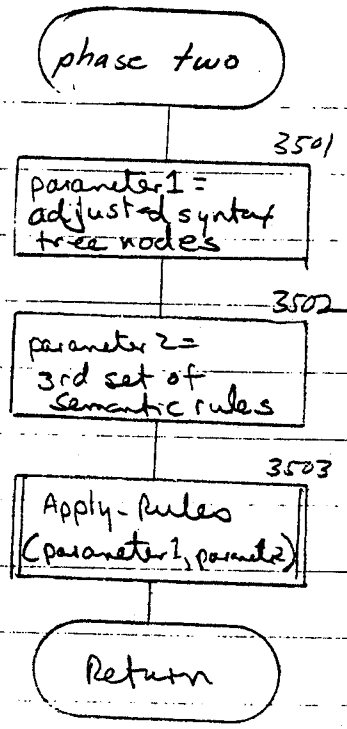

phase two
3501
parameter1 = adjusted syntax tree nodes
3502
parameter 2 = 3rd set of semantic rules
3503
Apply-Rules (parameter1, parameter2)
Return

Figure 36

1. LF_Dsub1: creates the Dsub (deep subject) label for subjects of clauses in the active voice
2. LF_Dsub2: for passive-voice clauses, if there is a "by"-PP, identifies this PP as the Dsub of the action
3. LF_Dobj1: creates the Dobj (deep object) label for, e.g., direct objects of clauses in the active voice
4. LF_Dobj2: for passive clauses, identifies the syntactic subject as the deep object of the action
5. LF_Dobj3: for clauses like "The door opened," identifies "the door" as the logical object of the action
6. LF_Dobj4: for constructions like "the nomination of the candidate," identifies "the candidate" as the logical object of an action of nominating
7. LF_Dind1: creates the Dind (deep indirect object) label for, e.g., "Mary" in "John gave Mary the book"
8. LF_Dind2: identifies the deep indirect object ("Mary") in paraphrases like "John gave the book to Mary"

Figure 37

9. LF_Dind3: chooses the right deep indirect object in trickier constructions like "The book was given her"; "She was given the book"

10. LF_Dnom: creates the Dnom (deep nominative) label for predicate nominatives, e.g., "our friends" in "They are our friends"

11. LF_Dcmp1: identifies the complement ("president"; "italic") in, e.g., "elect Tom president"; "make the word italic"

12. LF_Dcmp2: identifies the complement in tricker constructions, e.g., in "He gave Tom a place to call his own," "his own" is the Dcmp of "call"

13. LF_Dadj: creates the Dadj label for predicate adjectives, e.g, "blue" in "The sky is blue"

14. LF_CausBy: creates a causative relation where appropriate, e.g., "why" in "Why did you say that?"

15. LF_LocAt: creates a locative relation where appropriate, e.g., "where" in "Where did you find that?"

16. LF_TmeAt: creates a temporal relation where appropriate, e.g., "what day" in "What day did you read that?"

17. LF_Manr: creates a manner relation where appropriate, e.g., "how" in "How did you do that?"

Figure 38

18. LF_Ptcl: creates a Ptcl node to refer to particles in phrasal verb constructions

19. LF_PrpCnjs: creates temporary relations for PPs and subordinate clauses by naming these elations with the word that is the preposition or conjunction

20. LF_PrpCoord: handles cases of coordinated PPs or subordinate clauses

21. LF_Props: lists remaining clausal adjuncts for any given node

22. LF_Ops: identifies logical operators in noun phrases, e.g., "all" in "all my children"

23. LF_Nadj: lists remaining adjectives that premodify nouns

24. LF_Mods: lists remaining non-clausal modifiers for any given node

Figure 39A - Rule LF_Dobj2

If the Semantic Record

doesn't already have a Deep Object,

and has a Passive attribute,

and has a Subject on its syntactic record (SynNode), and this Subject (which is a syntactic record) has a SemNode attribute (i.e., it has a corresponding semantic record)

and there are no Coordinates

and if there is a Predicate Complement attribute on its synactic record, then the node type is not "COMPCL" (i.e., it is not a complement clause, as in: "some people were convinced <u>that he had written a book</u>"

and if the SynNode record has either a D5, D6, ObjC, or Psych feature[2]

then either the Object of the SynNode is not a noun phrase,

or the SynNode has an X1[3] feature (as in: He was named Arles")

or the Object of the SynNode has an Animate feature

or there is a Case feature on the Object of the SynNode and its Lemma is not "it"

Then,

give the Semantic record a Dobj attribute with, as its value, the semantic record corresponding to the Subject on the syntactic record

and, remove what is now the value of Dobj attribute from the list of Tmods

---

[2]D5, and D6 are features from Longman's Dictionary of Contemporary English; ObjC is verb subcategory for verbs which show object control (e.g., I want Harry to wash the car) and Psych is a verb subcategory for verbs like "scare" "excite".

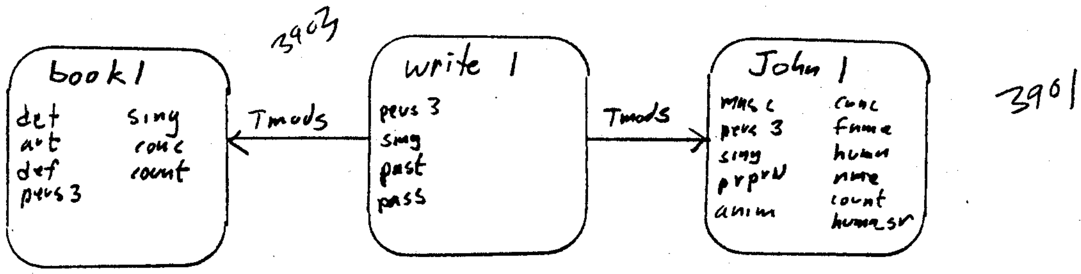
[3] X1 is a feature from Longman's Dictionary of Contemporary English.

08/674610

Figure 39B

sentence represented by the logical form: "The book was written by John."

logical form prior to application of rule LF_Dobj 2:



LF_Dobj 2:

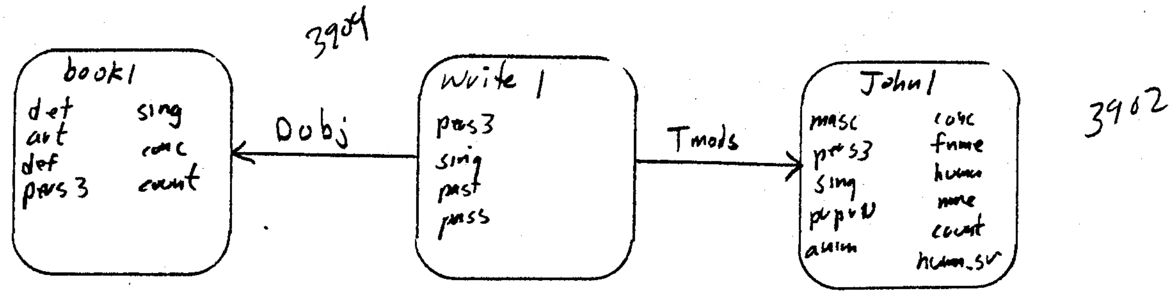


Figure 40

1. PsLF_RelPro: identifies proper referents for relative pronouns, e.g., "who" refers to "the man" in "the man who came to dinner"

2. PsLF_ReciprocalAnaphora: handles reciprocal pronouns like "each other" and "one another"

3. PsLF_ReflexiveAnaphora: handles reflexive pronouns like "myself, yourself, him/herself," etc.

4. PsLF_PronAnaphora: identifies possible NP referents for most pronouns

5. PsLF_ProtoAnaphora: handles special cases of pronouns which can agree with just about any NP

6. PsLF_NumberEllipsis: handles reference for number words, e.g., "A bird in the hand is worth two (birds) in the bush"

7. PsLF_FillInHead: adds "DUMMY" as a head word in special cases of unclear referents

8. PsLF_NumberCritique: takes note of pronouns that disagree in number with their referents

9. PsLF_FillDsub: fills in "x" as a placeholder for the deep subject in cases where that is missing, e.g., in passives like "The door was opened"

10. PsLF_UnifyProns: if two pronoun nodes refer to the same referent, this rule unifies them

11. PsLF_UnifyCopies1: unifies some nodes that should be identical

12. PsLF_UnifyCopies2: unifies other nodes that should be identical

13. PsLF_RaiseModality: deletes some verbs when they serve only an aspectual purpose, e.g., in "We used to go there," "used to" is deleted from the graph

14. PsLF_RaisePcs: makes fitted parses easier to read

Figure 41A - Rule PsLF_PronAnaphora

If the Semantic Record

has a Pers3 attribute, i.e., it is not either first (e.g., I or we) or second person (e.g., you)

and the node type of the head of its syntactic record is either "PRON" (pronoun) or the node type of the head of its syntactic record is "ADJ" (adjective) and it has a possessive attribute

and is not Reflexive

and none of the premodifiers of the Parent of its syntactic node has the Lemma "own"

and the Pred of this semantic record is not "each other" or "one another"

and does not have NonRef attribute (NonRef is an attribute set on words that cannot have a reference, such as true numbers, as in: One plus one is two.

and does not have a Negation attribute

and if it has an Indefinite attribute, then there must also be a Definite attribute

and is not a Wh- word (it does not have a Wh attribute)

and is not a Relative

and is not a Distal (Distl) or a Proxal (Proxl) determiner (e.g., "this" "that")

Then

add a FindRef attribute to the semantic record

for each of the records in the list of possible referents; [1]

if

the possible referent has a corresponding semantic record

and the possible referent is not the same as this record (i.e., the antecedent of a noun phrase can not be the noun phrase itself)

and if the head of both the possible referent and of this record's SynNode are pronouns (i.e., have the node type "PRON" as their head), then the possible referent must precede this record (no forwards reference to a pronoun; an example of forwards (cataphoric) reference is: with his hat on, the teacher left the room, where "his" refers forwards to "teacher"

and if the possible referent is the ancestor of the syntactic record of this record, then that ancestor must have a Prp attribute (i.e., must have a postmodifying Prepositional phrase), and its preposition must be either "in", "to", "for", or "by"

and there is no Time or Space feature on the possible referent

and this record and the possible referent agree in number

and this record and the possible referent agree in gender

and if the Lemma of the SynNode is "they" and the possible referent can be a Mass noun (i.e., the possible referent has a Mass feature),

then the possible referent must also be a Count noun (i.e., it must also have a Count feature).

and if the Lemma of the SynNode is "they" and the possible referent has a Sing feature (can be Singular), and the possible referent does not have a Plur feature (i.e., it cannot be Plural),

then the possible referent is either a Count noun, or the possible referent is a Coordinated noun phrase, or it has a Universal feature, or the possible referent is indefinite and has no possessive, or the possible referent has a Proxal feature,

---

[1] this list is created in a PrLF rule, so, after syntactic processing but before most logical form processing (it is a list of syntactic records). This is a list of all the words in the sentence which can be referred to, i.e. most of the nouns and pronouns in the sentence

Figure 41B - Rule PsLF_PronAnaphora

and if there is an ancestor of the possible referent that has a Coords attribute (i.e., has coordinate constituents) (but before there is an ancestor with a Subject attribute) then this ancestor is the same as the ancestor of this record that has a Coords attribute (but before there is an ancestor with a Subject attribute)

then if this record is a possessive (e.g., "his" in "John saw his son")

add the possible referent to the list of possible referents (the value of the Refs attribute)

if:

the possible referent is a genitive

and node type of the head of the possible referent is not a Noun

and the possible referent precedes this record (i.e., the semantic record being processed in this rule

or if:

the possible referent is not the first of this record's Parents

and the first of the Parents of the possible referent is not the first of this record's Parents

and if the possible referent follows this record and if any of the possible referent's ancestors have Coordinate constituents, then there should be no ancestor of this record for which the Parent has Coordinate constituents and for which the Parent is the same as the ancestor of the possible referent that has Coordinate constituents (but before there is ancestor whose node type is "NP")

or else if the node type of Parent of this record's syntactic record is "TAG" (i.e., if the pronoun is in a tag question)

add the possible referent to the list of possible referents (the value of the Refs attribute)

if:

the possible referent is the Subject of the Parent of the Parent of this record (e.g., "they" refers to "someone" in : Someone painted in here, didn't they?)

or else:

if

this record is a prepositional phrase

and this record precedes the Subject of this record's Parent

and the possible referent is the Subject of this record's Parent

then add the possible referent to the list of possible referents (the value of the Refs attribute);

else if

this record is not possessive

and this record precedes the possible referent

and node type of the head of the possible referent is "NOUN" and is not a Dummy noun (i.e., one that cannot be a possible referent)

and if this record is not one of possible referent's ancestors

and if it is not the case that there is an ancestor of this record that has Coordinate constituents and the Lemma of that ancestor is "but" and that ancestor is also an ancestor of this record which has Coordinate constituents

then add the possible referent to the list of possible referents (the value of the Refs attribute)

Figure 41C - Rule PsLF_PronAnaphora

else if
the possible referent is a Prepositional Phrase
and the Parent of the possible referent is not the Parent of this record's syntactic record
and if the Parent of the possible referent is an Adjective Phrase, then the Parent of the possible referent precedes this record
then add the possible referent to the list of possible referents (the value of the Refs attribute)

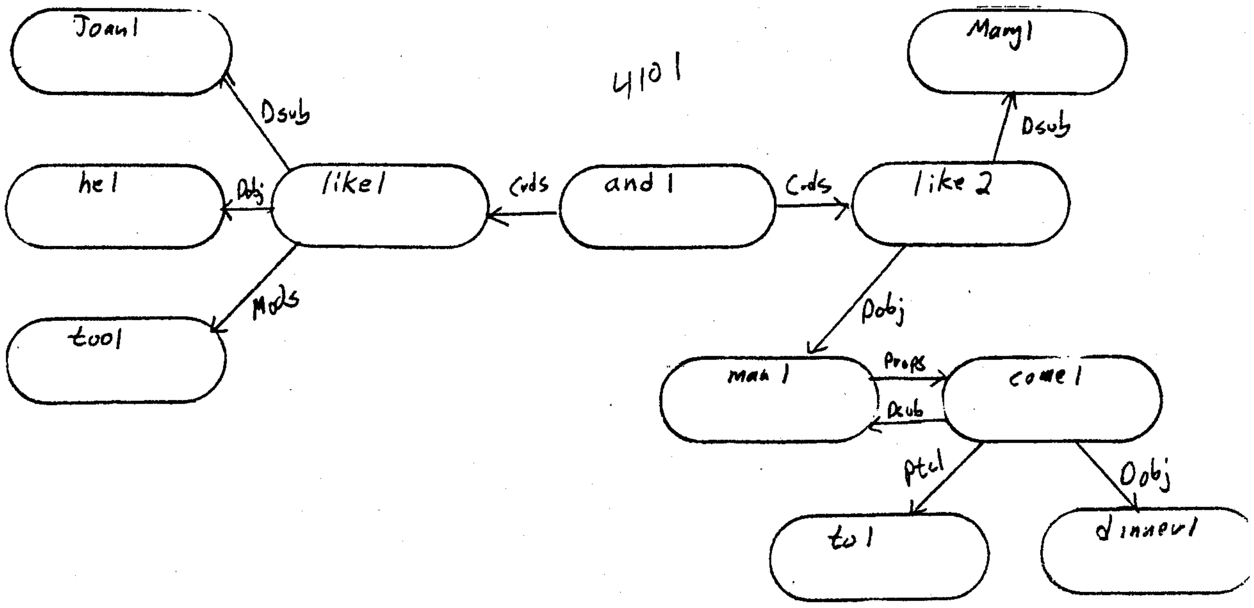
else if
there is no ancestor of the possible referent for which the Lemma is "be" (but before there is an ancestor with a Subject) that is the same as the ancestor of this record for which the Lemma is "be" (but before there is an ancestor with a Subject)
and none of the Parents on the semantic record of the possible referent is the same as the possible referent
and if this record precedes the possible referent, then the Head of the possible referent is not either a Noun or an Adjective
then add the possible referent to the list of possible referents (the value of the Refs attribute)

if the possible referent was added to the list of possible referents (the value of the Refs attribute)
then add of RefOf attribute to the possible referent and add this record to that list (provide cross pointers: this record gets a Ref attribute pointing to possible referents, and the possible referents each get a RefOf attribute, pointing back to this record.

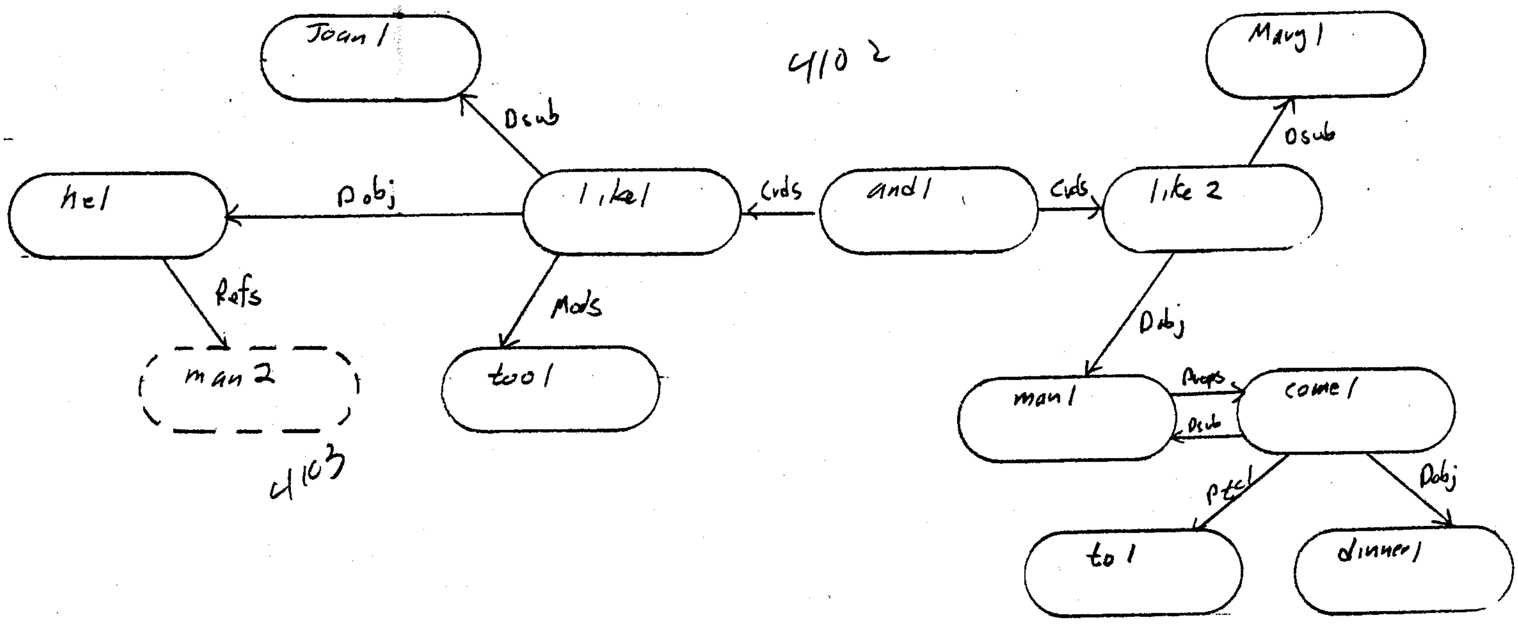
Figure 41D

sentence represented by logical form: "Mary likes the man who came to dinner, and Joan likes him too."

logical form prior to application of rule PsLF_ Pron Anaphora:



Rule PsLF_ Pron Anaphora:



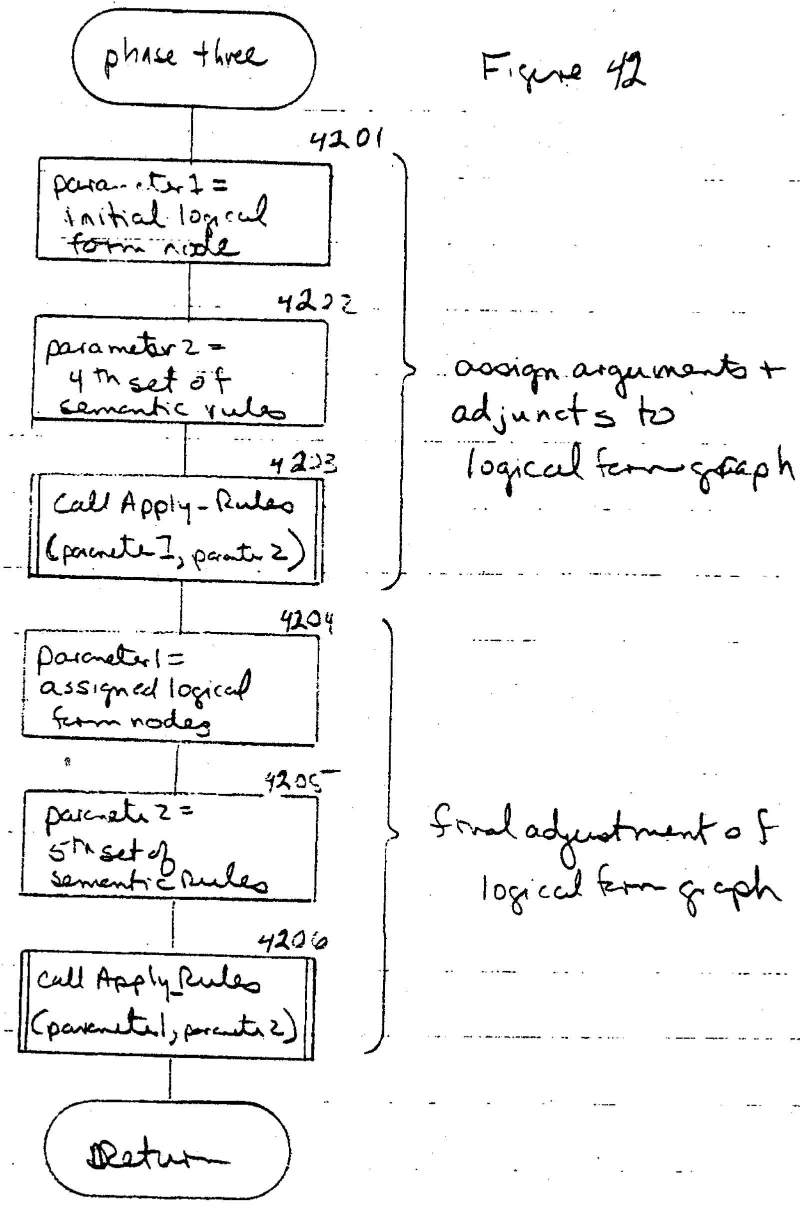

phase three
4201
parameter 1 = initial logical form node
4202
parameter 2 = 4th set of semantic rules
4203
Call Apply-Rules (parameter 1, parameter 2)
assign arguments + adjuncts to logical form graph
4204
parameter 1 = assigned logical form nodes
4205
parameter 2 = 5th set of semantic rules
4206
call Apply Rules (parameter 1, parameter 2)
final adjustment of logical form graph
Return

Figure 42

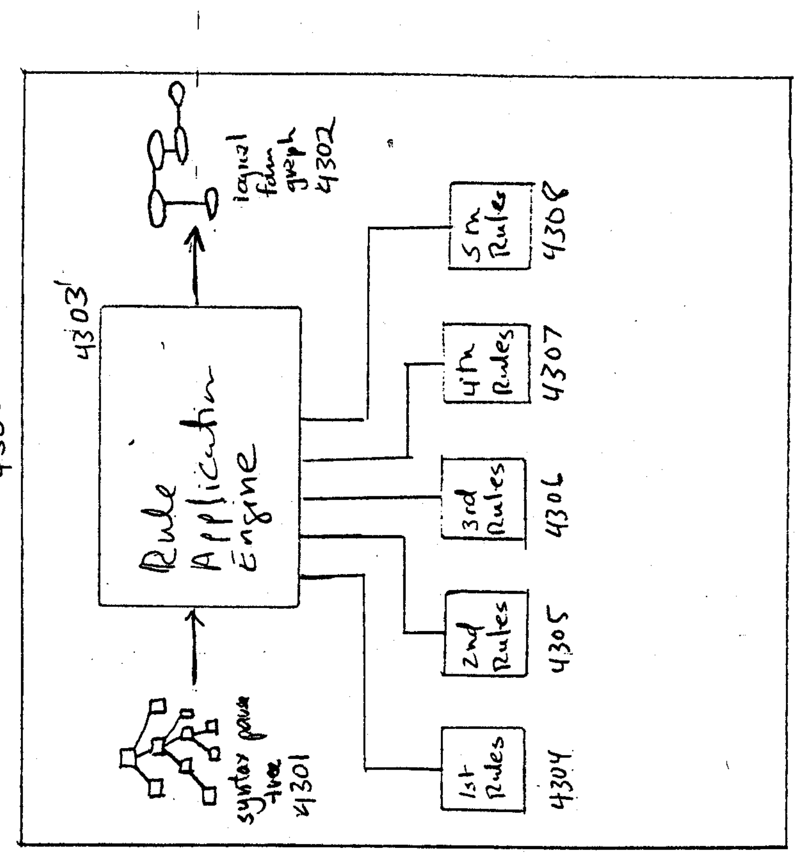
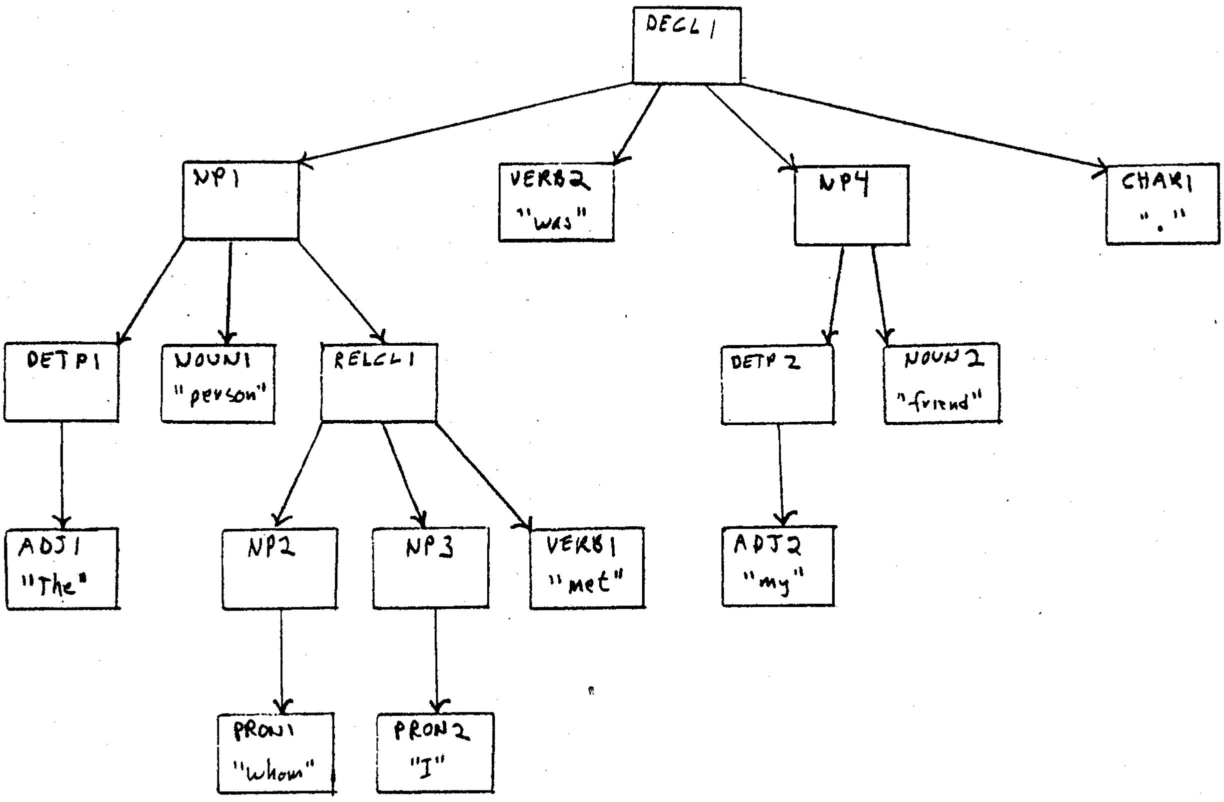

4300
4303'
Rule Application Engine
syntax parse tree 4301
logical form graph 4302
1st Rules
2nd Rules
3rd Rules
4th Rules
5th Rules
4304
4305
4306
4307
4308

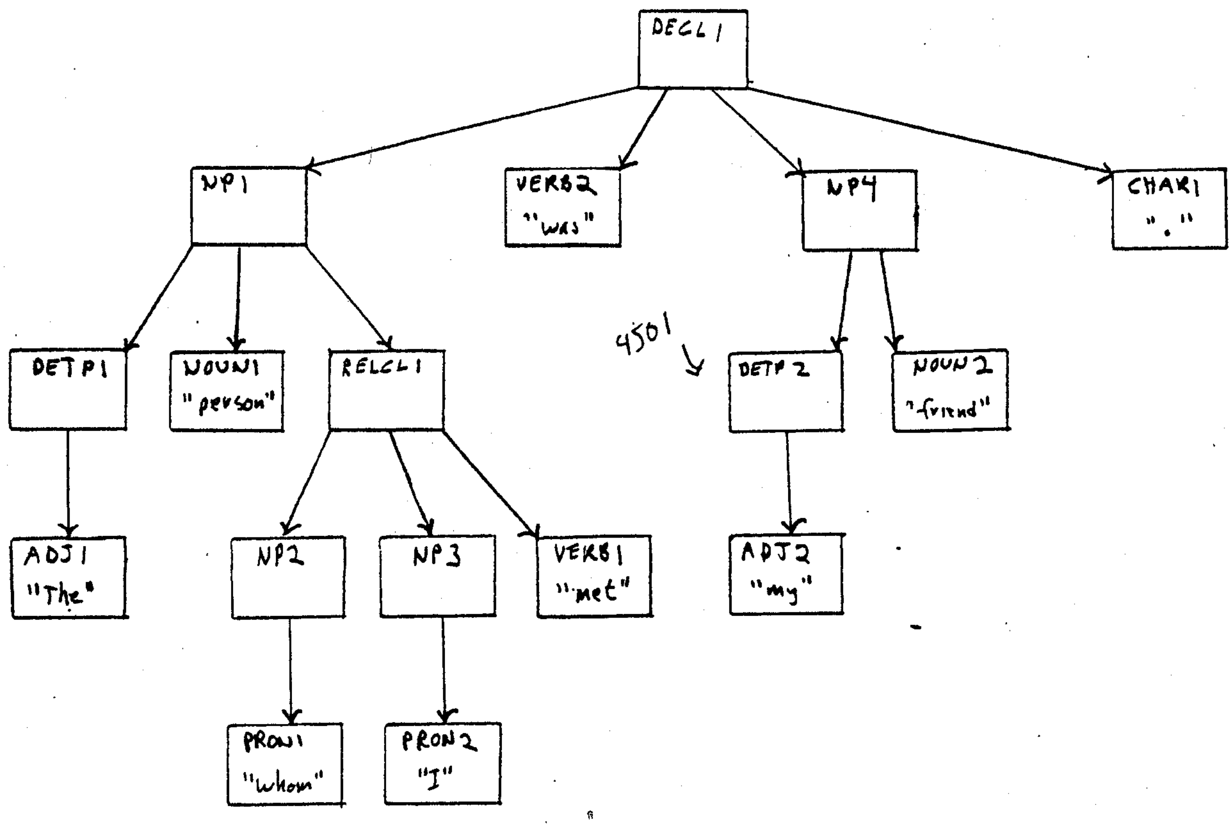
Figure 43

Figure 44


DECL1
NP1
VERB2
"was"
NP4
CHAR1
"."
DETP1
NOUN1
"person"
RELCL1
DETP2
NOUN2
"friend"
ADJ1
"The"
NP2
NP3
VERB1
"met"
ADJ2
"my"
PRON1
"whom"
PRON2
"I"

08/674610

ure 45



Rule: SynToSem1 produces logical form graph node from DETP2 ("my")

⇓

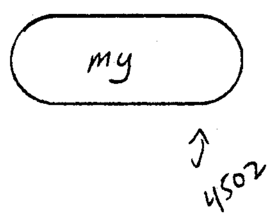
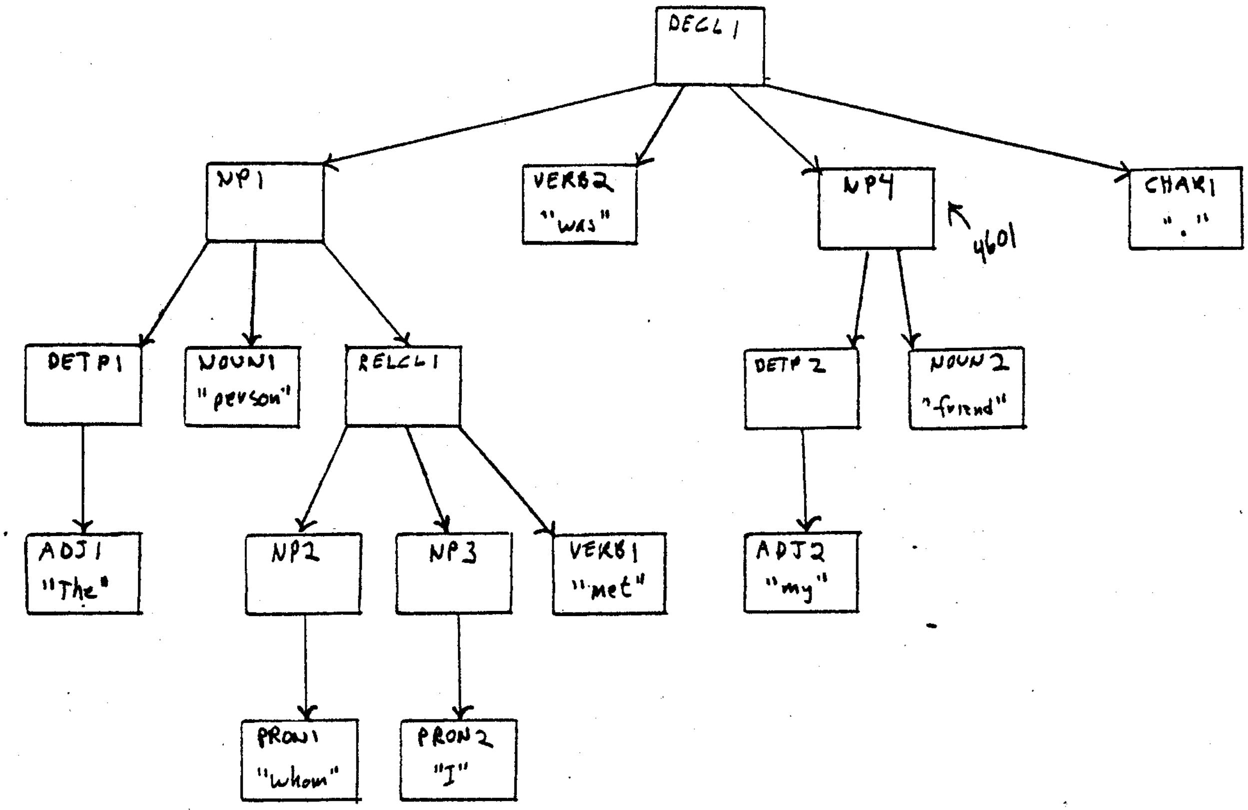


Figure 46



Rule: SynToSem1 produces logical form graph node "friend" from NP4 ("my friend")

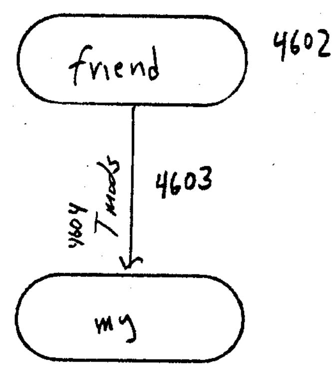
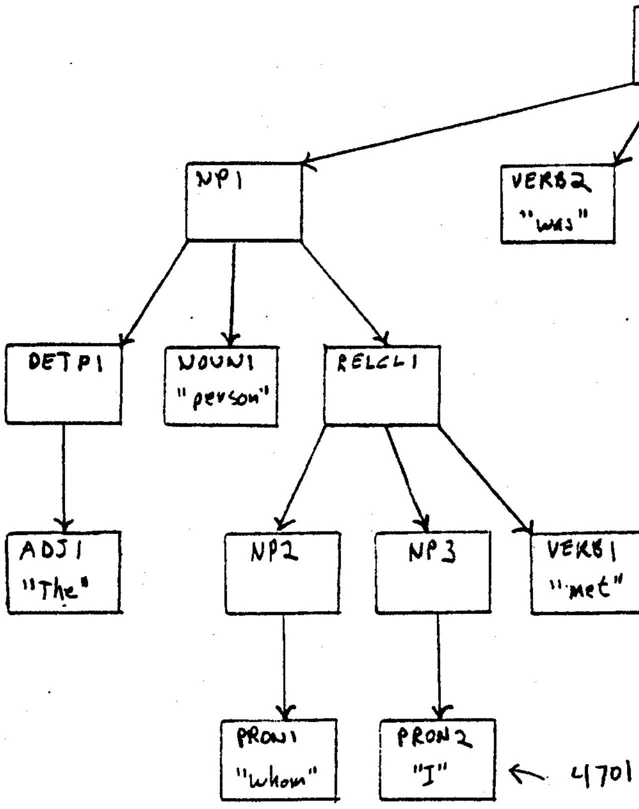


Figure 47



Rule: SynToSem1 produces logical form graph node "I" from NP3("I")

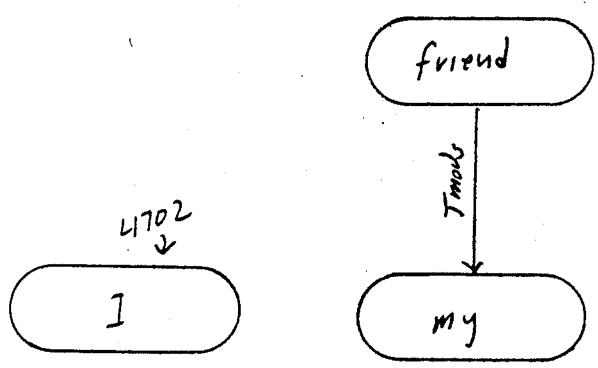
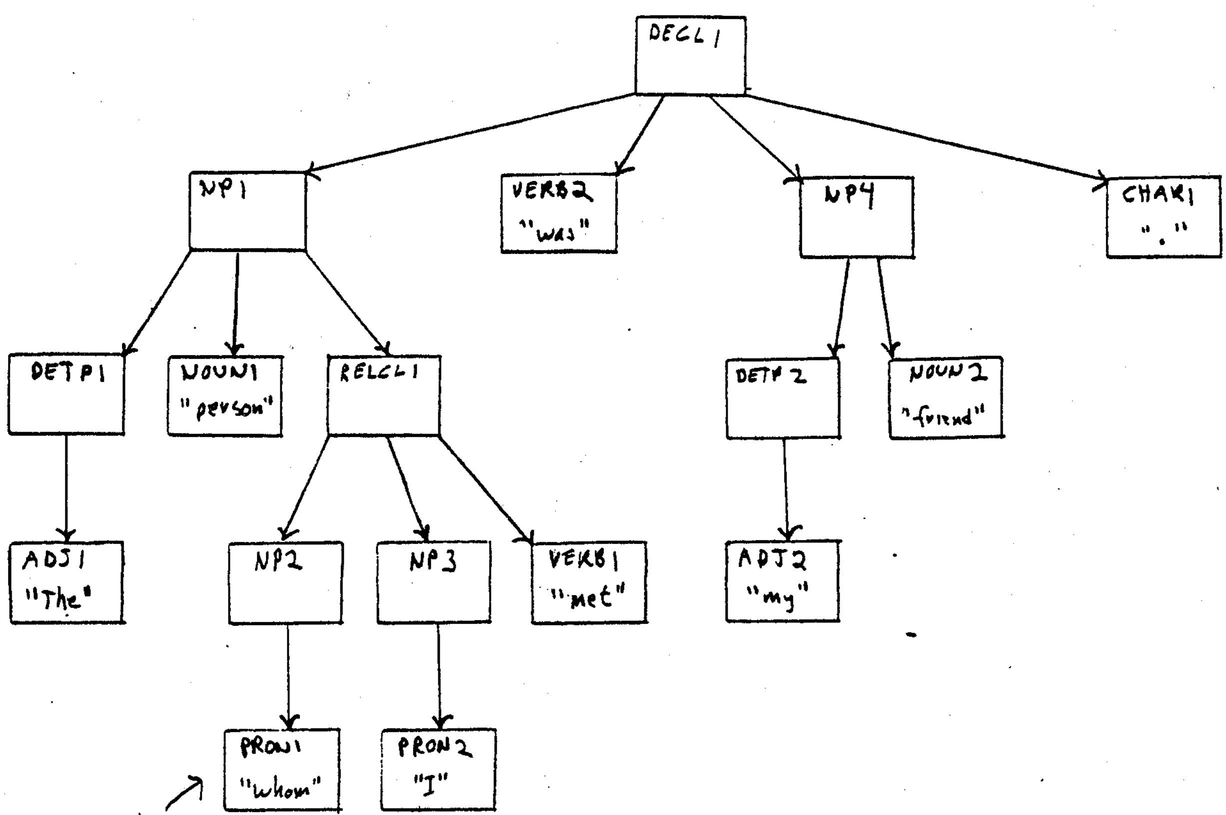
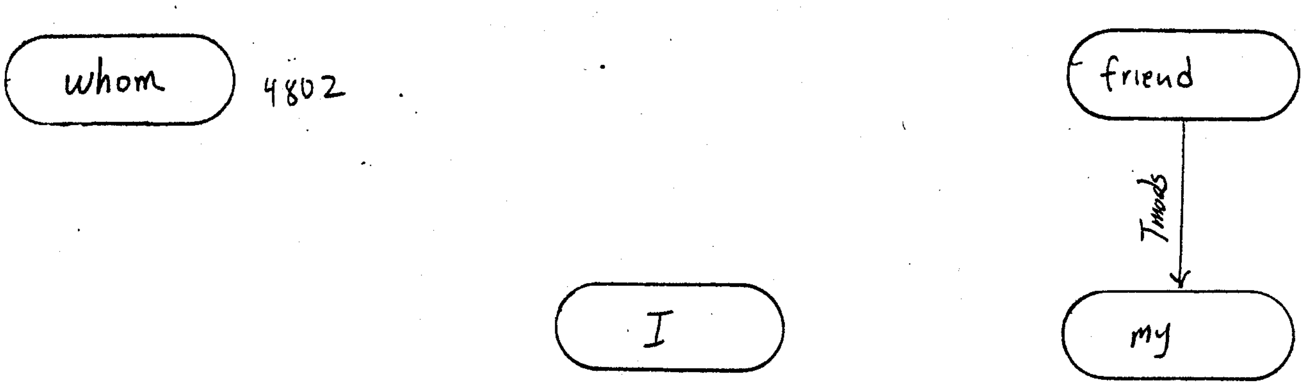


Figure 48



4801

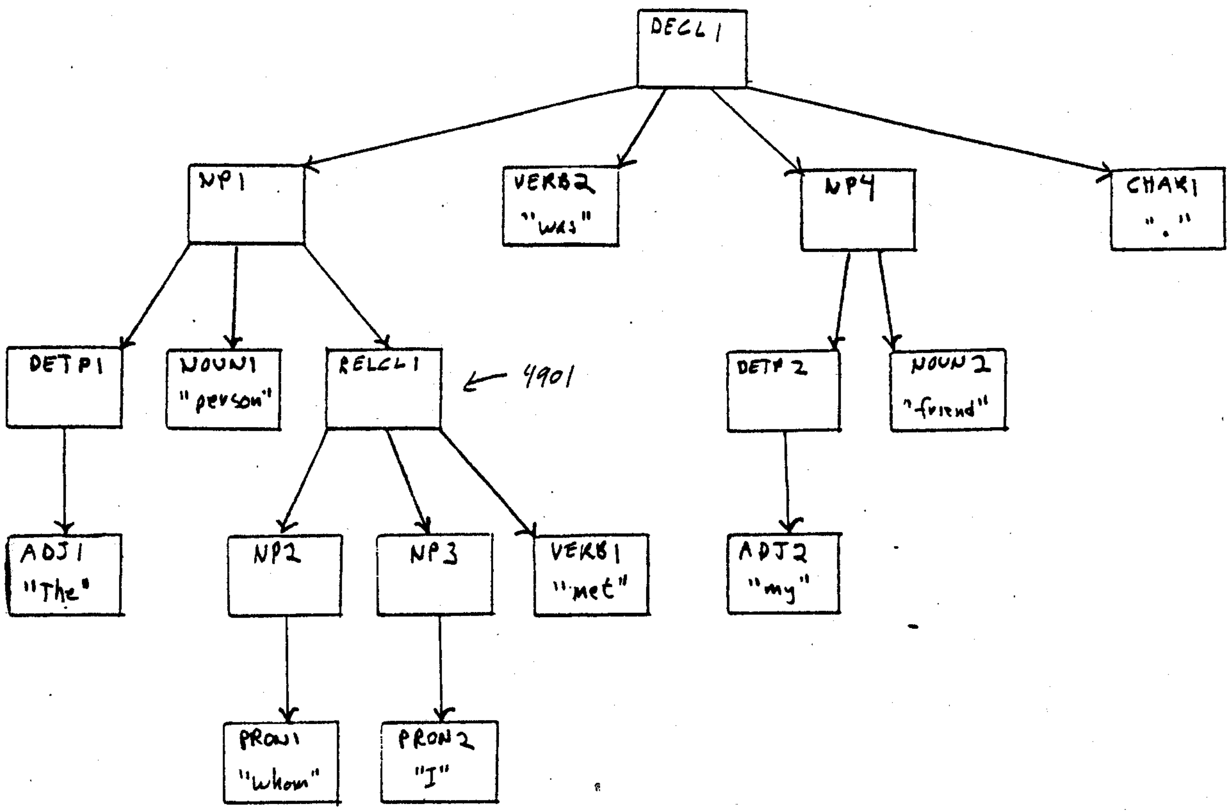
Rule: SyntoSeml produces logical form graph node "whom" from NP2 ("whom")



4802

Figure 49

08/674610



Rule: SyntoSem1 produces logical form graph node "meet" from RELCL1 ("whom I met")

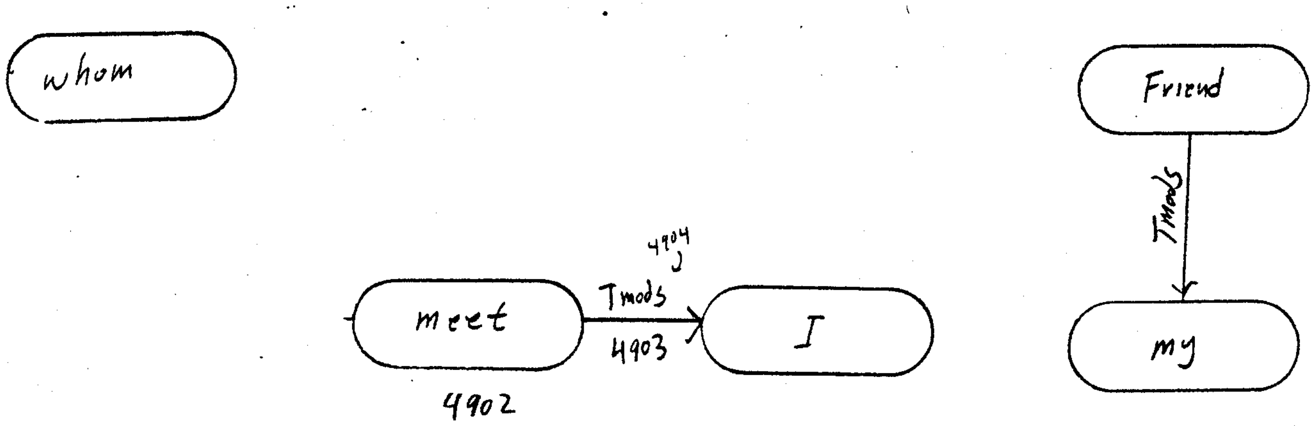
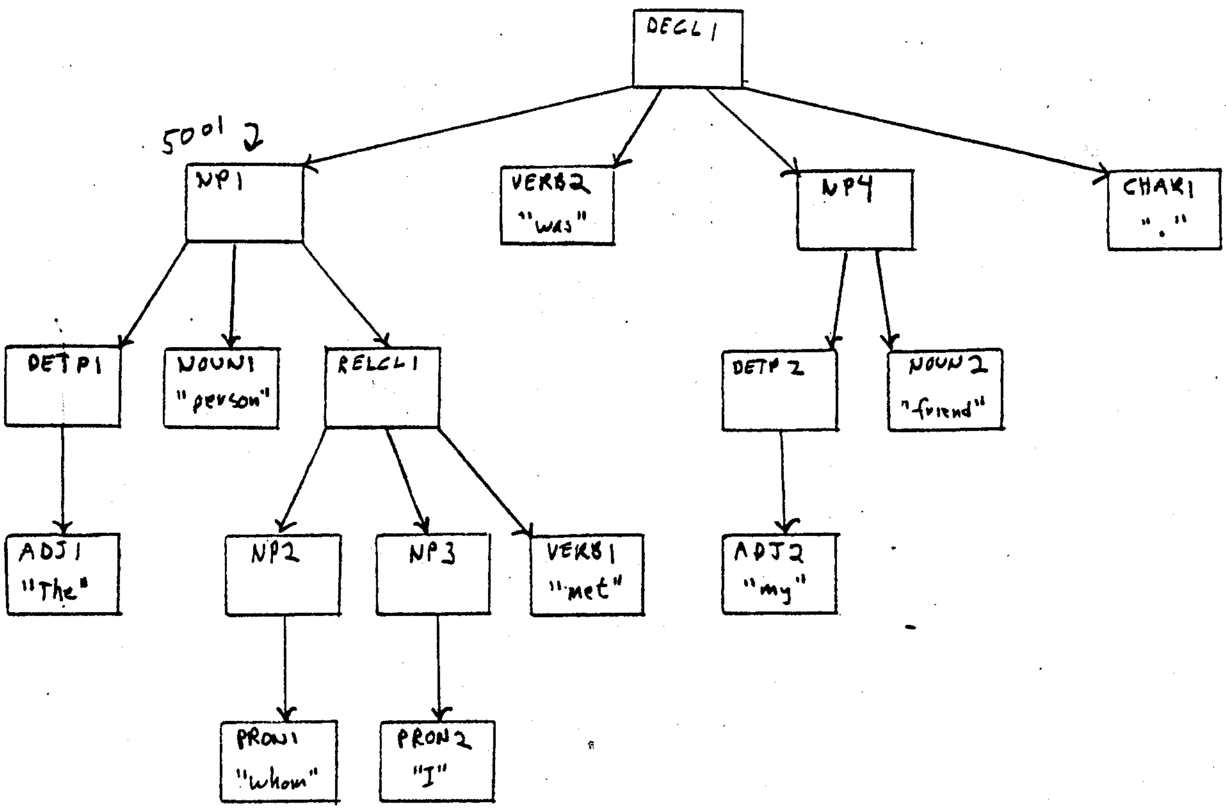


Figure 50



Rule: SyntoSem1 produces logical form graph node "person" from NP1 ("The ..... met")

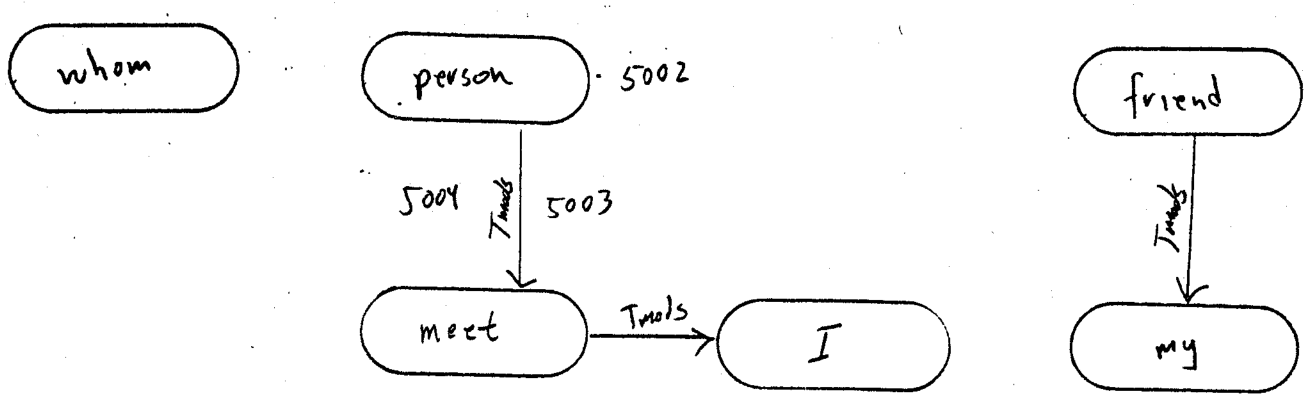
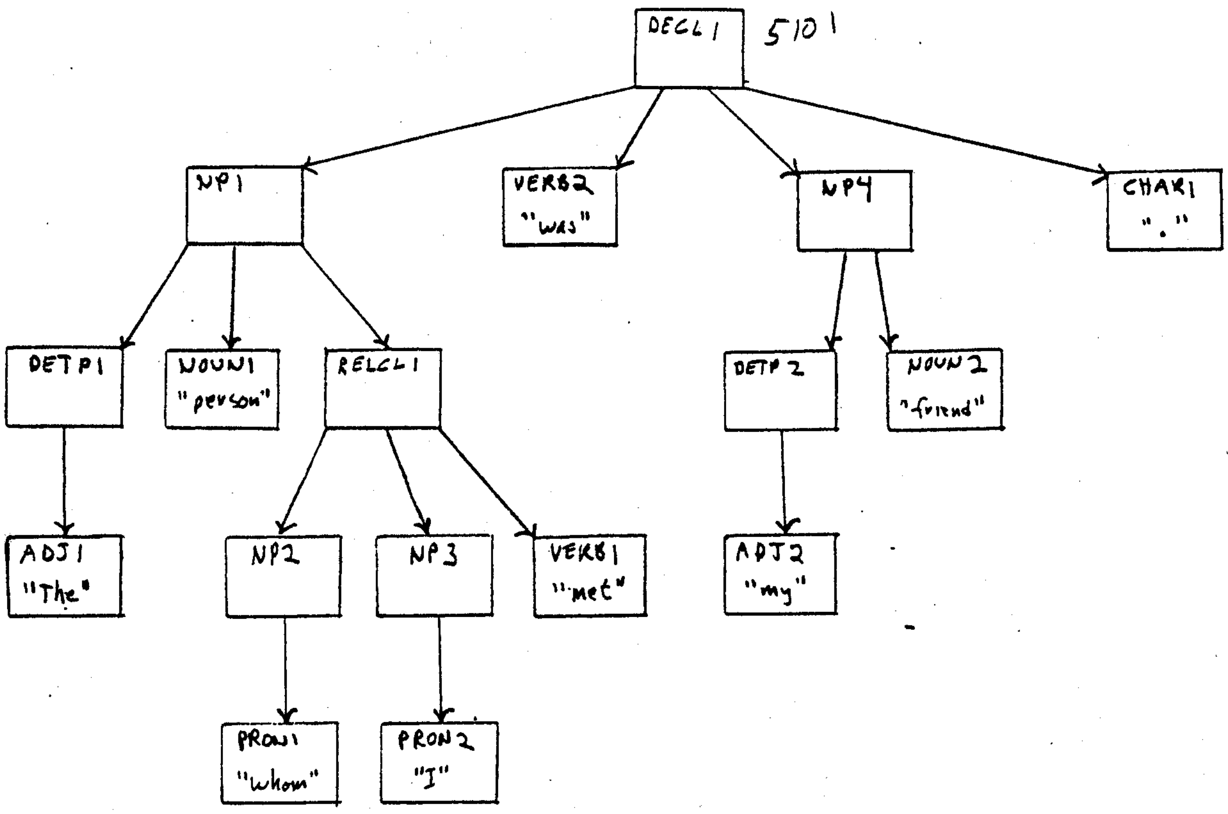


Figure 51



Rule: SynToSem1 produces logical form graph node "be" from DECL1

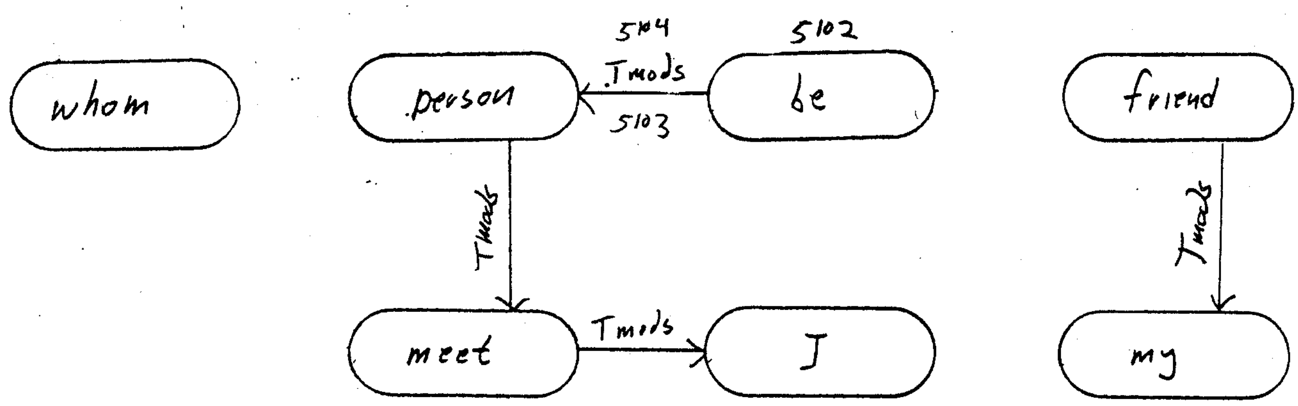
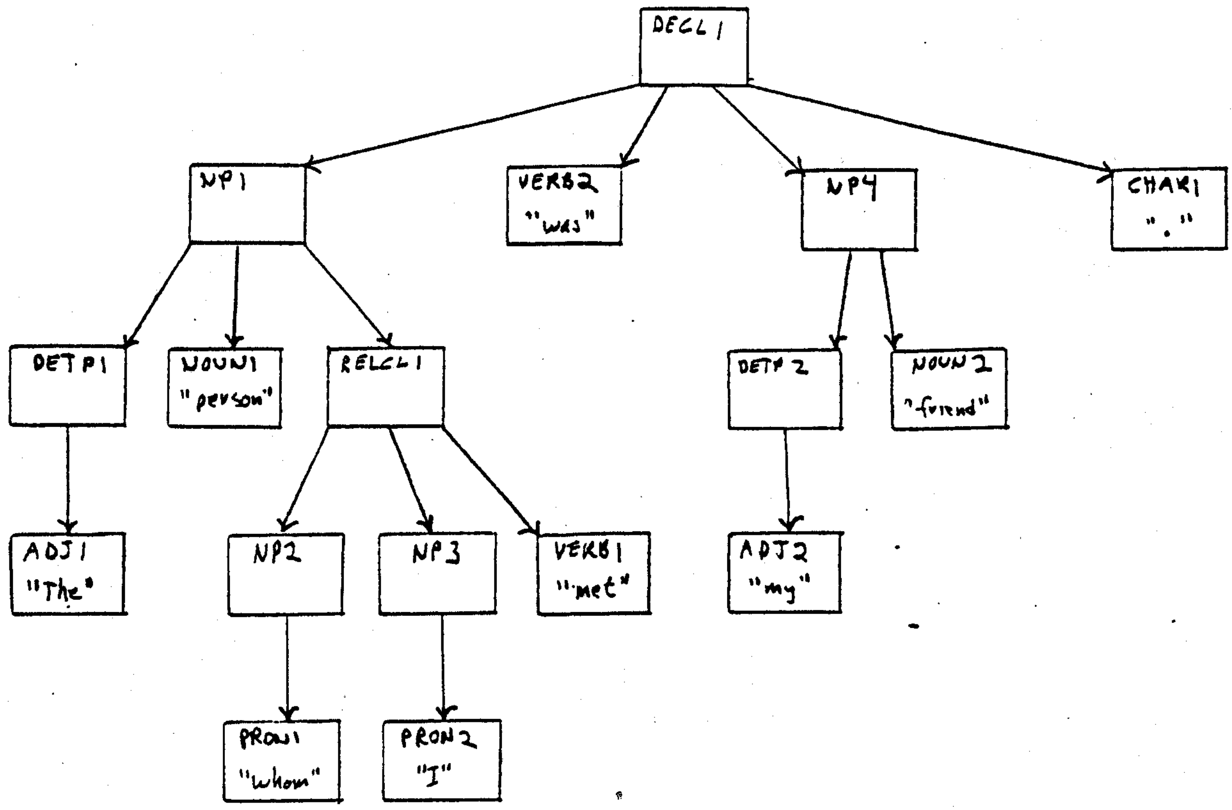


Figure 52

08/674610



Rule: LF-Osub1 with node "be" labels link and creates another link

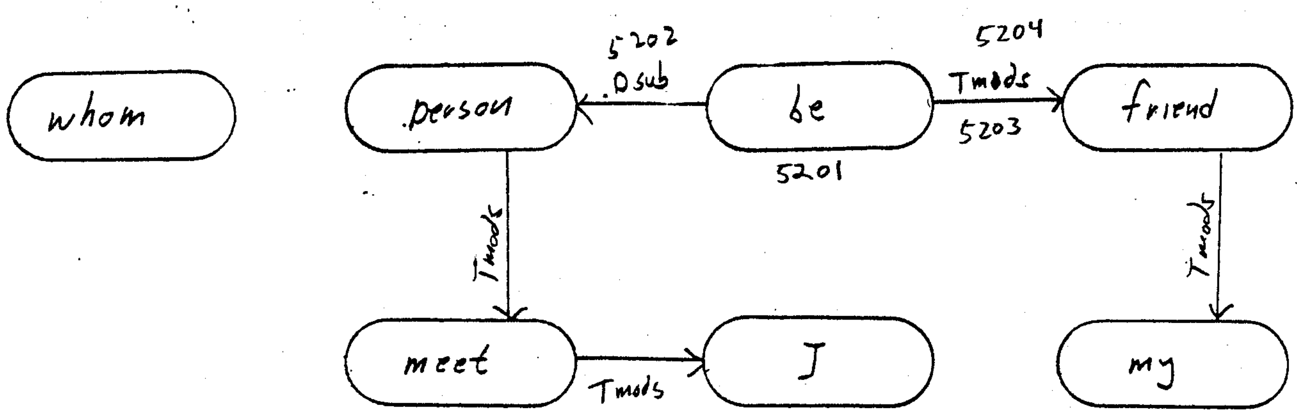
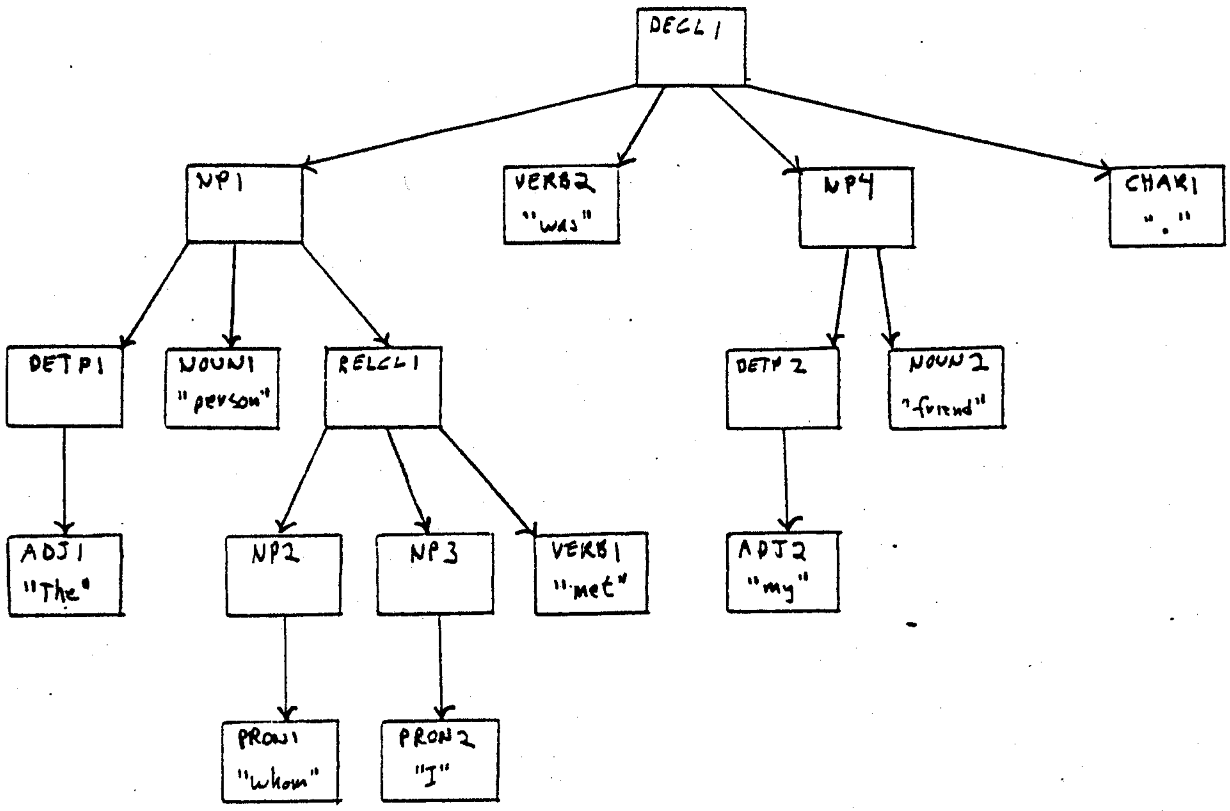


Figure 53



Rule: LF_Dnom with node "be" labels link

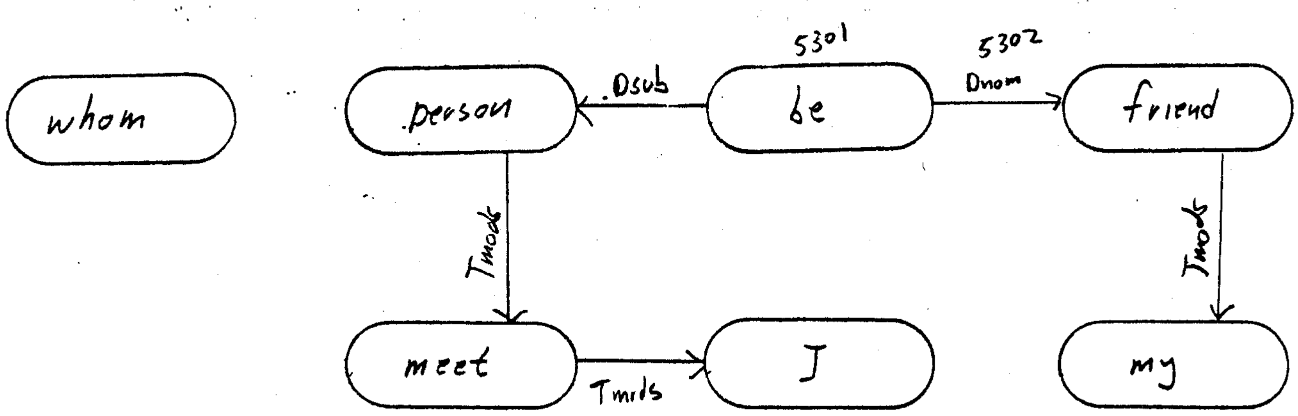
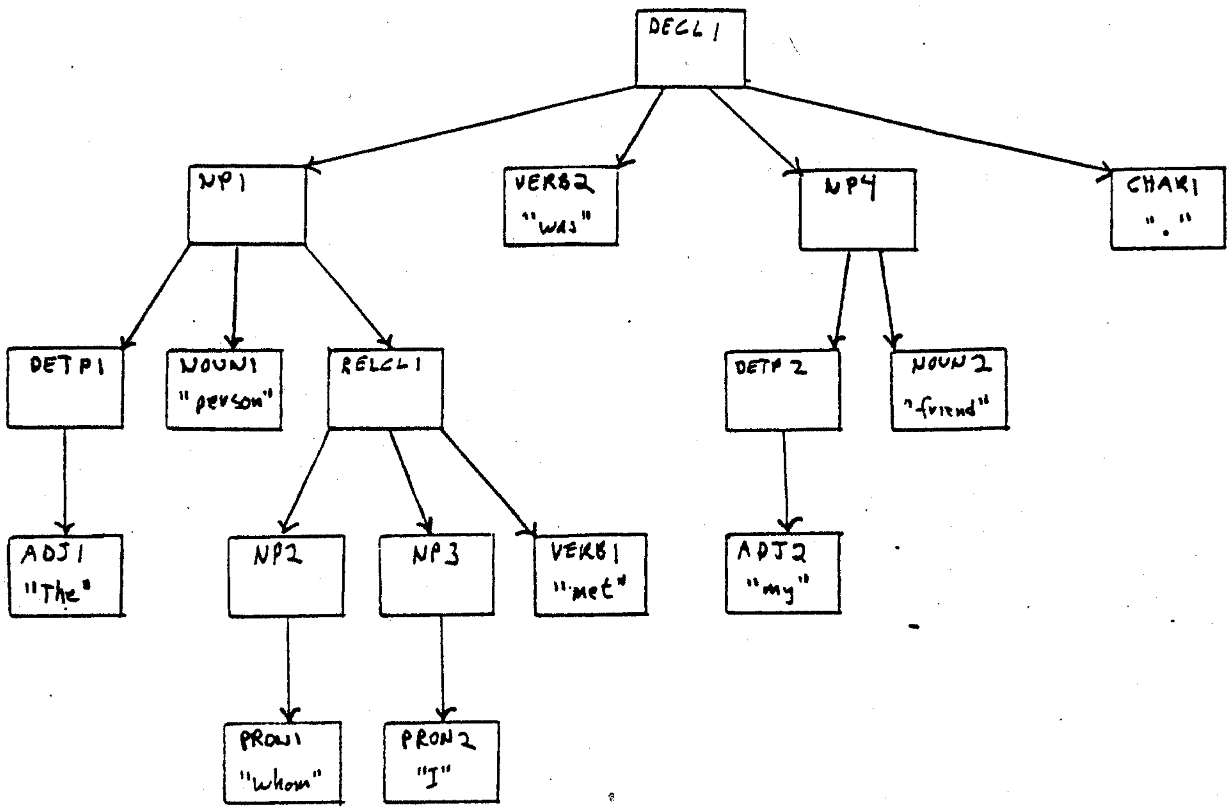


Figure 54



Rule: LF_Props with node "person" labels link

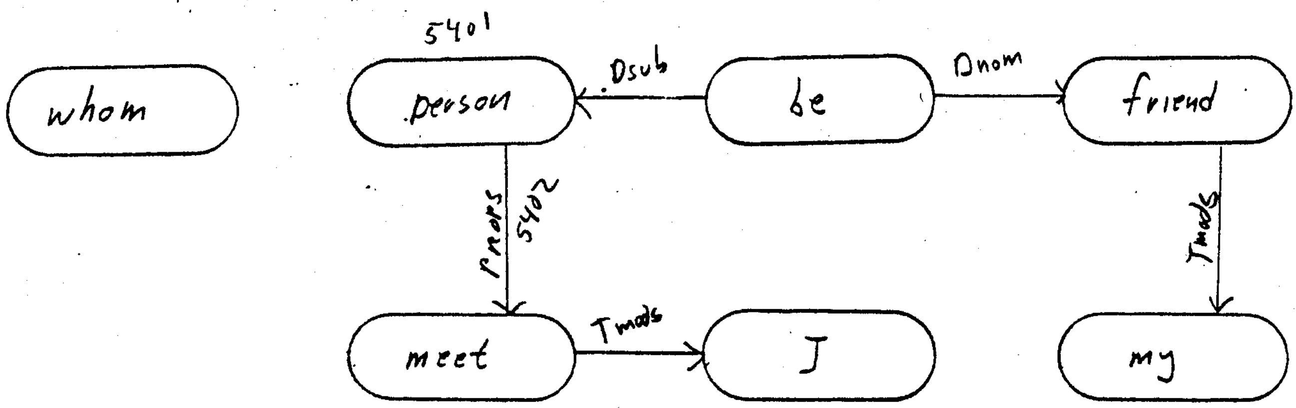
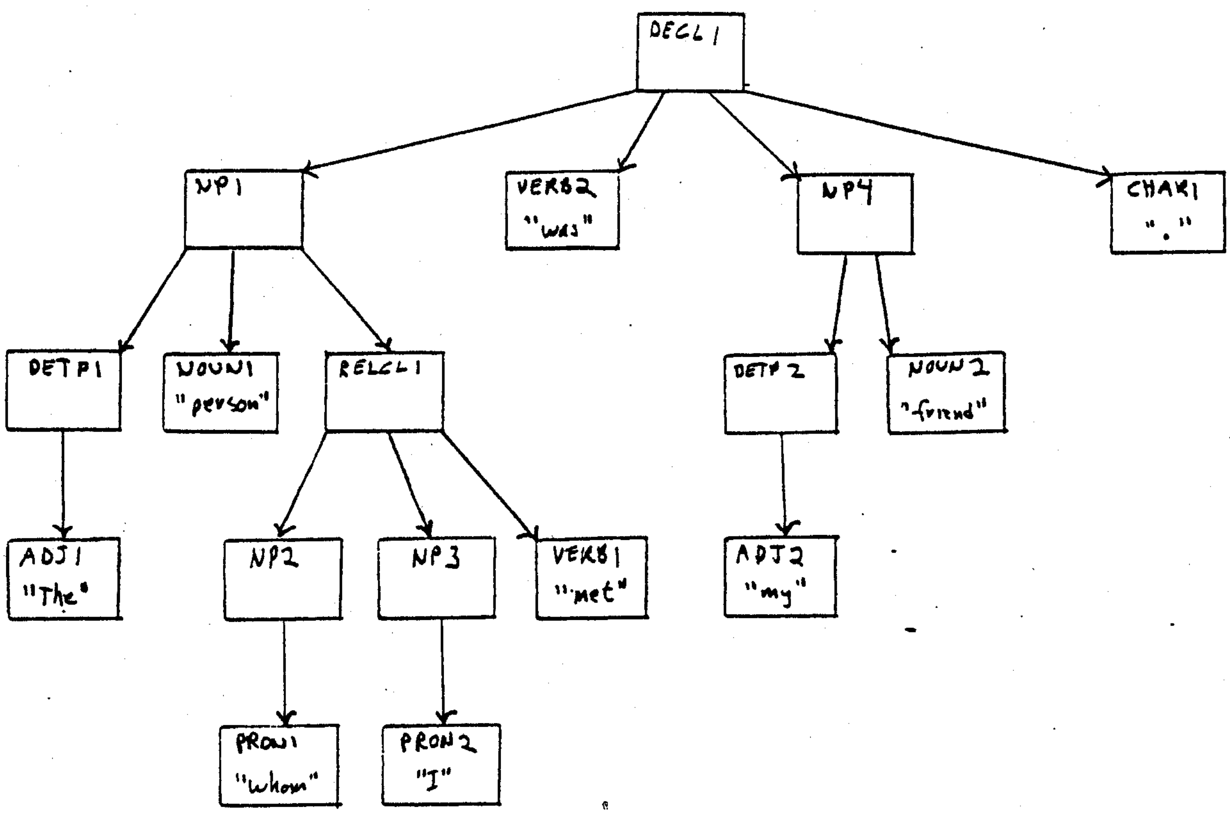
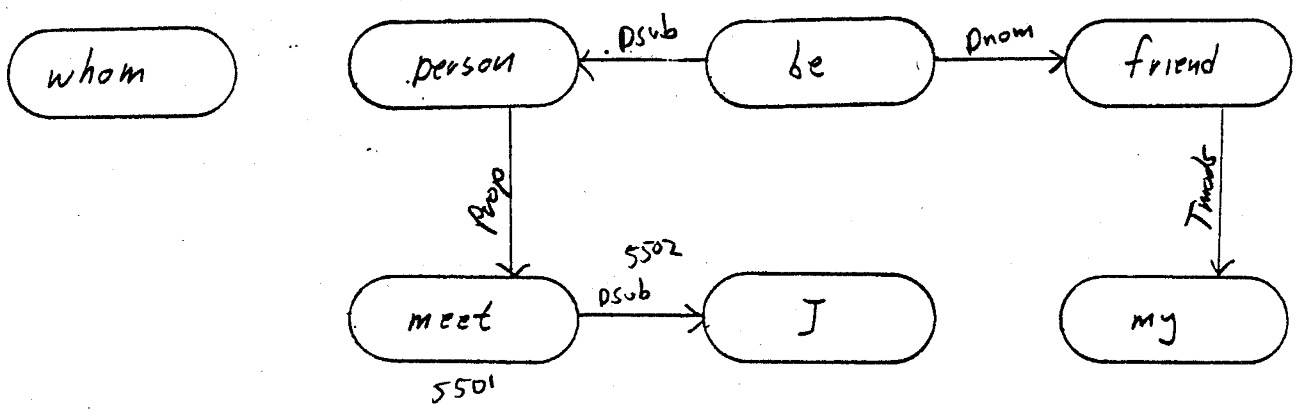


Figure 55

08/674610

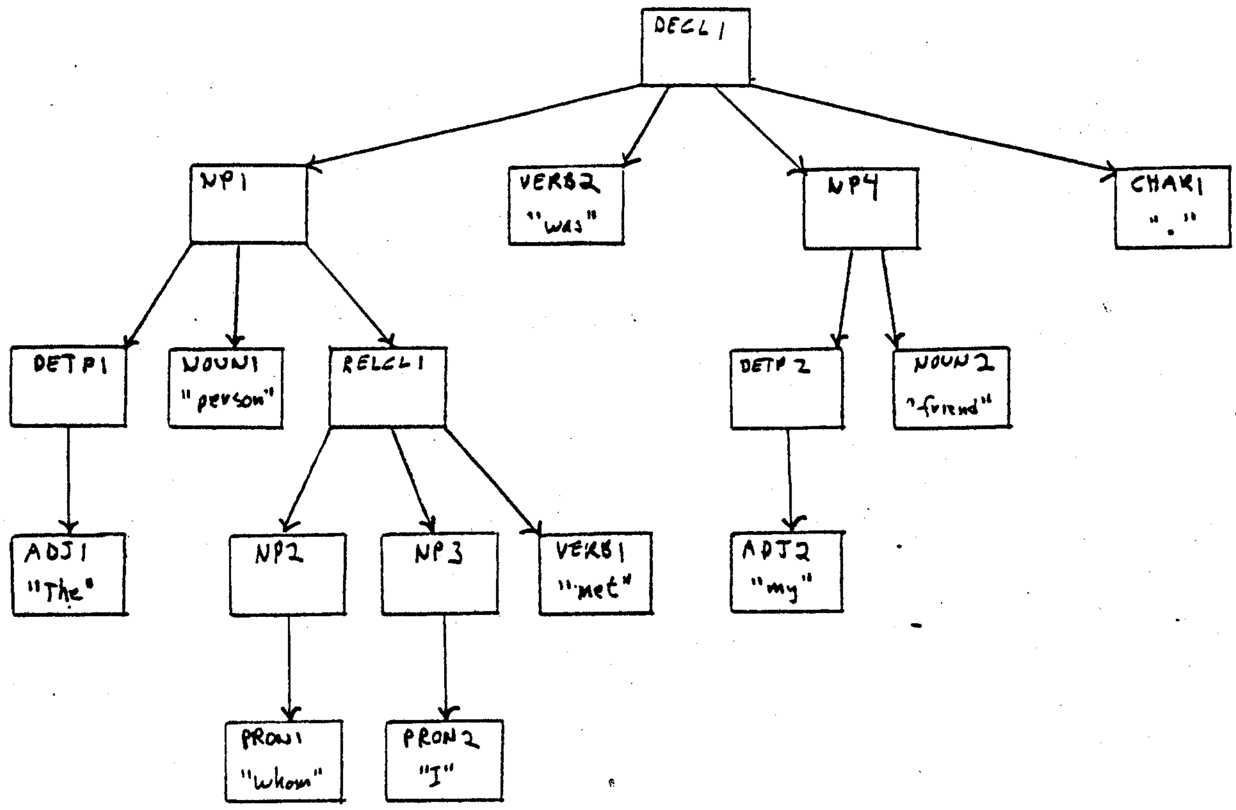


Rule: LF_Dsub1 with node "meet" labels link

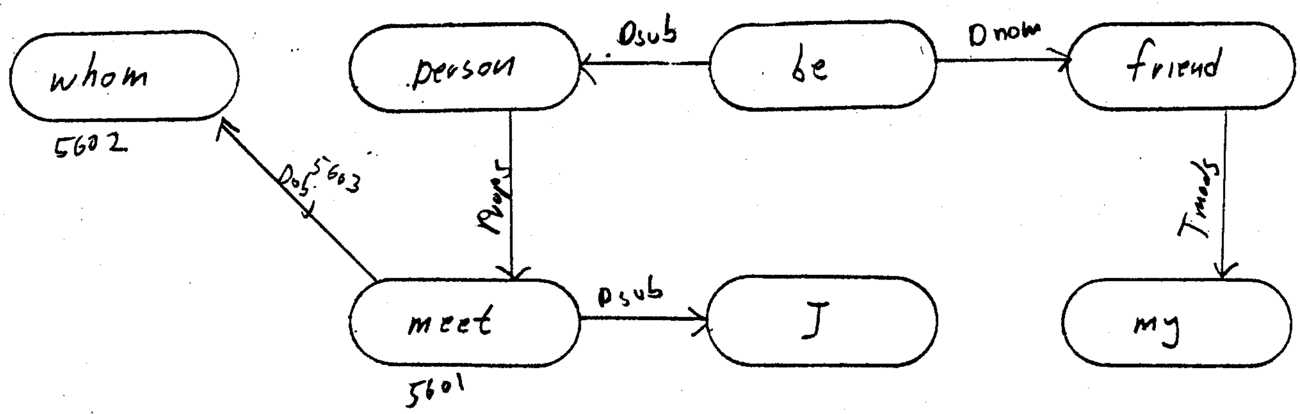


08/674610

Figure 56

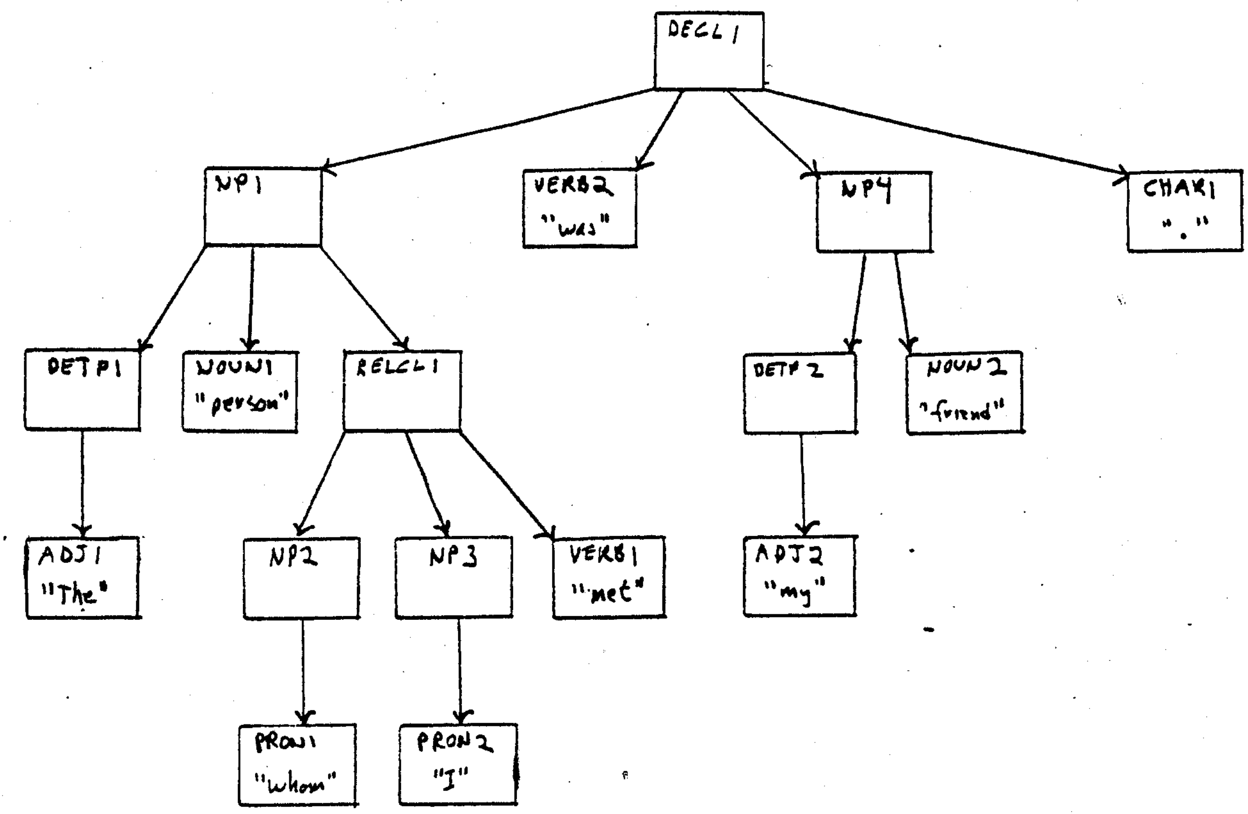


Rule: LF_Dobj1 with node "meet" adds link and labels it



08/674610

Figure 57



Rule: LF_ops with node "friend" labels link

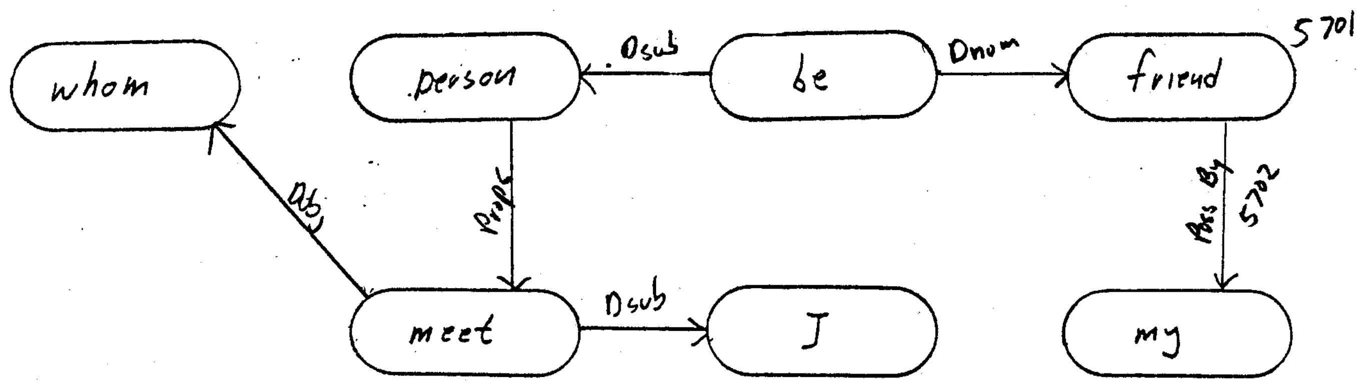
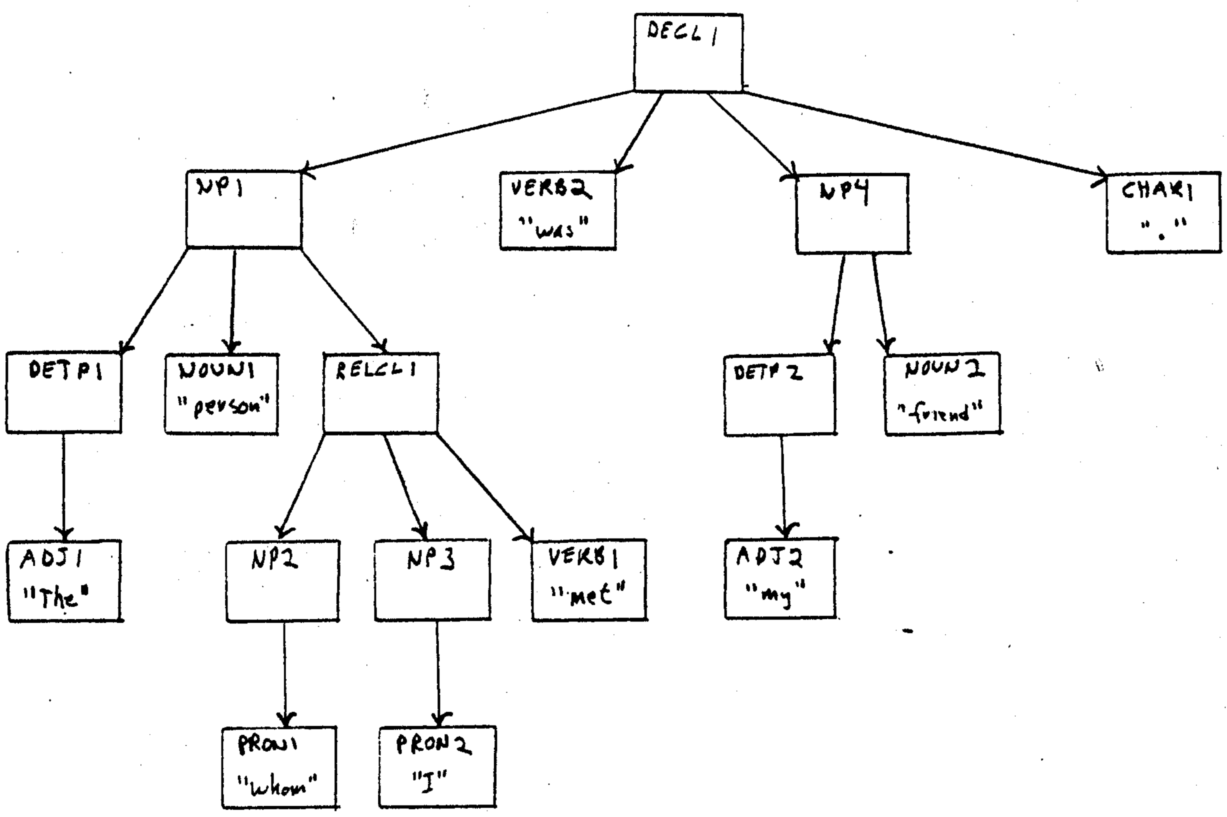


Figure 58



Rule: PsLF_RelPro with node "whom" removes node and adds link

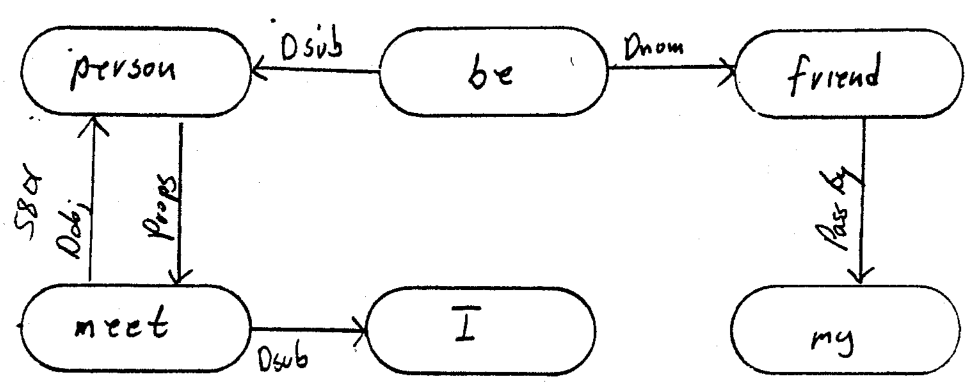
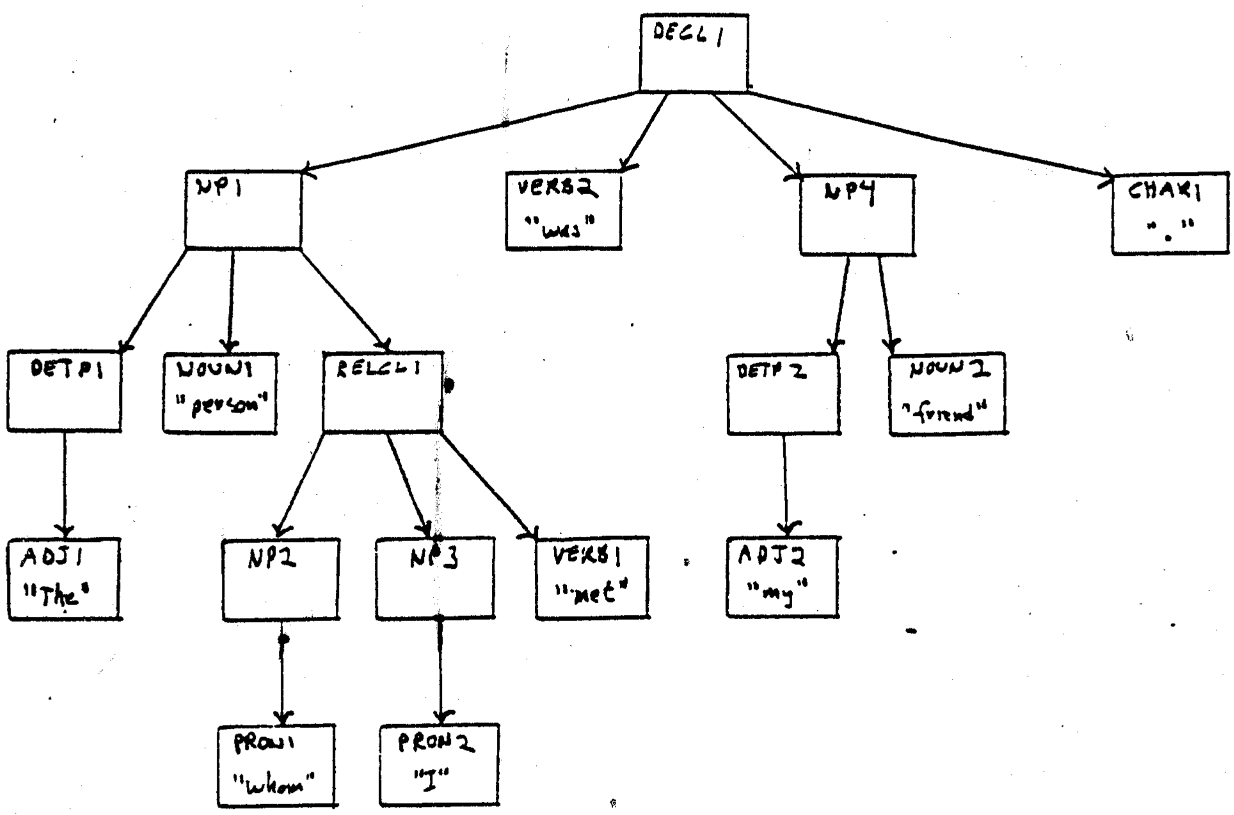
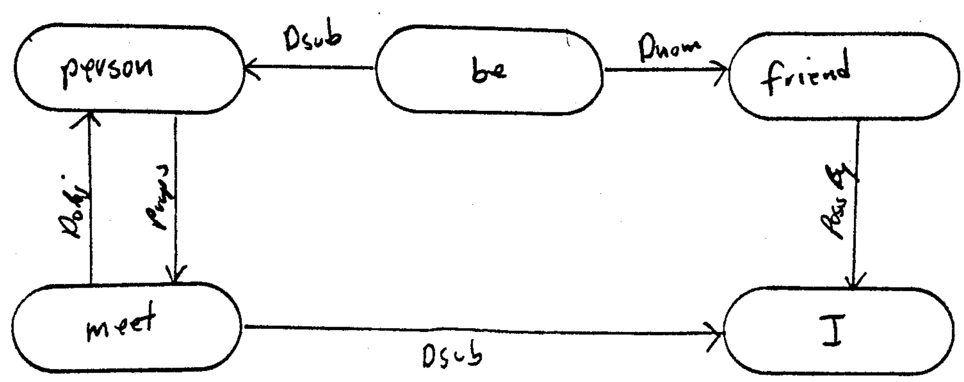


Figure 59



Rule: PsLF_UnifyProns consolidates nodes "I" and "my" into a single node



08/674610

704/9
Thomas
2747

10/69

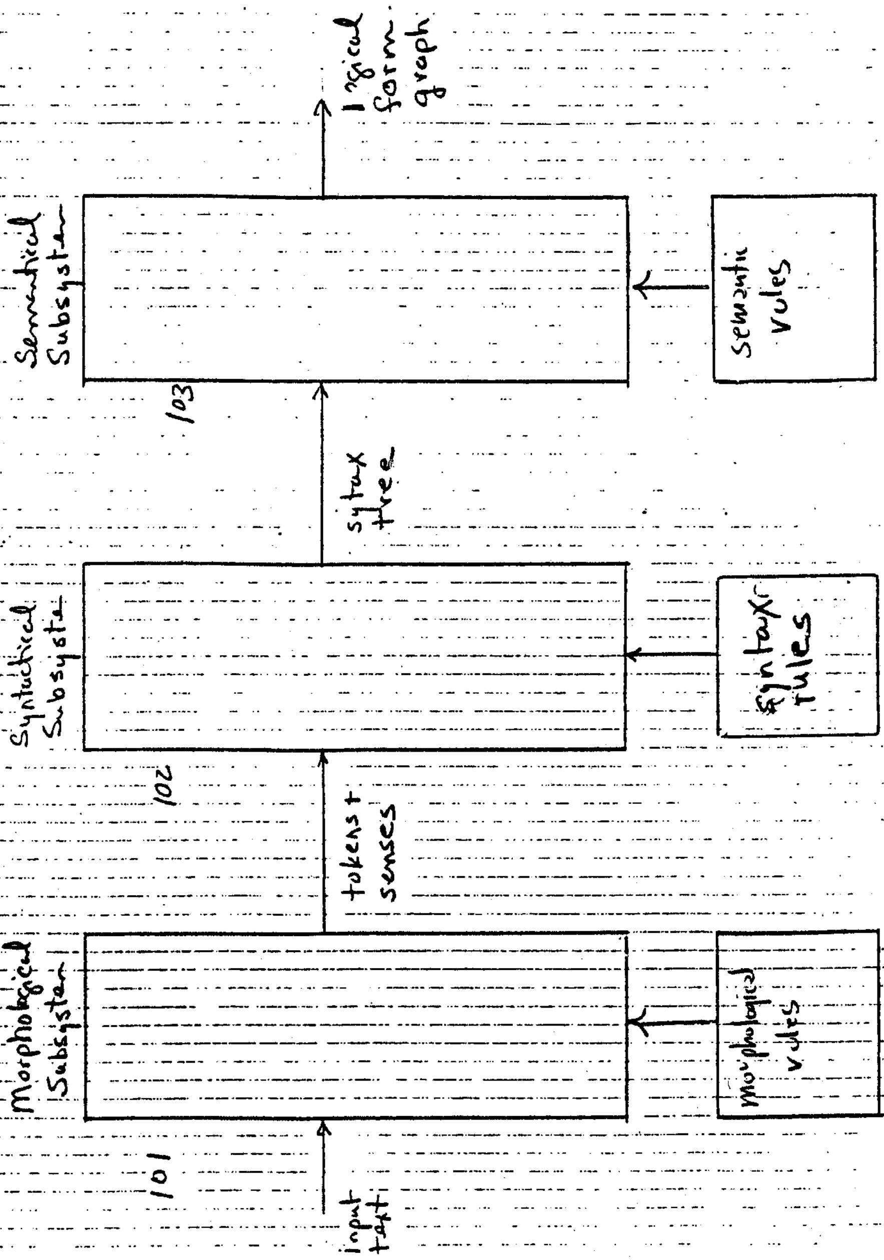


Figure 1

Figure 2

PRINT OF DRAWINGS
AS ORIGINALLY FILED

08/674610

203 201

```
the
{                    204
Adj                  205
      {Lemma        "the"
       Bits         Sing Plur Wa6 Det Art
            212     B0 Def}  206
Adv
      {Lemma        "the"
       Bits         Wa5}
Senses   208
      {Bits         Sing Plur Wa6 Closed
                    Det Art Def
207    Lemma        "the"
       Cat          Adj
       Infl         Adj-nil
       Defin        "used when it is clearly understood who or what is meant"
       Exs          "We have a cat and a dog. The cat (= our cat) is black and the dog (=
our dog) white."
                    "the history of China (= Chinese history)"
                    "The Danes that I know work very hard."
                    "Take these letters to the post office (it is understood that you know
which post office and where it is)"} 209

  210 {Lemma        "the"
       Cat          Adv
       Defin        "To that extent; by that much"
       Exs          "the sooner the better."} 211

      (more sense records) 213
}
```

202

```
person
{
Noun
      {Lemma        "person"
       Bits         Pers3 Sing Humn Mass
                    Anim Count Conc C9
                    Humn_sr
       Infl         Noun-default }
Senses
      {Lemma        "person"
       Cat          Noun
       Defin        "A living human being."
       Exs          "chairperson"
                    "spokesperson"
                    "salesperson."}
      (more sense records)

}
```

```
whom
{Pron
       {Lemma       "who"
        Bits        Pers3 Sing Plur Rel Wh
                    Humn Obj Anim}
Senses
       {Lemma       "who"
        Bits        Pers3 Sing Plur Rel Wh
                    Closed Humn Obj Anim
        Cat         Pron
        Defin       "(the object form of who, used esp. in writing and careful speech)"
        Exs         "With whom?"
                    "The man with whom he talked."
                    "You saw whom?"
                    "Whom did they see?"
                    "the man (whom) they saw arriving"
                    "a man (whom) you may know of"}
      (more sense records)

}
```

```
i
{
 Noun
       {Lemma       "i"
        Bits        Pers3 Sing TakesAn
        Infl        Noun-irreg}
Pron
       {Lemma       "I"
        Bits        Sing Nom TakesAn Pers1
                    Humn Anim LexCap}
Senses
       {Lemma       "i"
        Cat         Noun
        Infl        Noun-irreg
        Defin       "The ninth letter of the modern English alphabet."}

       {Lemma       "I"
        Cat         Pron
        Defin       "Used to refer to oneself as speaker or writer."}
      (more sense records)
}
```

```
met
{
 Verb
       {Lemma       "meet"
        Bits        Sing Plur Past
                    Pastpart
        Infl        Verb-meet }
 Senses
       {Lemma       "meet"
        Bits        Past Pastpart
        Cat         Verb}

  }
```

Figure 4

PRINT OF DRAWINGS AS ORIGINALLY FILED

08/674610

```
was
{
 Verb
       {Lemma        "be"
        Bits         Pers3 Sing Past Pers1
        Infl         Verb-be }}
 Senses
       {Lemma        "be"
        Bits         Past Pastpart
        Cat          Verb}
       (more sense records)
}
```

```
my
{
 Adj
       {Lemma        "I"
        Bits         Wa5 Det Poss Pers1 Def
                     Gen A0
        Infl         Adj-none }
 Ij
       {Lemma        "my" } }
 Senses
       {Lemma        "I"
        Bits         Wa5 Closed Det Poss
                     Pers1 Def Gen A0
        Cat          Adj
        Infl         Adj-none
        Defin        "belonging to me"
        Exs          "my car"
                     "my mother"}

       {Cat          Ij
        Defin        "Used as an exclamation of surprise, pleasure, or dismay"
        Exs          "Oh, my! What a tiring day!"}

       (more sense records)
}
```

PRINT OF DRAWINGS
AS ORIGINAL FILED

Figure 5

08/574610

```
friend
{
 Noun
       {Lemma        "friend"
        Bits         Pers3 Sing Humn Anim
                     Count Conc Humn_sr N0
                     Wrdy
        Infl         Noun-default
        Vprp         (of to)
        Bitrecs
              {Bits        Humn Count Conc
               Vprp        (of) }

              {Bits        Humn Count Conc
               Vprp        (to) } }
 Verb
       {Lemma        "friend"
        Bits         Inf Plur Pres T1
        Infl         Verb-default } }
Senses
       {Lemma        "friend"
        Bits         Humn Conc
        Cat          Noun
        Defin        "A person whom one knows, likes, and trusts."}

       {Bits         T1
        Lemma        "friend"
        Cat          Verb
        Infl         Verb-default
        Defin        "To befriend."}

       (more sense records)

}
```

Figure 6

1610

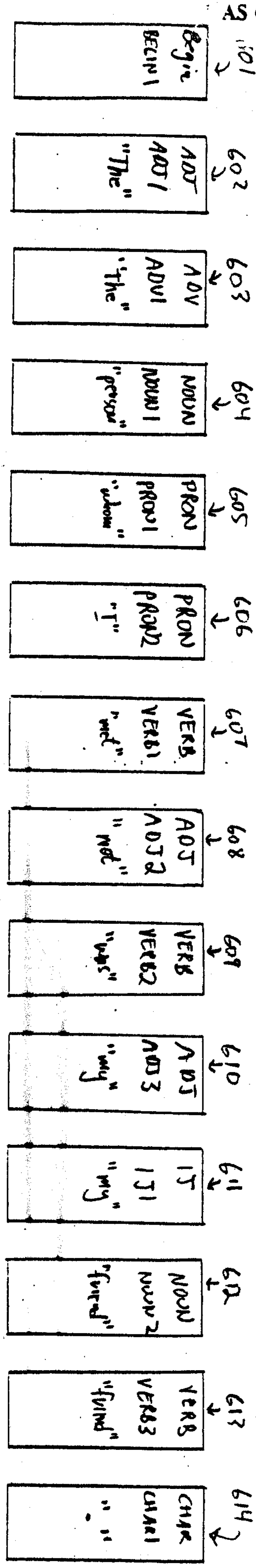


Figure 7

08/674610

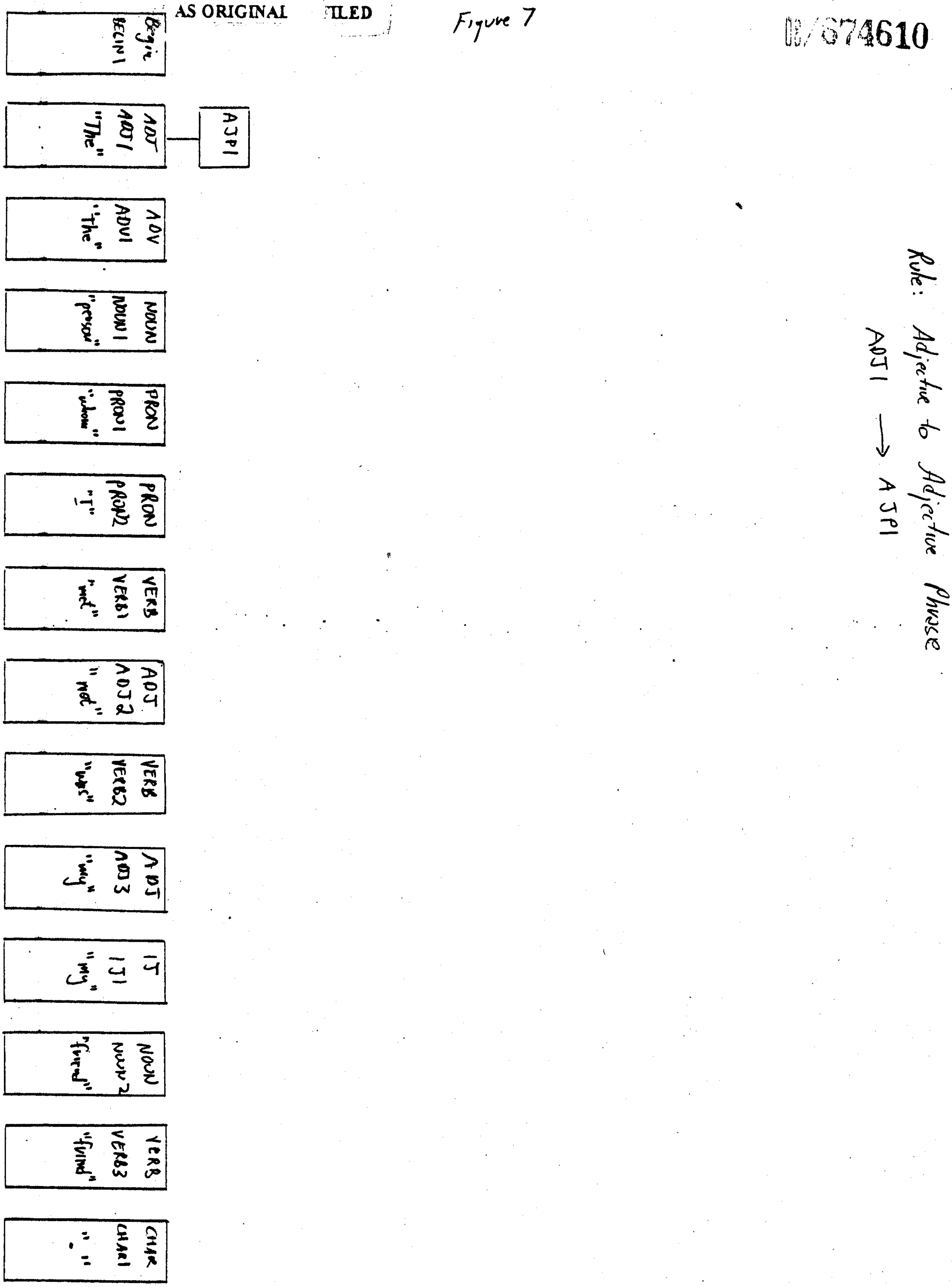


Figure 8

08/674610

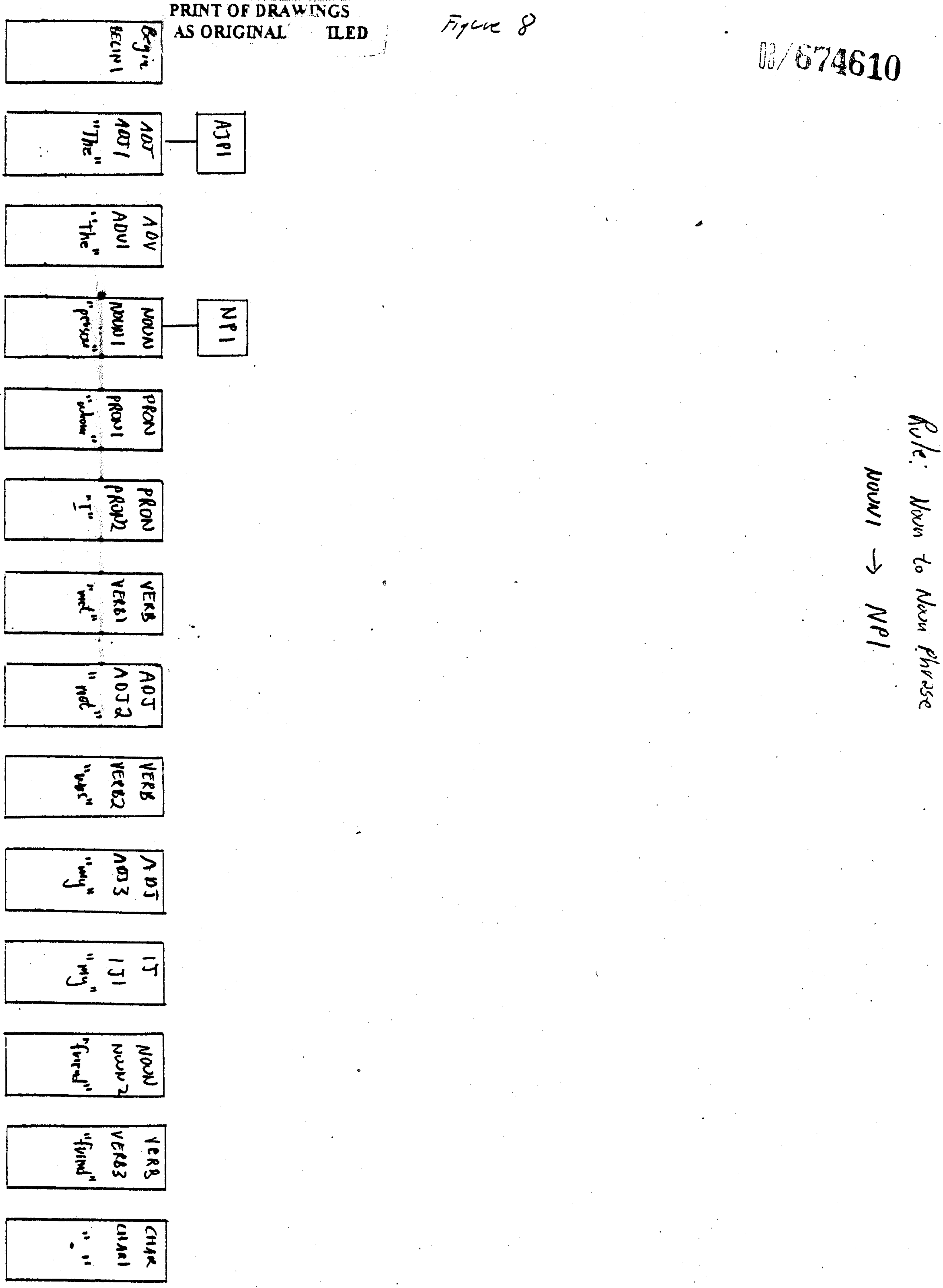
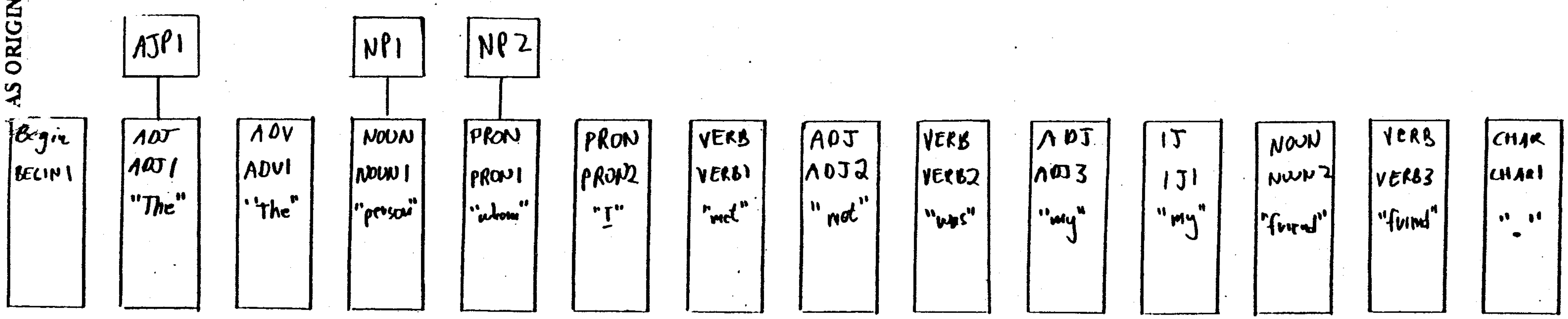


Figure 9

PRINT OF DRAWINGS AS ORIGINAL TLED

Rule: Pronoun to Noun Phrase

PRON1 → NP2



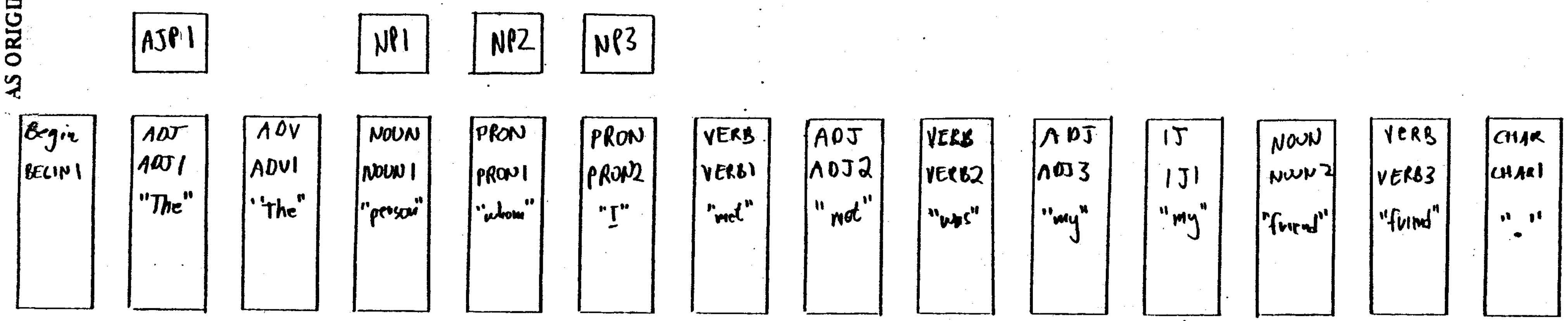
08/674610

Figure 10

PRINT OF DRAWINGS
AS ORIGINAL FILED

Rule: Pronoun to Noun Phrase

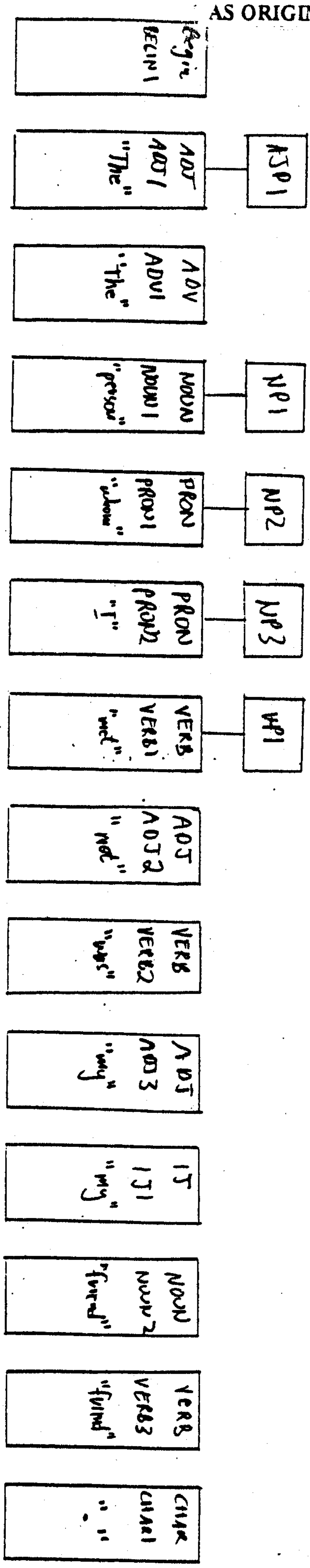
PRON2 → NP3



PRINT OF DRAWINGS AS ORIGINALLY FILED

08/674610

Figure 11



Rule: Verb to Verb Phrase

VERB1 → VP1

PRINT OF DRAWINGS
AS ORIGINAL FILED

Figure 12

08/674610

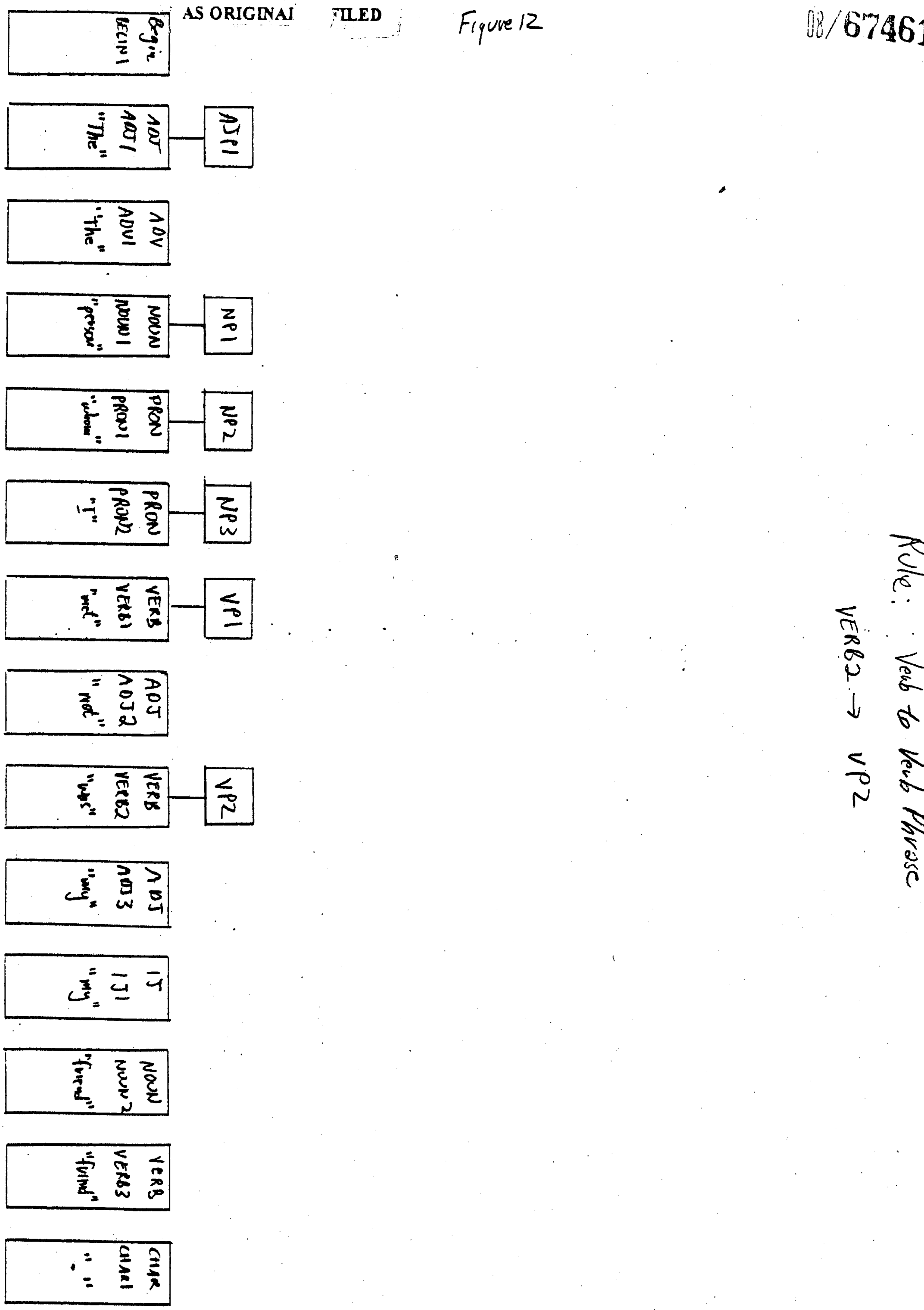
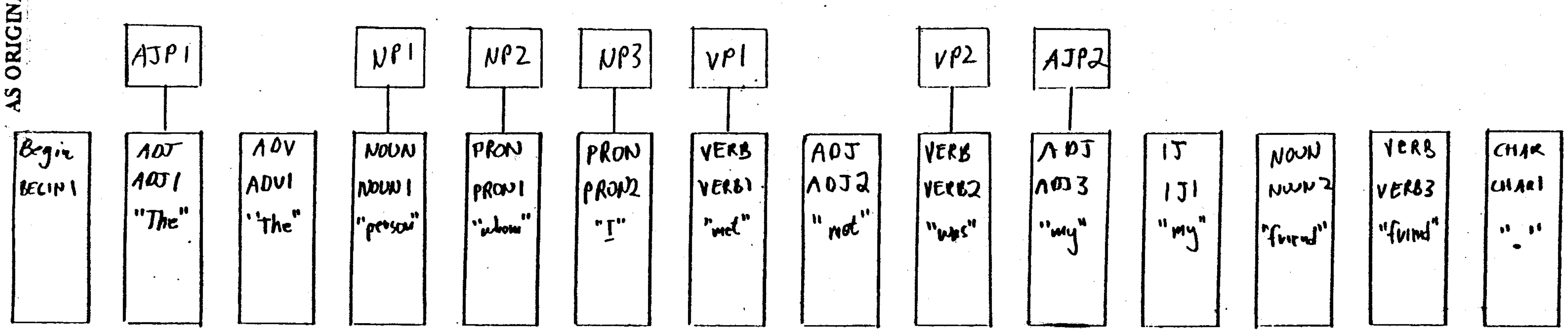


Figure 13

08/674610

Rule: ADJECTIVE to ADJECTIVE PHRASE

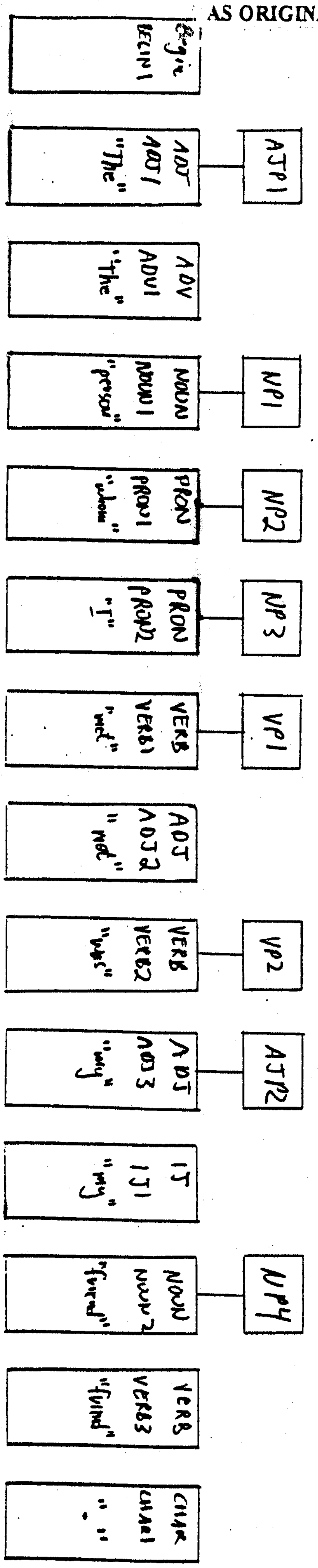
ADJ3 → AJP2



PRINT OF DRAWINGS
AS ORIGINALLY FILED

Figure 14

08/674610



Rule: Noun to Noun phrase

PRINT OF DRAWINGS AS ORIGINALLY FILED

08/674610

Figure 15

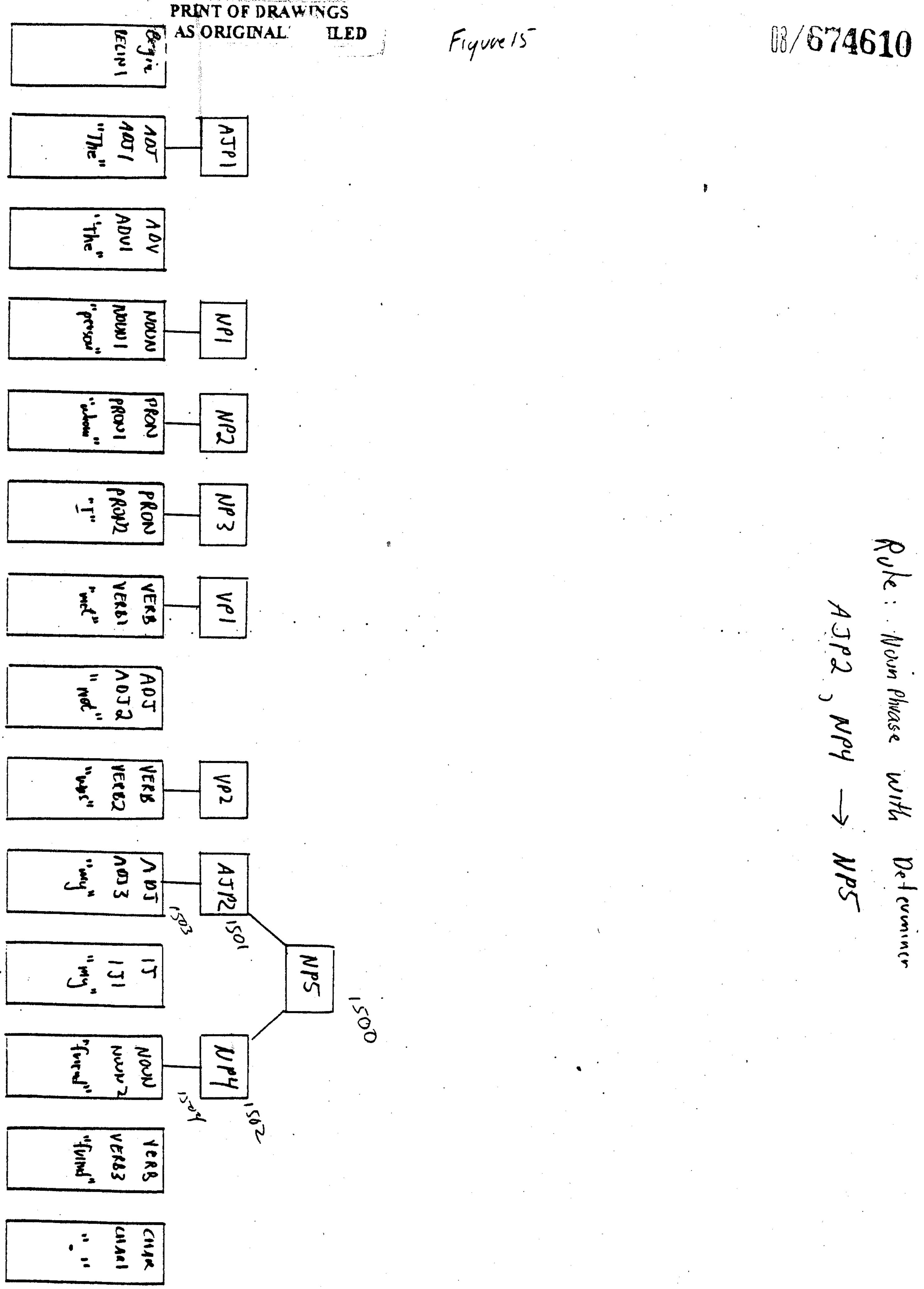


Figure 16

Rule: Verb Phrase with Noun Phrase as Object of Transitive Verb

VP2, NP5 → VP3

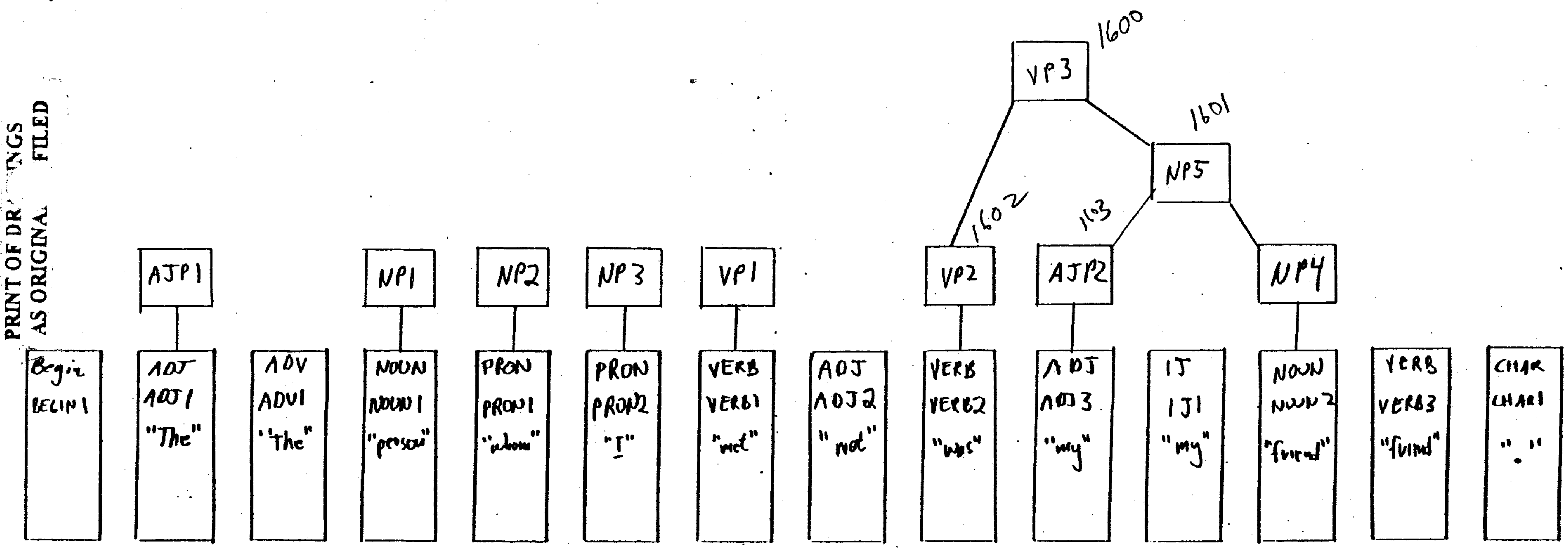
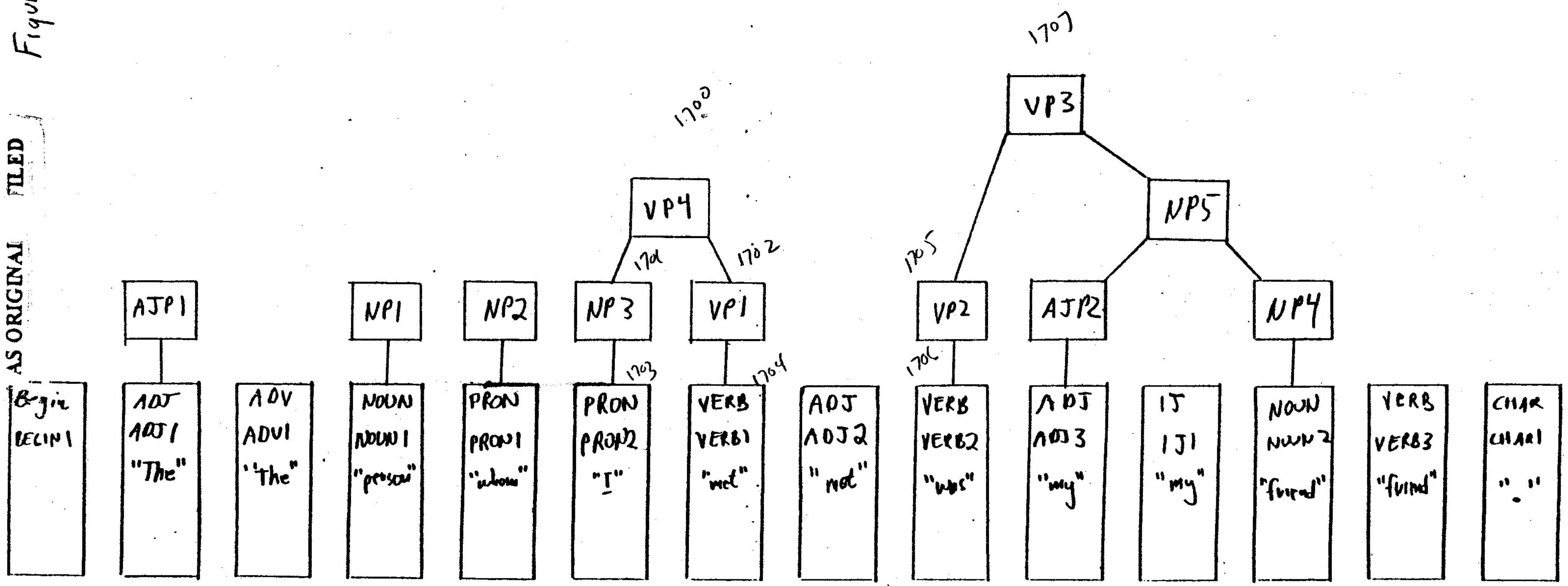


Figure 17

Rule: Verb Phrase with Noun Phrase as Subject

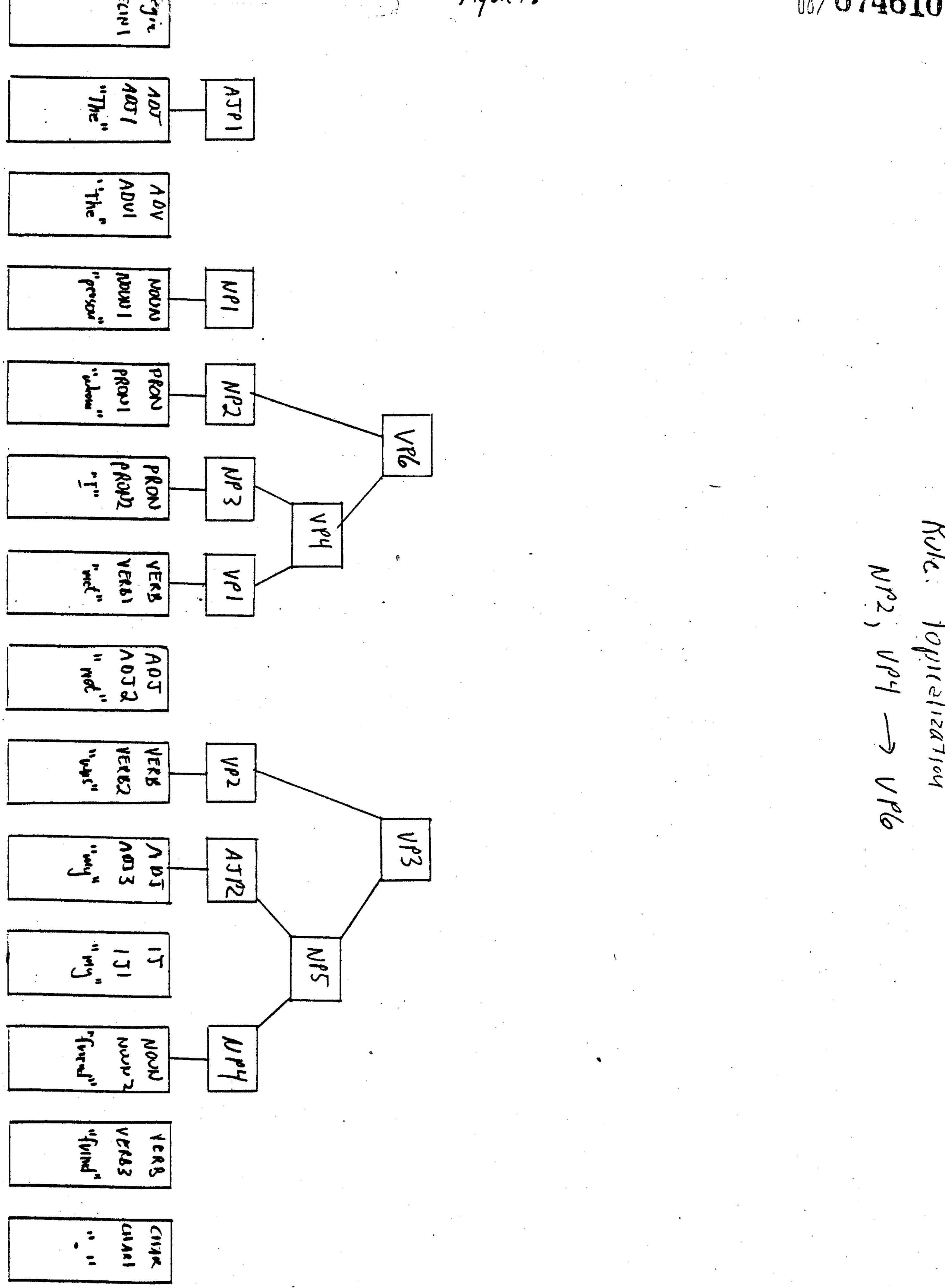
NP3; VP1 → VP4



PRINT OF DRAWINGS
AS ORIGINALLY FILED

08/674610

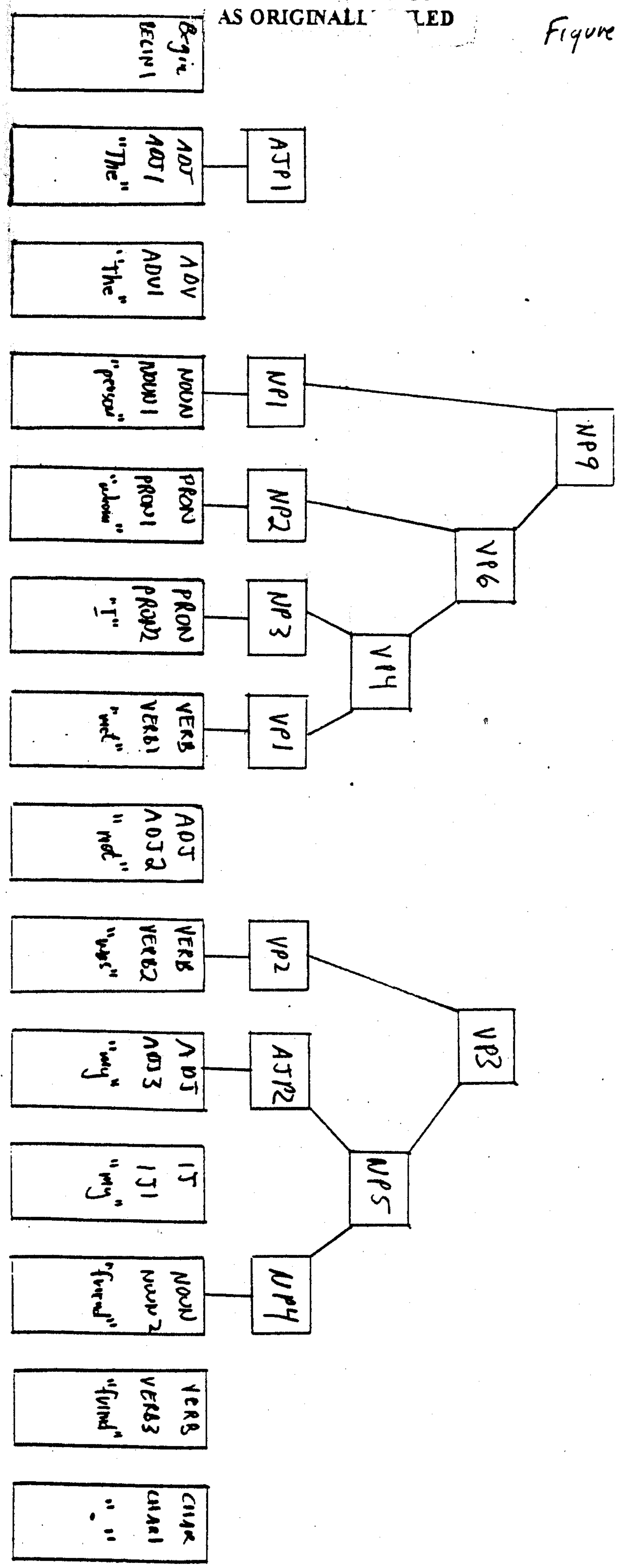
Figure 18



PRINT OF DRAWINGS
AS ORIGINALL[illegible]LED

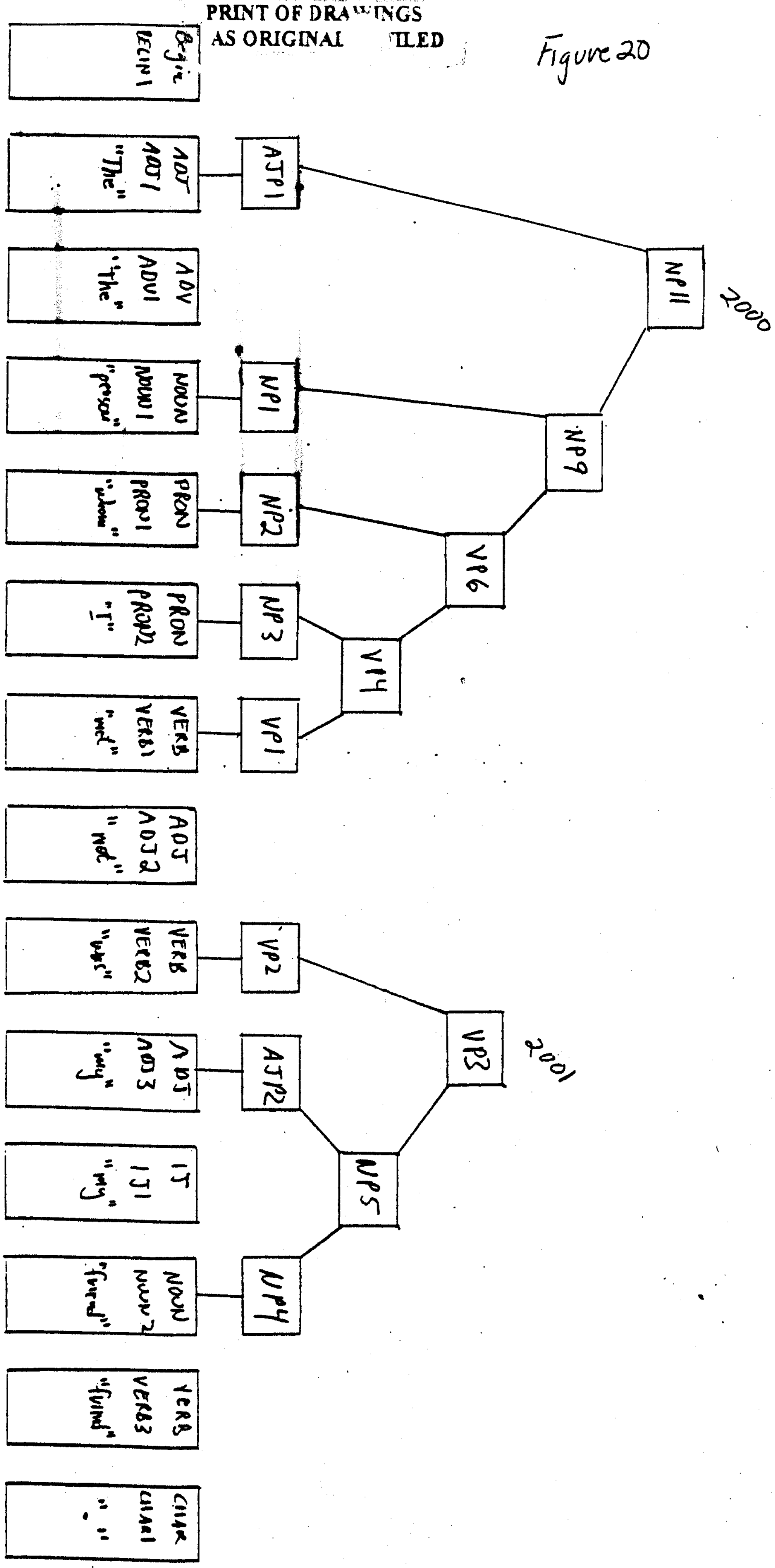
Figure 19

08/674610



Rule: Noun Phrase with Relative Clause
NP1, VP6 → NP9.

Figure 20

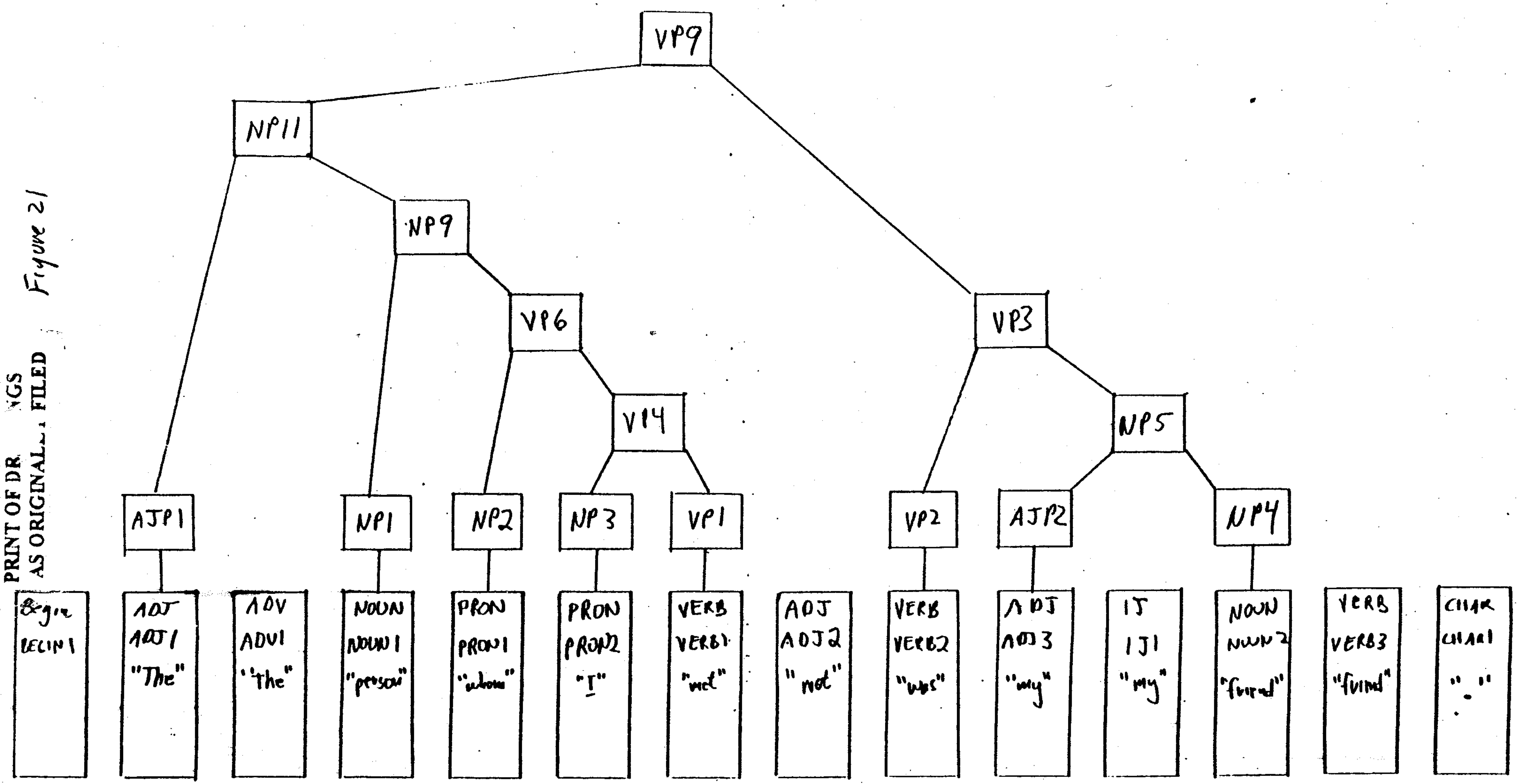


Rule: Noun Phrase with Determinate Quantifier
AJP1, NP9 → NP11

Rule: Verb Phrase with Noun Phrase Subject

NP11, VP3 → VP9

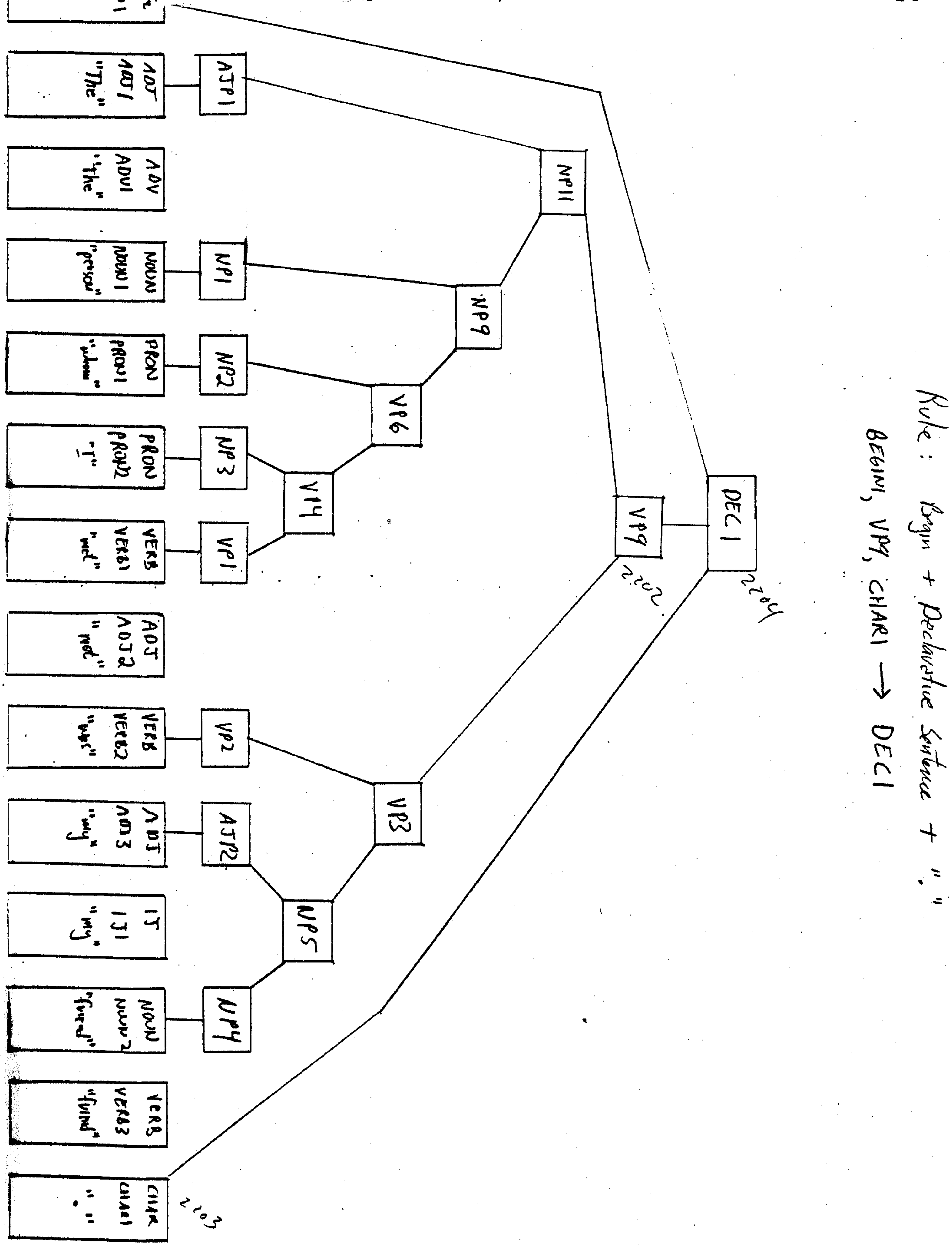
Figure 21



PRINT OF DRAWINGS
AS ORIGINAL[illegible]ILED

Figure 22

08/674610



PRINT OF DRAWINGS
AS ORIGINALLY FILED

08/674610

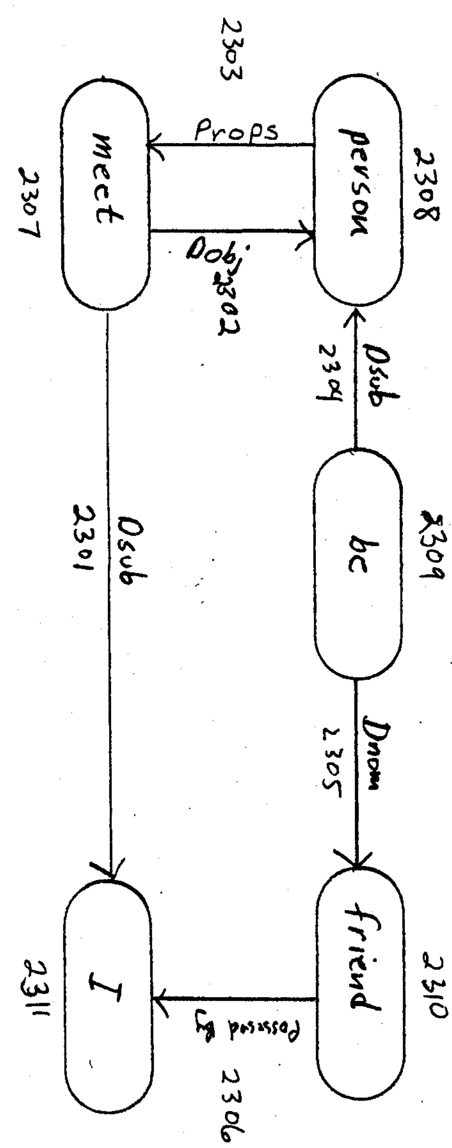
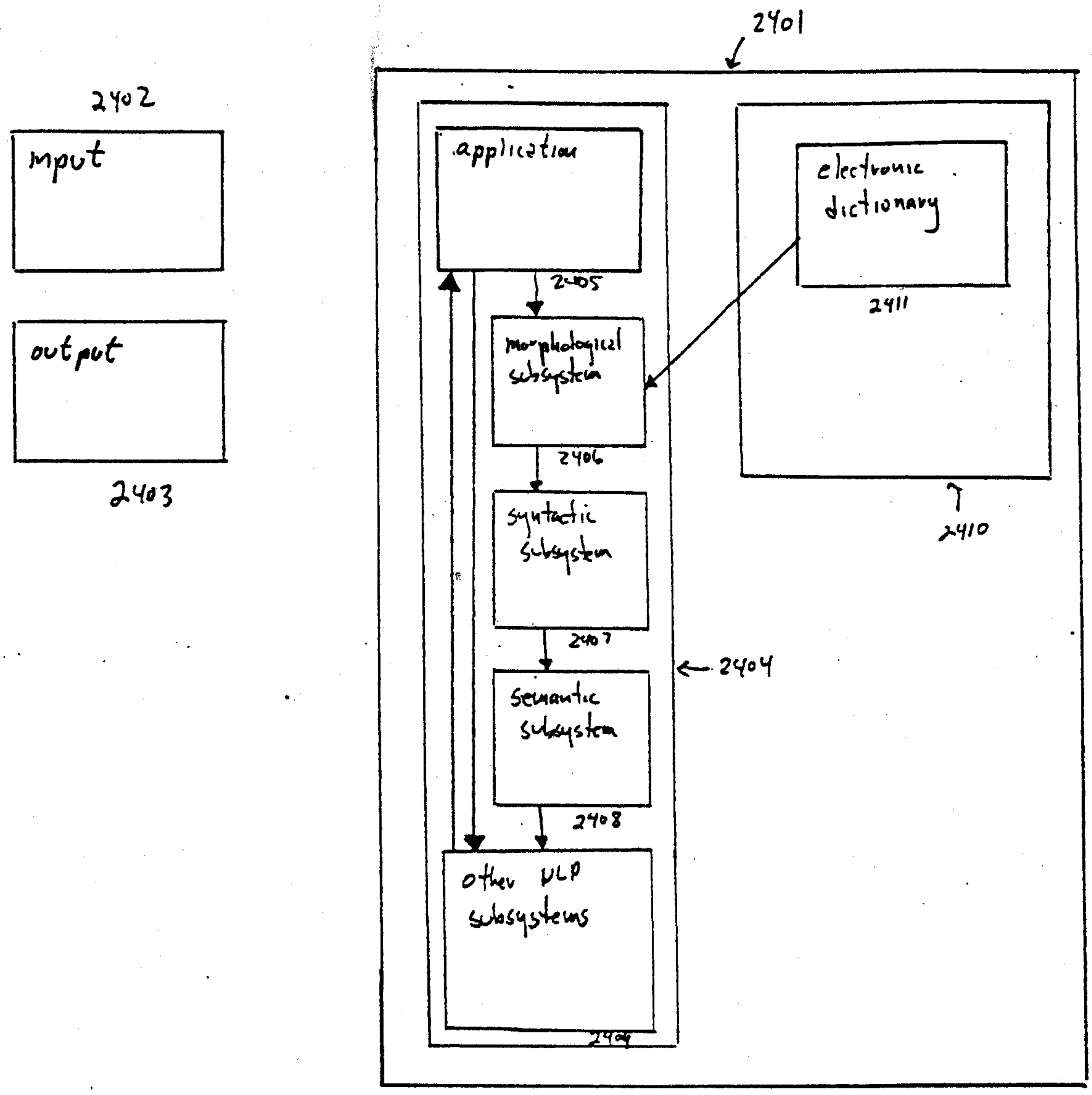


Figure 23

PRINT OF DRAWINGS
AS ORIGINALLY FILED

Figure 24

08/674610



PRINT OF DRAWINGS
AS ORIGINAL FILED

08/674610

The New Semantic Subsystem

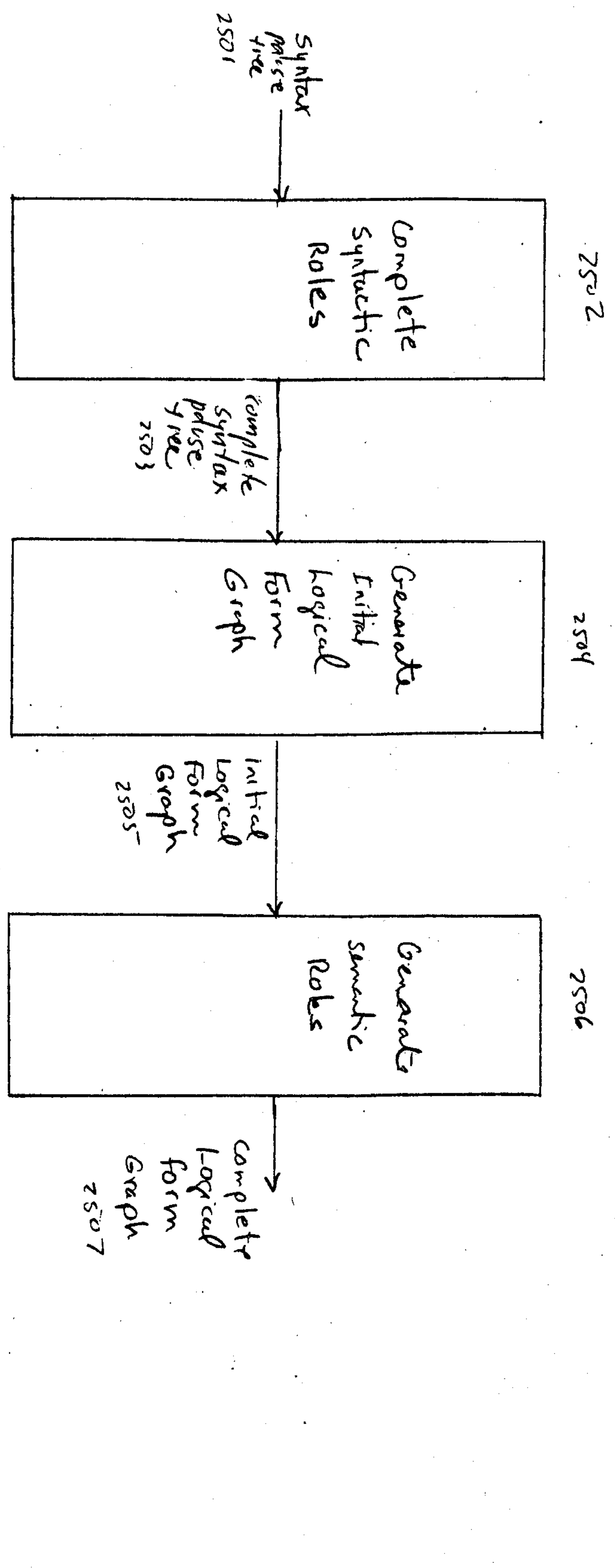


Figure 25

PRINT OF DRA GS
AS ORIGINALL ILED

08/674610

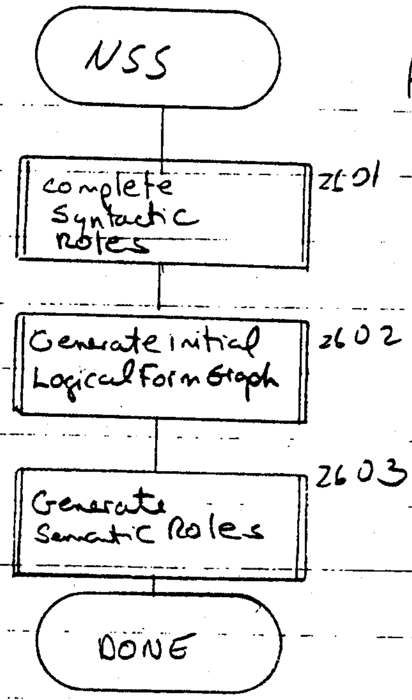


Figure 26

Figure 27

1. PrLF_NPQuantOf: for NPs like "a number of books," makes "books" the head and "a number of" the modifier

2. PrLF_PPQuantOf: same but for PPs, like "with a number of books"

3. PrLF_notAnaphora: prepares to fill VP anaphora like "John thought he would go but Jim thought not _________"

4. PrLF_soAnaphora: prepares to fill VP anaphora like "Mary wondered if it was true but Jane knew so _________"

5. PrLF_toAnaphora: prepares to fill VP anaphora like "Chris wanted to go but Pat didn't want to ____"

6. PrLF_You: supplies the understood "you" in commands like "(You) please close the door"

7. PrLF_HowAbout: supplies the understood "you" in constructions like "How about (you) closing the door"

8. PrLF_We: supplies the understood "we/us" in constructions like "Let's (us) go to the movies"

9. PrLF_I: supplies the understood "I" in, for example, "(I) Thank you" or "(I) Have not yet received your letter"

10. PrLF_SubjectMods: connects "we" and "all" in, e.g., "We are all reading the book"; connects "he" and "hungry" in, e.g., "He arrived hungry"

11. PrLF_RightShift: connects "the man" and "who was my friend" in, e.g., "The man arrived who was my friend"

12. PrLF_InfclPP: prepares for correct interpretation in constructions like "a person on whom to rely"

13. PrLF_QuantifierEllipsis: having to do with the resolution of pronoun references

14. PrLF_PossessivePronHead: having to do with the resolution of pronoun references

15. PrLF_PossibleCorefsOfProns: having to do with the resolution of pronoun references

16. PrLF_VPAnaphora: identifies and fills missing arguments in all cases of VP anaphora, e.g., "Sarah likes basketball and I do too"

17. PrLF_DistCoords: distributes elements across coordinated structures, like "They washed ____ and dried the dishes"

Figure 28A - PrLF_You

If the Syntax Record

has the attribute "Infinitive"
and does not have the attribute "Subject"
or has the attribute "Verb Phrase Invert" and does not have any of the attributes "Object2," "Yes/No/Question," or "Old Subordinate Clause"
and does not meet the "There Subject Test"
and does not have the "Coordinate Constructions" attribute
and does not have any premodifiers with the node type "Auxiliary Phrase" or the attribute "Modal Verb"
and does not have any premodifiers with the lemma "let" or the node type "Adverbial Phrase,"
and does not have the node type "Abbreviated Clause," "Auxiliary Phrase," "Complement Clause," "Infinitive Clause," "Noun Relative," "Past Participle Clause," or "Relative Clause"
and does not have a parent with the node type "Past Participle Clause"
and if the head of the parent has node type "Conjunction,"
then the parent does not have a "Subject" attribute and does not have the node type "Auxiliary Phrase," "Complement Clause," "Infinitive," "Noun Relative," or "Relative Clause"
and if there is an Auxiliary Attribute on its Head
then for all its Premodifiers their Lemma must not be "neither" nor "so,"
and if it has a Do Modifier,
then it must have an Infinitive attribute and either there must not be a Modal on the First Verb attribute, or the Lemma of its First Verb must be either "dare" or "need,"
and if it has a Perfective attribute,
then its Lemma must be do,
and if it has a Verb Phrase Invert attribute,
then either there must not be a L9 attribute
or there must not be a Comma attribute and for all of its Premodifiers their node type must not be equal to "Prepositional Phrase" and for all of its Premodifiers their node type must either not be "Adverbial Phrase" or there must be a Comma attribute or the node type of their Head must be an Interjection,
and has neither "ect" nor "ect." as its Lemma,
and if its Lemma is "suffice,"
then the Lemma of its Object1 cannot be "it,"
and if its Lemma is "thank,"
then the Lemma of its Object 1 cannot be "you,"

Then

create a pronoun record for the lemma "you";
make the Subject attribute of the syntax record be a copy of the pronoun record and set the Segtype to be "NP," set the node type to be Segtype, and set the head attribute to be the pronoun record;
and set the premodifiers of the syntax record to be the value of the subject attribute plus all of the original premodifiers and set the Undersubject attribute flag.

PRINT OF DRAWINGS
AS ORIGINALLY FILED

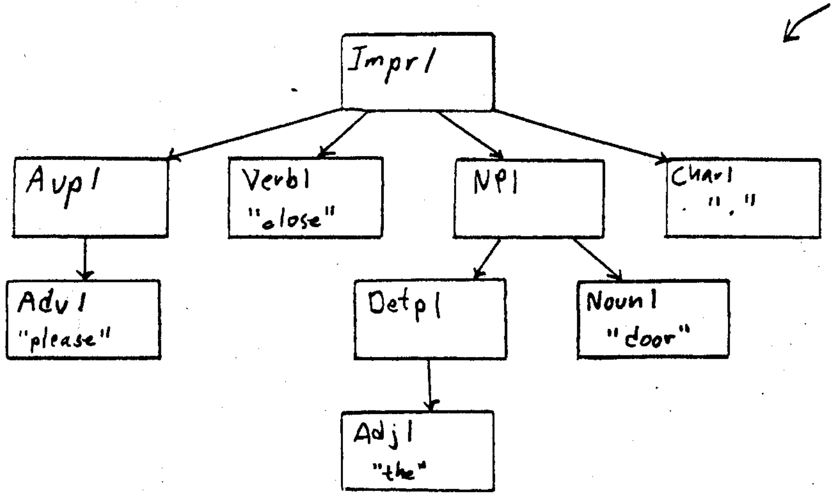
08/674610

Figure 28B

sentence represented by parse tree: "Please close the door."

syntax parse tree generated by syntactic subsystem:

2801



Rule PrLF_You

2802

2803

2804

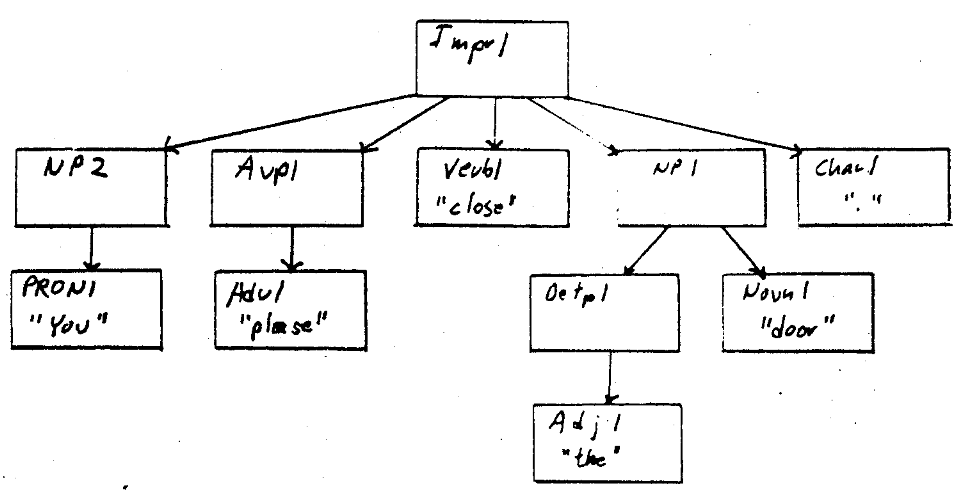


Figure 29

1. TrLF_LongDist1: locates NPs that are removed from their semantic heads and reattaches them, e.g., "Who did John say that Mary likes _(who)_?"

2. TrLF_LongDist2: performs the same kind of long-distance attachment for AJPs, INFCLs, PPs, PRPRTCLs, PTPRTCLs, SUBCLs

3. TrLF_PhrasalVerb: defines semantic objects of certain verbs when they appear hidden inside PPs: "his hat" is really the semantic object of "took off" in "He took off his hat"

4. TrLF_ControlwNP: e.g., in "Chris told Pat what to eat," "Pat" is really the subject of "eat" and "what" is its object

5. TrLF_ControlwAJP: e.g., in "I find this difficult to believe," "this" is really the object of "believe"

6. TrLF_ForInfcl: used in "for-to" constructions, e.g., in "For Mary to talk to John is easy," "Mary is really the subject of "talk"

7. TrLF_ForInfclCoords: used in "for-to" constructions that have coordinated PPs

8. TrLF_MoveProp: given our strategy for attachment, it is sometimes necessary to move clauses from a lower to a higher level so that the proper argument structure can be assigned

9. TrLF_ControlatVP: e.g., in "Farmers grow food by using salt water," "farmers" is really the subject of "use salt water."

10. TrLF_PropsAsArgs: some clauses (propositions) can be arguments, e.g., in "Has he to answer the letter?" the object of "has" is "to answer the letter"

11. TrLF_Extraposition: e.g., in "It makes me happy to meet you," the real subject of "makes" is "to meet you" -- "it" is an empty word and must drop out

12. TrLF_FillCoords: fills in missing arguments in coordinated structures

13. TrLF_RedefineSubject: e.g., in "What is John's address?" we interpret "John's address" as the logical subject even though it is not in canonical subject postition

Figure 30A - TrLF MoveProp

If the Syntax Record

has either a node type of Abbreviated Clause, Infinitive Clause, Present Participle Clause, Past Participle Clause

or if it has a Gerund attribute and an Object of a Prepositional Phrase and

if it has Premodifiers,

then the node type of all Premodifiers must be either Auxilary Phrase, Adverbial Phrase, or Prepositional Phrase,

and the node type of the Head attribute of the Parent is not "verb"

and this syntax record is the last of the post modifiers of its parent

and this syntax record is not in the coordinates attribute of its parent

and among the ancestors of the parent there is a record whose node type of the Head is "Verb" but none of those ancestors can have a Coordinates attribute (this record will later be referred to as "same ancestor")

and there should be no For To Prepositional Phrase attribute on the parent,

and if the node type equals Infinitive Clause,

then there must be either no WH attribute on PP obj of the parent or the syntax record is not equal to the Nominal Relative of the parent,

and if the node type is either Present Participle or Past Participle,

then its Parent does not have an Object of a Prepositional Phrase,

and if the node type is a Present Participle Clause,

then there must be an 'ING' Complement on the same ancestor

and if the node type is a Past Participle Clause,

then there must be a V8 (code from Longman's dictionary) attribute on the same ancestor and if there is an X1 attribute on the syntax record then there must not be an Object 1

and there is no B3 attribute on its parent,

and this syntax record must follow the head of the same ancestor or there is a passive attribute on the same ancestor

and if the Lemma of the Parent is 'certain'

then the node type of the parent must not be an Adjective Phrase

and if the Lemma of the Preposition is either "as" or "of,"

then there must be a To Noun attribute of its Parent

and if the Lemma of the same ancestor is either "be" or "become"

then either the node type of the Parent must be an Adjective Phrase

or there must be a WH attribute on the Parent

or there must be both a To Noun attribute on parent and no There Subject Test on the same ancestor

or the Lemma of the Parent must be one of the following: "delight," "horror," "joy," "pleasure," "riot," "shame," "surprise," "terror,"

Figure 30B - TrLF MoveProp

Then

the syntax record whose attributes will be changed is the same ancestor syntax record (see above);

if the Parent of the syntax record has the Subject attribute and the Parent of syntax record also has the Object attribute,

then delete the object attribute from the ancestor;

if the Parent of the syntax record has the Subject attribute and the Parent of the Syntax Record does not also have the Object 1 attribute,

then set the Subject attribute of same ancestor to be the syntax record;

if the same ancestor has

the DI (Longman code) attribute and there is an Object Complement attribute and no Indirect Object attribute and there is a To Infinitive on the syntax record and the Parent of syntax record is the Object

and there is no WH attribute on the Parent of Syntax Record

and either there is an Animate attribute on Parent of syntax record

or there is a Case attribute on Parent of Syntax Record and the Lemma of the Parent of the syntax record is not "it"

or there is a Human attribute on the Parent of Syntax Record

or there is a Proper Name attribute on Parent of syntax record,

then make the Indirect Object Attribute on same ancestor equal to that of the Parent of syntax record;

if there is an To Infinitive attribute on the syntax record and no Passive attribute on same ancestor,

then make the Predicate Complement attribute equal to the syntax record;

if the Parent of syntax record is in the Propositions attribute of same ancestor,

then take that Propositions list and replace the Parent of the syntax record with the syntax record itself in the propositions list;

delete the Infinitive attribute of the Parent of the syntax record;

delete the Alternatives attribute on the syntax record;

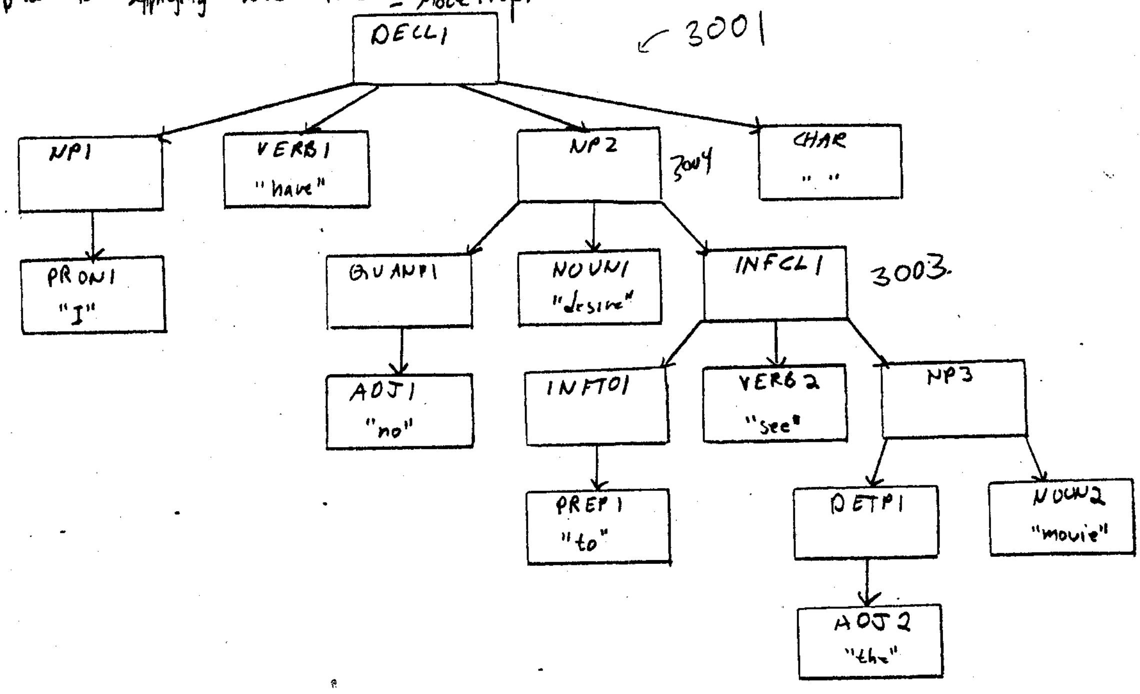
reattach the syntax record to the same ancestor.

PRINT OF DRAWINGS
AS ORIGINALI LED

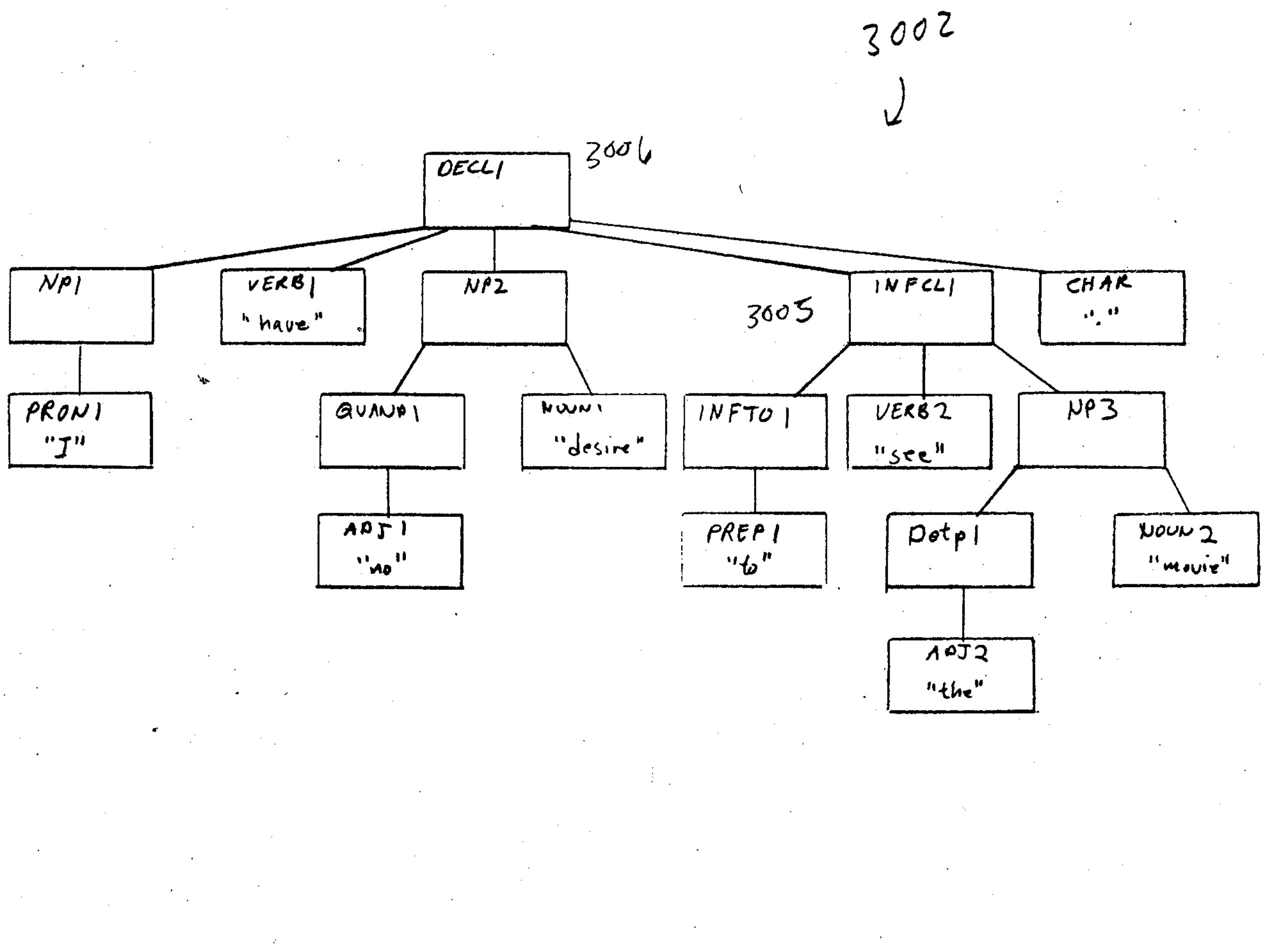
08/674610

Figure 30 C

sentence represented by parse tree: "I have no desire to see the movie."
syntax parse tree prior to applying rule TrLF_MoveProp:



Rule TrLF_MoveProp:



PRINT OF DRAWINGS
AS ORIGINAL FILED

Fig. 1

08/674610

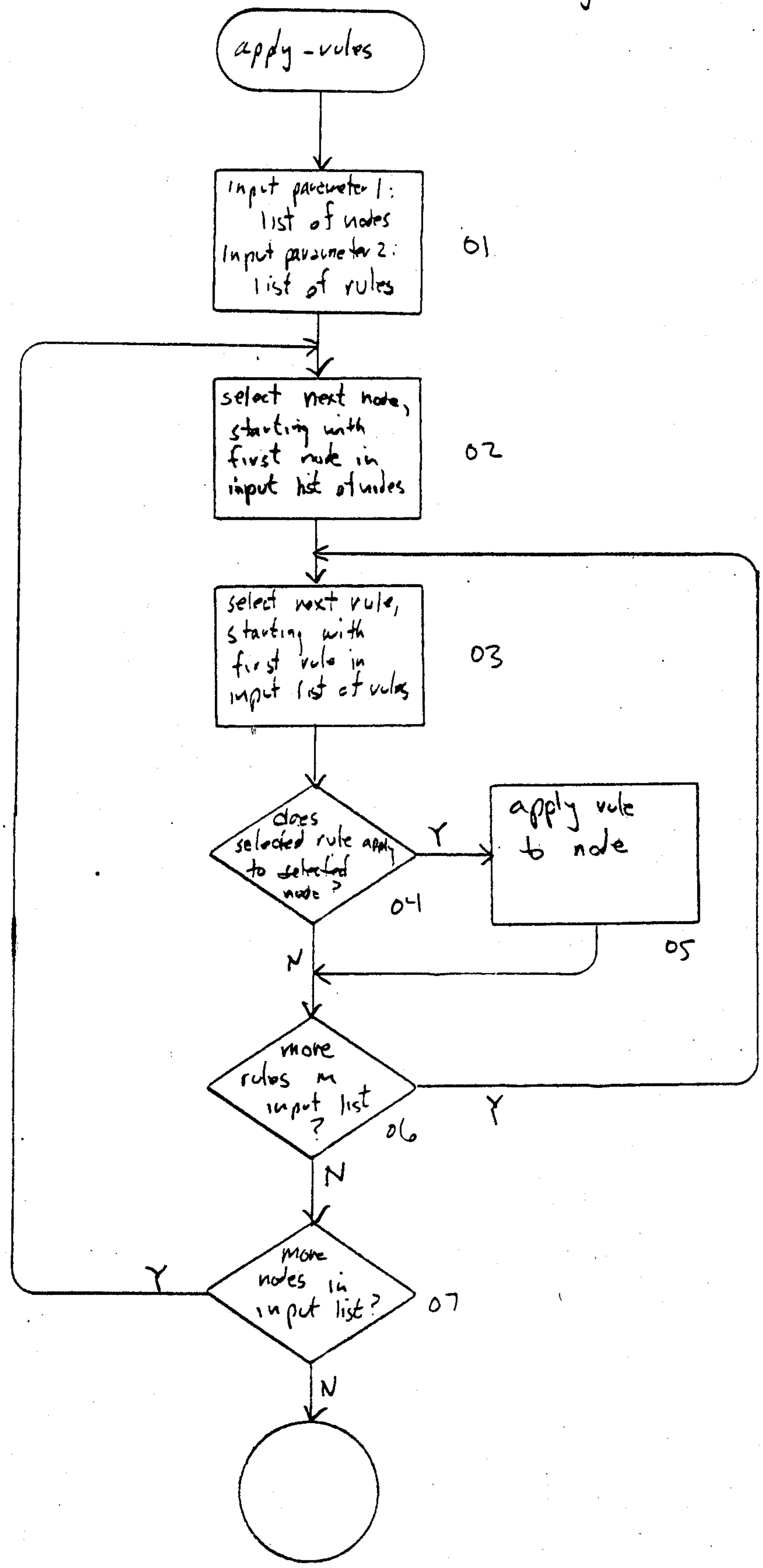


Figure 32

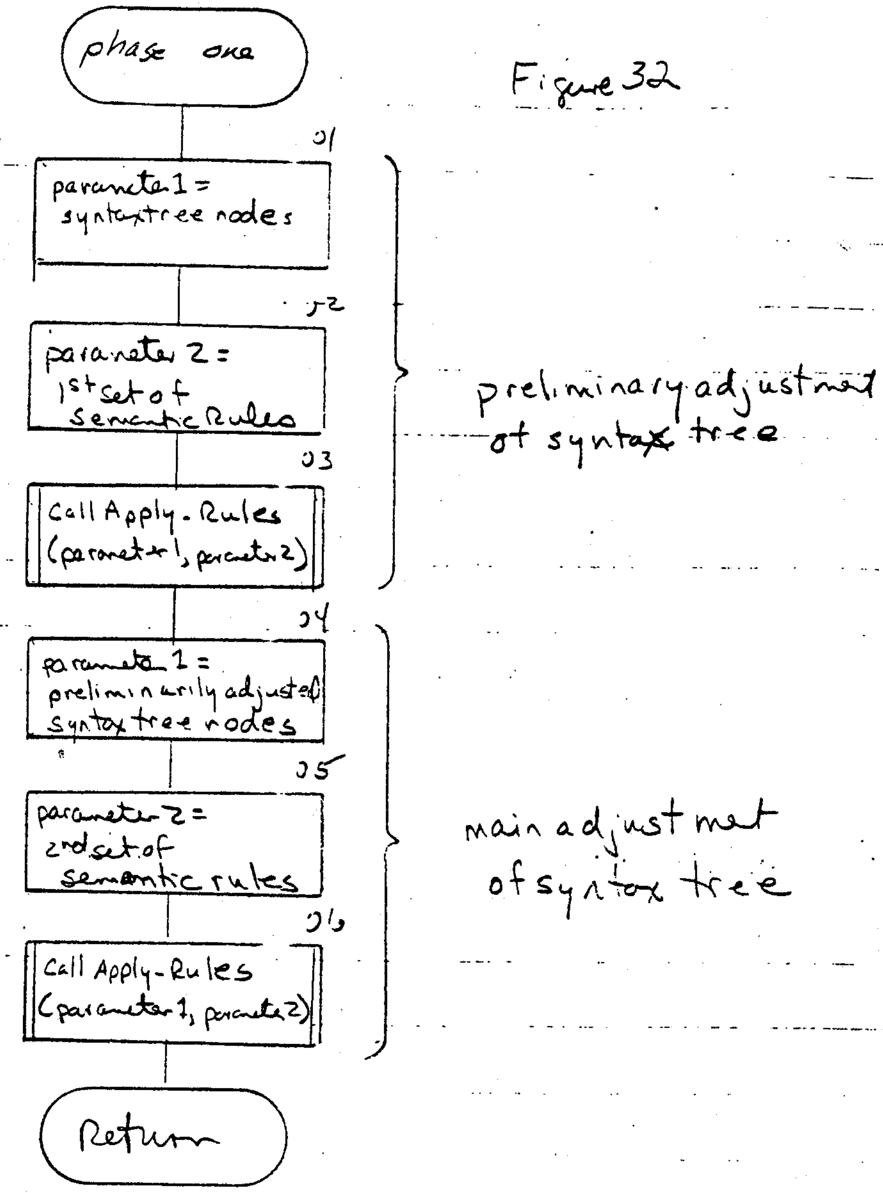

phase one
01
parameter 1 =
syntax tree nodes
02
parameter 2 =
1st set of
Semantic Rules
03
Call Apply-Rules
(parameter 1, parameter 2)
04
parameter 1 =
preliminarily adjusted
syntax tree nodes
05
parameter 2 =
2nd set of
semantic rules
06
Call Apply-Rules
(parameter 1, parameter 2)
Return
preliminary adjustment
of syntax tree
main adjustment
of syntax tree

Figure 33

1. SynToSem1: creates semantic nodes and a basic semantic graph in es

2. SynToSem2: creates the top-level semantic node and graph for fitted parses

3. SynToSem3: creates semantic nodes for a special subclass of elements in fitted parses

Figure 34 A - Rule SynToSem1

If

the Syntax Record
has a Head and
there is no Subordinate Conjunction and
there is no Correlative and
there is no "It subject" and
there is no "There subject" and
there is no Ancestor of the Head for which it is true that that node
is the Emphatic of its Parent and is not a fraction and the head node
is not a verb and
if the segment is the Relative Pronoun of its Parent,
then there must not be a Nominal Relative on the Object of its Parent
and for all of its Parents last records there must not be a VPDone attribute and
if the lemma equals 'that'
then there must not be an Extra Position on the Parent of the Parent and
the node type is not "Auxiliary Phrase," "To Infinitive," "Determiner Phrase,"
or "Tag" or
there is a Possessive attribute or
there is an EVR attribute or
the Lemma equals "other" or
there are Coordinates and for all of those coordinates there is either a
Possessive attribute or an EVR attribute or the lemma is "other" and
if the node type is "Adverb Phrase"
then if the node type of Parent equals Prepositional Phrase
then the segment must not be the first of the Premodifiers of its Parent
and
either the Lemma must not be equal to 'well' or there must not be any Degree
attribute or there must not be any Weak Obligation on the Parent and
If the node type of the Head is a Conjunction or a Preposition,
then the segment node must not be a Conjunction of the Parent and the
segment node must not be a Preposition of the Parent and
If the node type is a Conjunctive Phrase
then there must not be any Coordinates of the Parent or there must not be a
Coordinate Conjunction attribute and
If the node type is a Quantifier Phrase,
then the Lemma of the Head must not be "no" and
If the word could have been an Interjection
then the node type must not be an Adverb Phrase or
there must be Premodifiers or
there must be no comma or
the segment must be the Post Adverbial of the Parent or
the number of Post Modifiers must be greater than one and
If there is an Intensifier attribute
then either the node type of Head of Parent is a "verb" or
the node type of Parent equals "fitted" or
there is an Adverbial Phrase attribute or
there is a WH marker and a Nominal Relative on the Parent and
If there is a Preposition attribute,
then there must be an Object of the Prepositional Phrase or
there is a Particle attribute on the Parent or
the word also could have been an Adverb and

Figure 34 B - Rule SynToSem1

If the Lemma is "also", "so," or "too,"
then there must not be a VPDone attribute on the Parent and
If the Lemma is "as" or "than"
then there must not be a Comparative on the Parent and
If the Lemma equals "for"
then there must not be a "for to" Preposition on the Parent and
If the Lemma equals "it"
then if there is a Topic Clause on the Parent
then the segment must be equal to the Subject of the Parent or
the segment must be equal to the Object of the Parent and
If the Lemma equals "it"
then the segment must not be in the Premodifiers of the Parent or
If there is an Extra Position on the Predicate Adjective of the Parent
then there must not be a Right Shift attribute on the Parent and
if there is a WH Question attribute on the Parent
then there is no "To Infinitive" attribute on the Predicate Compliment of the Parent and it's not the case that for any of the Post Modifiers of the Parent that there is a "For to" prepositional phrase on the first of the Premodifiers and
If the Lemma equals "let"
then the node type is not equal to "Adverb Phrase" and
If the Lemma equals "not"
then there must be a Coordinate Conjunction on the Parent and
If the Lemma equals "there"
then there must not be any Skipover attribute and
either there must not be any "Yes No" question on the parent or
there must not be a Copulative on the Parent or
there must be a T1 attribute on the Parent or
the first token integer must be greater than the first token integer of the Subject of the Parent and
If the Lemma is "whether" or "whether or not"
then the node type of the Nominative Relative must not be an "Infinitive Clause" and
the Lemma must not be "etc," "etc.," "the," "hm," "mm," "uh," or "um"

THEN

(If syntax node was kept, then create a corresponding semantic node.)
If the node type of the syntax node is a Noun Phrase and
there are Bases on the syntax node and
there is a Subject or an Object on the syntax node,
then make the Predicate equal to the Lemma of the first Basis of the syntax node
Else if there is a Proper Noun attribute on the syntax node and
if there is a dictionary entry for that word,
then make the Predicate equal to that dictionary entry
Else set the Predicate equal to the Lemma of the syntax node
If the word could have been a Verb and has a Present Participle attribute and
if for any of the Premodifiers of the syntax node there is a Possessive or
if the Lemma of the Preposition of the first of the Postmodifiers of the syntax node is "by," "for," "of," or "to"

Figure 34 C - Rule SynToSem1

then make the Predicate equal to the Lemma of the Verb entry of the Part of Speech
Record
Copy the appropriate fields from syntax node to the semantic node.
Go through each of the Premodifiers of syntax record and examine each Premodifier
For each record of Premodifiers of the syntax record
if there is a semantic node on the record and
if the semantic node of the record is not in the temporary modifiers attributes of this semantic record and there is no Skipover attribute on the record and the record is not equal to the Preposition of the Parent of the record and the record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record, or Coordinate Subordinate Clauses
then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record
For each record of the Postmodifiers of the syntax record
if there is a semantic node on record and
if the semantic node of record is not in the Temporary Modifiers attributes of this semantic record and there is no Skipover attribute on record and record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record or Coordinate Subordinate Clauses
then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record
If there are Coordinates of the syntax record and no Coordinates of the Prepositional Phrase on that syntax record and no Coordinate Subordinate Clauses
then
for each of the Coordinates of syntax record
if there is a Semantic node on record,
then add that Semantic node to Coordinates attribute on this new Semantic record.

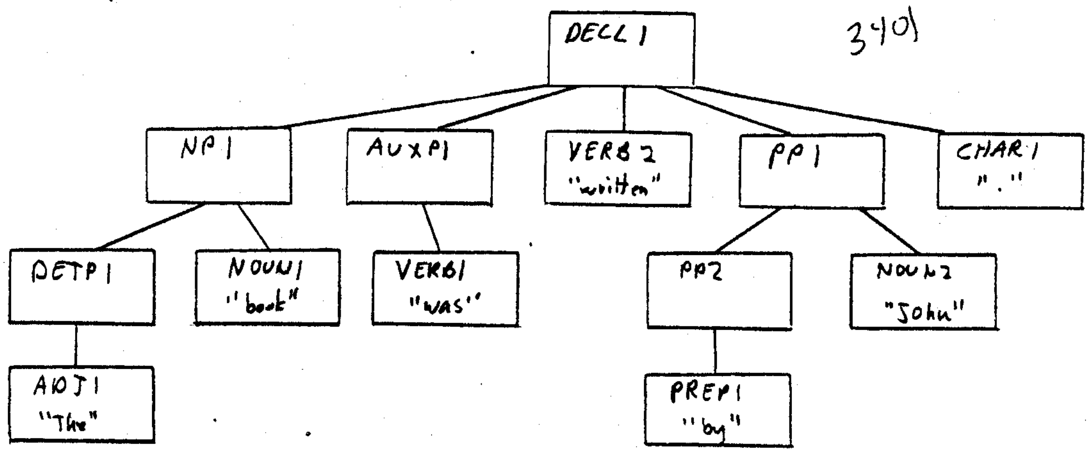
PRINT OF DRAWINGS
AS ORIGINAL FILED

08/674610

Figure 34D

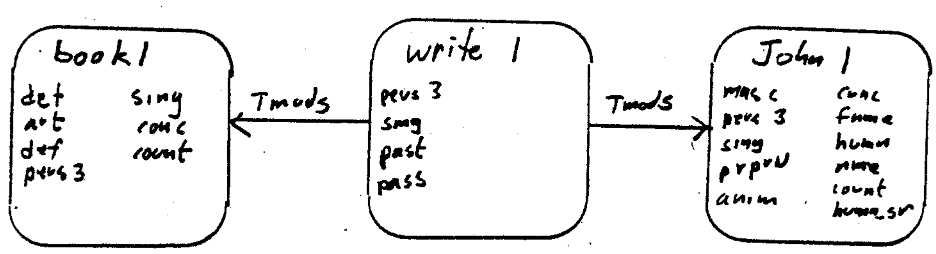
Sentence represented by syntax parse tree: "The book was written by John."
syntax tree prior to application of rule SynToSem1:

3401



Rule SynToSem1:

3402



PRINT OF DRAWINGS
AS ORIGINAL FILED

08/674610

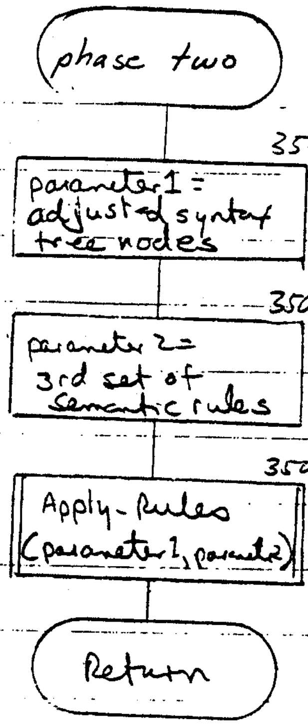


Figure 35

Figure 36

1. LF_Dsub1: creates the Dsub (deep subject) label for subjects of clauses in the active voice

2. LF_Dsub2: for passive-voice clauses, if there is a "by"-PP, identifies this PP as the Dsub of the action

3. LF_Dobj1: creates the Dobj (deep object) label for, e.g., direct objects of clauses in the active voice

4. LF_Dobj2: for passive clauses, identifies the syntactic subject as the deep object of the action

5. LF_Dobj3: for clauses like "The door opened," identifies "the door" as the logical object of the action

6. LF_Dobj4: for constructions like "the nomination of the candidate," identifies "the candidate" as the logical object of an action of nominating

7. LF_Dind1: creates the Dind (deep indirect object) label for, e.g., "Mary" in "John gave Mary the book"

8. LF_Dind2: identifies the deep indirect object ("Mary") in paraphrases like "John gave the book to Mary"

Figure 37

9. LF_Dind3: chooses the right deep indirect object in trickier constructions like "The book was given her"; "She was given the book"

10. LF_Dnom: creates the Dnom (deep nominative) label for predicate nominatives, e.g., "our friends" in "They are our friends"

11. LF_Dcmp1: identifies the complement ("president"; "italic") in, e.g., "elect Tom president"; "make the word italic"

12. LF_Dcmp2: identifies the complement in tricker constructions, e.g., in "He gave Tom a place to call his own," "his own" is the Dcmp of "call"

13. LF_Dadj: creates the Dadj label for predicate adjectives, e.g, "blue" in "The sky is blue"

14. LF_CausBy: creates a causative relation where appropriate, e.g., "why" in "Why did you say that?"

15. LF_LocAt: creates a locative relation where appropriate, e.g., "where" in "Where did you find that?"

16. LF_TmeAt: creates a temporal relation where appropriate, e.g., "what day" in "What day did you read that?"

17. LF_Manr: creates a manner relation where appropriate, e.g., "how" in "How did you do that?"

Figure 38

18. LF_Ptcl: creates a Ptcl node to refer to particles in phrasal verb constructions

19. LF_PrpCnjs: creates temporary relations for PPs and subordinate clauses by naming these elations with the word that is the preposition or conjunction

20. LF_PrpCoord: handles cases of coordinated PPs or subordinate clauses

21. LF_Props: lists remaining clausal adjuncts for any given node

22. LF_Ops: identifies logical operators in noun phrases, e.g., "all" in "all my children"

23. LF_Nadj: lists remaining adjectives that premodify nouns

24. LF_Mods: lists remaining non-clausal modifiers for any given node

Figure 39A - Rule LF_Dobj2

If the Semantic Record

doesn't already have a Deep Object,

and has a Passive attribute,

and has a Subject on its syntactic record (SynNode), and this Subject (which is a syntactic record) has a SemNode attribute (i.e., it has a corresponding semantic record)

and there are no Coordinates

and if there is a Predicate Complement attribute on its synactic record, then the node type is not "COMPCL" (i.e., it is not a complement clause, as in: "some people were convinced <u>that he had written a book</u>"

and if the SynNode record has either a D5, D6, ObjC, or Psych feature[2]

then either the Object of the SynNode is not a noun phrase,

or the SynNode has an X1[3] feature (as in: He was named Arles")

or the Object of the SynNode has an Animate feature

or there is a Case feature on the Object of the SynNode and its Lemma is not "it"

Then,

give the Semantic record a Dobj attribute with, as its value, the semantic record corresponding to the Subject on the syntactic record

and, remove what is now the value of Dobj attribute from the list of Tmods

[2]D5, and D6 are features from Longman's Dictionary of Contemporary English; ObjC is verb subcategory for verbs which show object control (e.g., I want Harry to wash the car) and Psych is a verb subcategory for verbs like "scare" "excite".

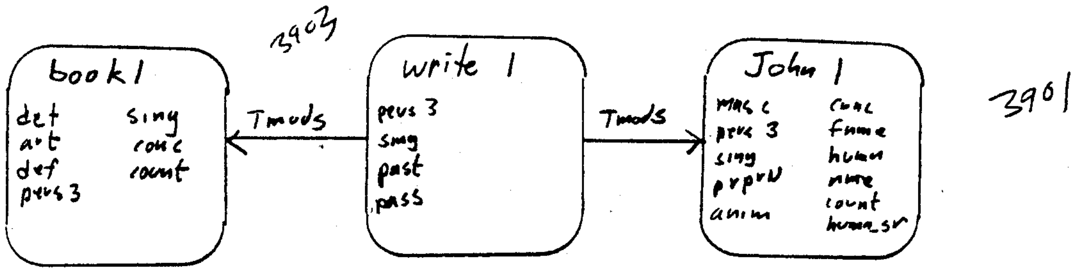
[3] X1 is a feature from Longman's Dictionary of Contemporary English.

08/674610

Figure 39B

sentence represented by the logical form: "The book was written by John."

logical form prior to application of rule LF_Dobj2:



LF_Dobj2:

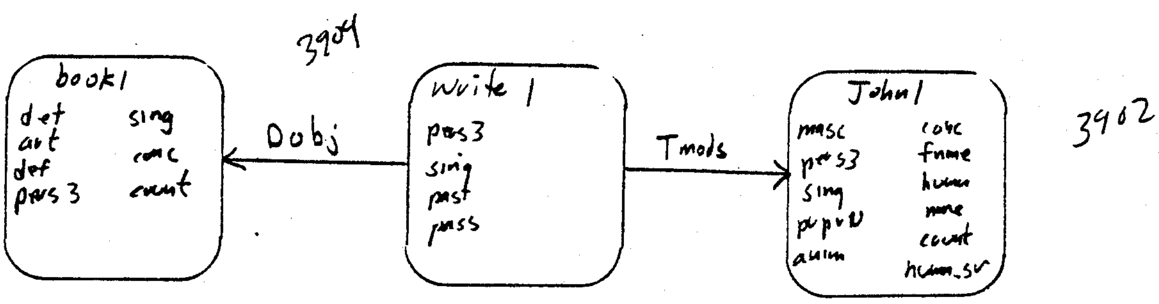


Figure 40

1. PsLF_RelPro: identifies proper referents for relative pronouns, e.g., "who" refers to "the man" in "the man who came to dinner"

2. PsLF_ReciprocalAnaphora: handles reciprocal pronouns like "each other" and "one another"

3. PsLF_ReflexiveAnaphora: handles reflexive pronouns like "myself, yourself, him/herself," etc.

4. PsLF_PronAnaphora: identifies possible NP referents for most pronouns

5. PsLF_ProtoAnaphora: handles special cases of pronouns which can agree with just about any NP

6. PsLF_NumberEllipsis: handles reference for number words, e.g., "A bird in the hand is worth two (birds) in the bush"

7. PsLF_FillInHead: adds "DUMMY" as a head word in special cases of unclear referents

8. PsLF_NumberCritique: takes note of pronouns that disagree in number with their referents

9. PsLF_FillDsub: fills in "x" as a placeholder for the deep subject in cases where that is missing, e.g., in passives like "The door was opened"

10. PsLF_UnifyProns: if two pronoun nodes refer to the same referent, this rule unifies them

11. PsLF_UnifyCopies1: unifies some nodes that should be identical

12. PsLF_UnifyCopies2: unifies other nodes that should be identical

13. PsLF_RaiseModality: deletes some verbs when they serve only an aspectual purpose, e.g., in "We used to go there," "used to" is deleted from the graph

14. PsLF_RaisePcs: makes fitted parses easier to read

Figure 41A - Rule PsLF_PronAnaphora

If the Semantic Record

has a Pers3 attribute, i.e., it is not either first (e.g., I or we) or second person (e.g., you)

and the node type of the head of its syntactic record is either "PRON" (pronoun) or the node type of the head of its syntactic record is "ADJ" (adjective) and it has a possessive attribute

and is not Reflexive

and none of the premodifiers of the Parent of its syntactic node has the Lemma "own"

and the Pred of this semantic record is not "each other" or "one another"

and does not have NonRef attribute (NonRef is an attribute set on words that cannot have a reference, such as true numbers, as in: One plus one is two.

and does not have a Negation attribute

and if it has an Indefinite attribute, then there must also be a Definite attribute

and is not a Wh- word (it does not have a Wh attribute)

and is not a Relative

and is not a Distal (Distl) or a Proxal (Proxl) determiner (e.g., "this" "that")

Then

add a FindRef attribute to the semantic record

for each of the records in the list of possible referents; [1]

if

the possible referent has a corresponding semantic record

and the possible referent is not the same as this record (i.e., the antecedent of a noun phrase cannot be the noun phrase itself)

and if the head of both the possible referent and of this record's SynNode are pronouns (i.e., have the node type "PRON" as their head), then the possible referent must precede this record (no forwards reference to a pronoun; an example of forwards (cataphoric) reference is: with his hat on, the teacher left the room, where "his" refers forwards to "teacher"

and if the possible referent is the ancestor of the syntactic record of this record, then that ancestor must have a Prp attribute (i.e., must have a postmodifying Prepositional phrase), and its preposition must be either "in", "to", "for", or "by"

and there is no Time or Space feature on the possible referent

and this record and the possible referent agree in number

and this record and the possible referent agree in gender

and if the Lemma of the SynNode is "they" and the possible referent can be a Mass noun (i.e., the possible referent has a Mass feature),

then the possible referent must also be a Count noun (i.e., it must also have a Count feature).

and if the Lemma of the SynNode is "they" and the possible referent has a Sing feature (can be Singular), and the possible referent does not have a Plur feature (i.e., it cannot be Plural),

then the possible referent is either a Count noun, or the possible referent is a Coordinated noun phrase, or it has a Universal feature, or the possible referent is indefinite and has no possessive, or the possible referent has a Proxal feature,

[1] this list is created in a PrLF rule, so, after syntactic processing but before most logical form processing (it is a list of syntactic records). This is a list of all the words in the sentence which can be referred to, i.e. most of the nouns and pronouns in the sentence

Figure 41B - Rule PsLF_PronAnaphora

and if there is an ancestor of the possible referent that has a Coords attribute (i.e., has coordinate constituents) (but before there is an ancestor with a Subject attribute) then this ancestor is the same as the ancestor of this record that has a Coords attribute (but before there is an ancestor with a Subject attribute)

then if this record is a possessive (e.g., "his" in "John saw his son")

add the possible referent to the list of possible referents (the value of the Refs attribute) if:

the possible referent is a genitive

and node type of the head of the possible referent is not a Noun

and the possible referent precedes this record (i.e., the semantic record being processed in this rule

or if:

the possible referent is not the first of this record's Parents

and the first of the Parents of the possible referent is not the first of this record's Parents

and if the possible referent follows this record and if any of the possible referent's ancestors have Coordinate constituents, then there should be no ancestor of this record for which the Parent has Coordinate constituents and for which the Parent is the same as the ancestor of the possible referent that has Coordinate constituents (but before there is ancestor whose node type is "NP")

or else if the node type of Parent of this record's syntactic record is "TAG" (i.e., if the pronoun is in a tag question)

add the possible referent to the list of possible referents (the value of the Refs attribute) if:

the possible referent is the Subject of the Parent of the Parent of this record (e.g., "they" refers to "someone" in : Someone painted in here, didn't they?)

or else:

if

this record is a prepositional phrase

and this record precedes the Subject of this record's Parent

and the possible referent is the Subject of this record's Parent

then add the possible referent to the list of possible referents (the value of the Refs attribute);

else if

this record is not possessive

and this record precedes the possible referent

and node type of the head of the possible referent is "NOUN" and is not a Dummy noun (i.e., one that cannot be a possible referent)

and if this record is not one of possible referent's ancestors

and if it is not the case that there is an ancestor of this record that has Coordinate constituents and the Lemma of that ancestor is "but" and that ancestor is also an ancestor of this record which has Coordinate constituents

then add the possible referent to the list of possible referents (the value of the Refs attribute)

Figure 41C - Rule PsLF_PronAnaphora

else if

the possible referent is a Prepositional Phrase
and the Parent of the possible referent is not the Parent of this record's syntactic record
and if the Parent of the possible referent is an Adjective Phrase, then the Parent of the possible referent precedes this record

then add the possible referent to the list of possible referents (the value of the Refs attribute)

else if

there is no ancestor of the possible referent for which the Lemma is "be" (but before there is an ancestor with a Subject) that is the same as the ancestor of this record for which the Lemma is "be" (but before there is an ancestor with a Subject)
and none of the Parents on the semantic record of the possible referent is the same as the possible referent
and if this record precedes the possible referent, then the Head of the possible referent is not either a Noun or an Adjective

then add the possible referent to the list of possible referents (the value of the Refs attribute)

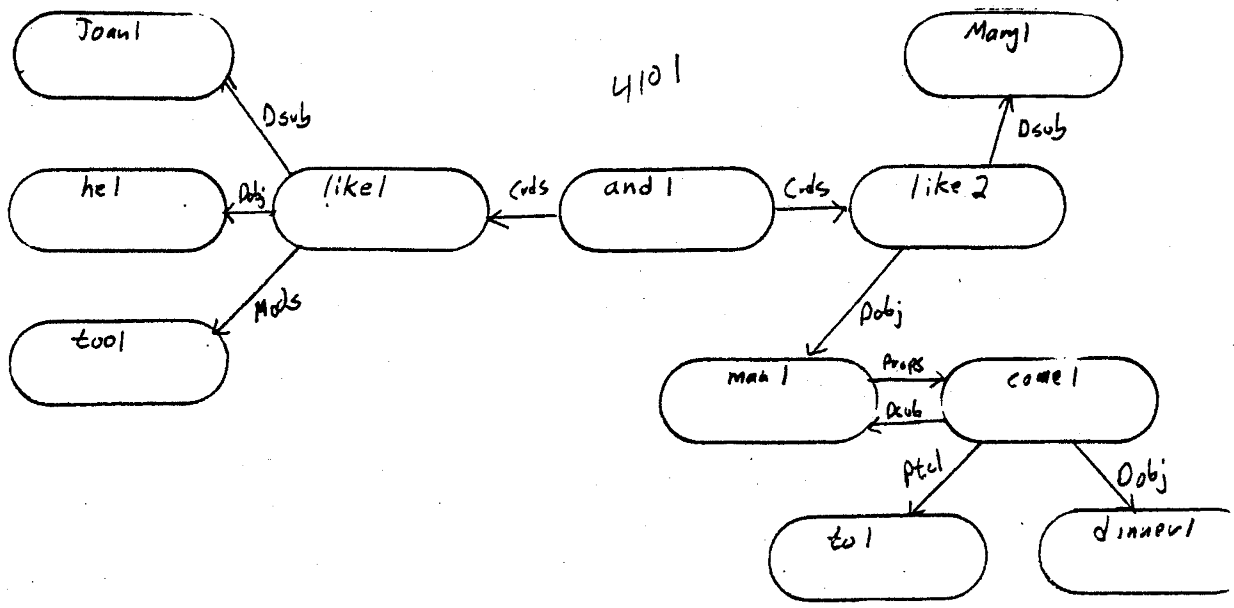
if the possible referent was added to the list of possible referents (the value of the Refs attribute)
then add of RefOf attribute to the possible referent and add this record to that list (provide cross pointers: this record gets a Ref attribute pointing to possible referents, and the possible referents each get a RefOf attribute, pointing back to this record.

PRINT OF DRAWINGS
AS ORIGINALLY FILED

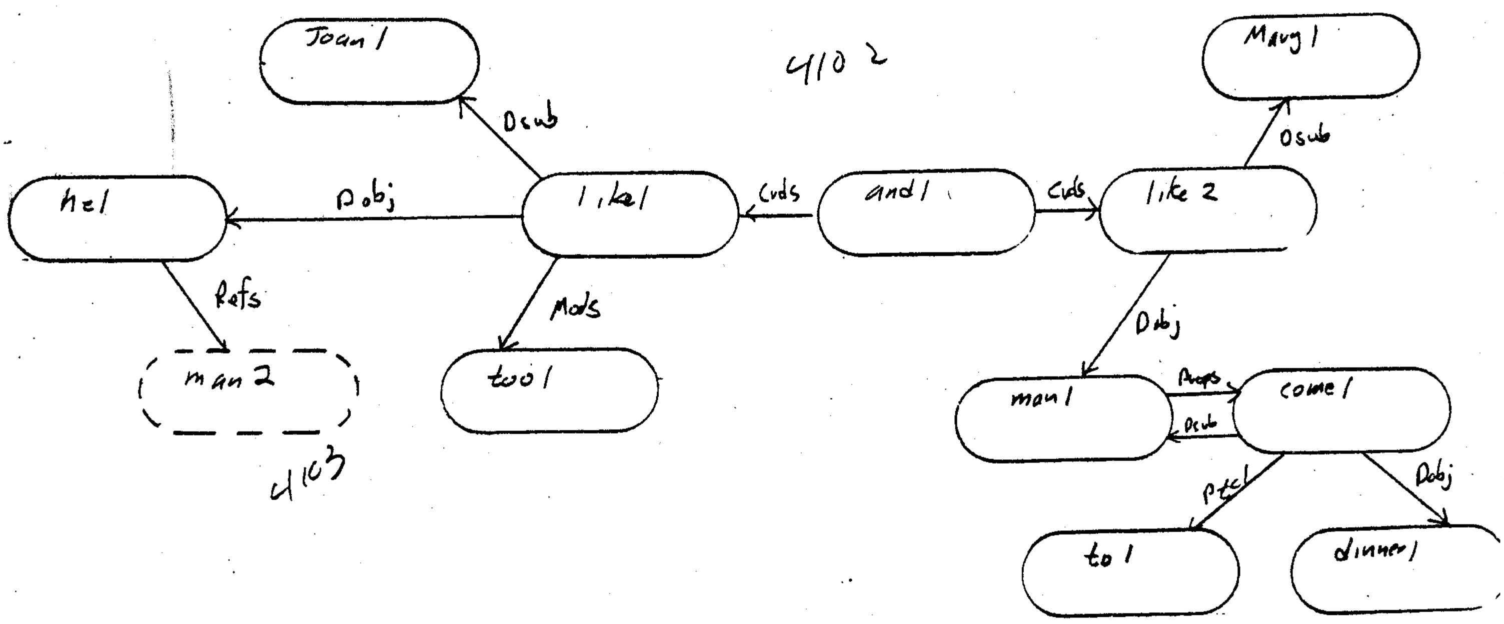
08/674610

Figure 41D

sentence represented by logical form: "Mary likes the man who came to dinner, and Joan likes him too."
logical form prior to application of rule PsLF_Pron Anaphora:

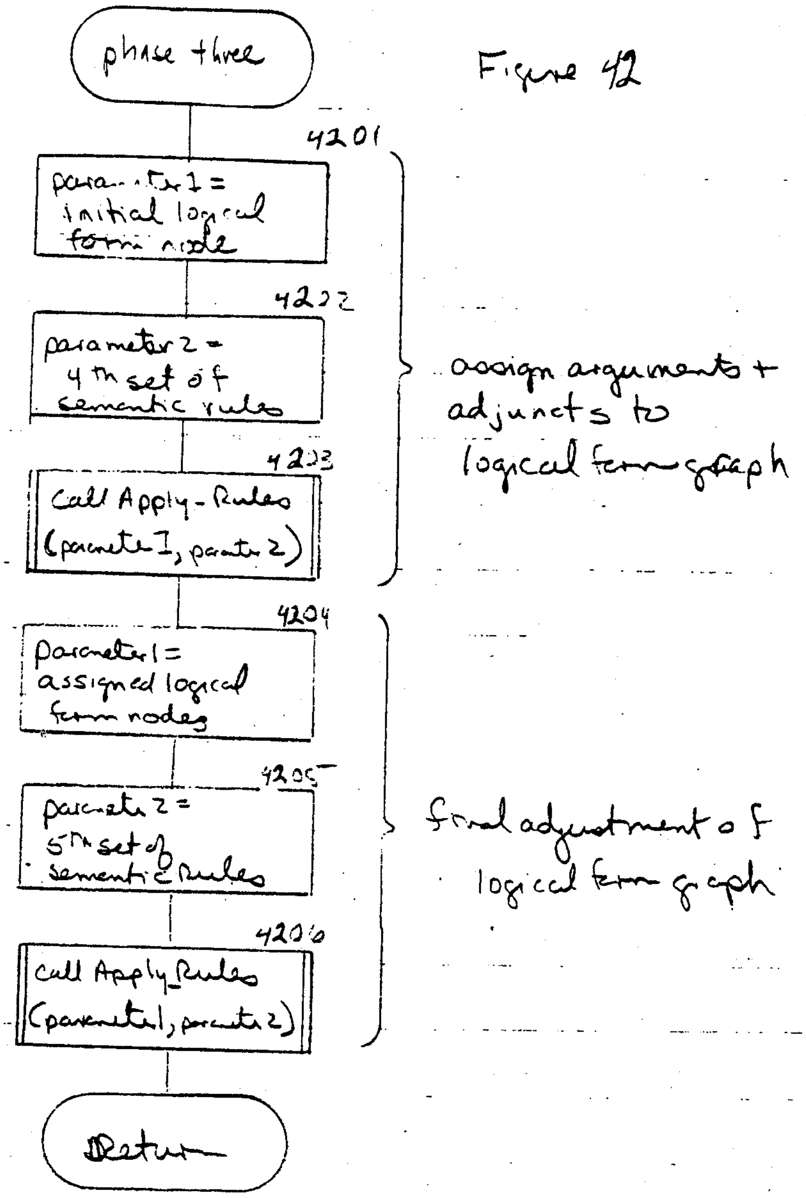


Rule PsLF_Pron Anaphora:



08/674610

Figure 42



PRINT OF DRAWINGS
AS ORIGINALLY FILED

08/674610

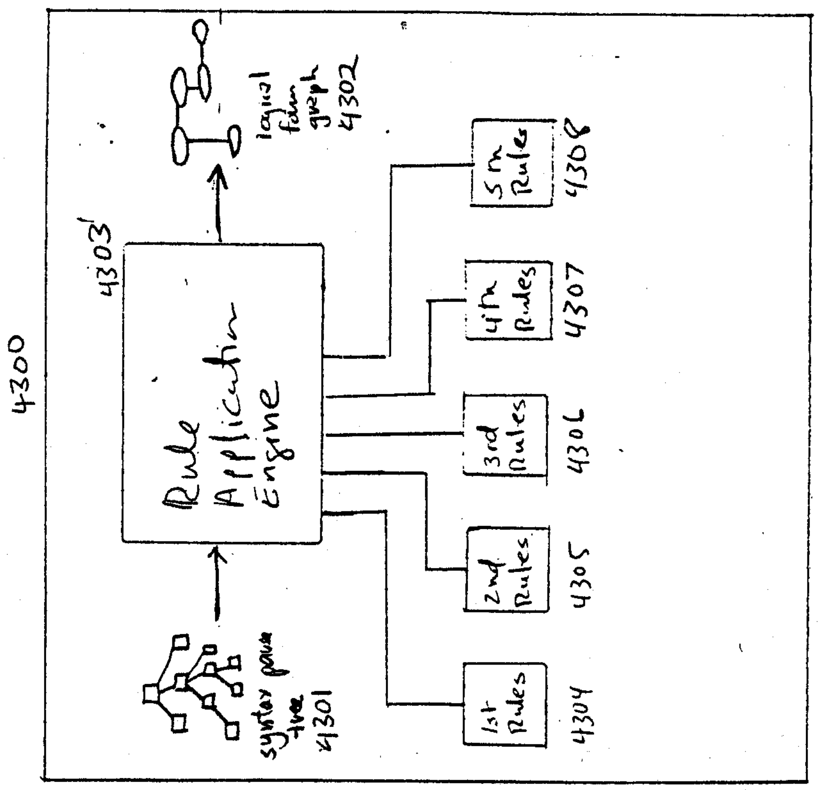


Figure 43

Figure 44

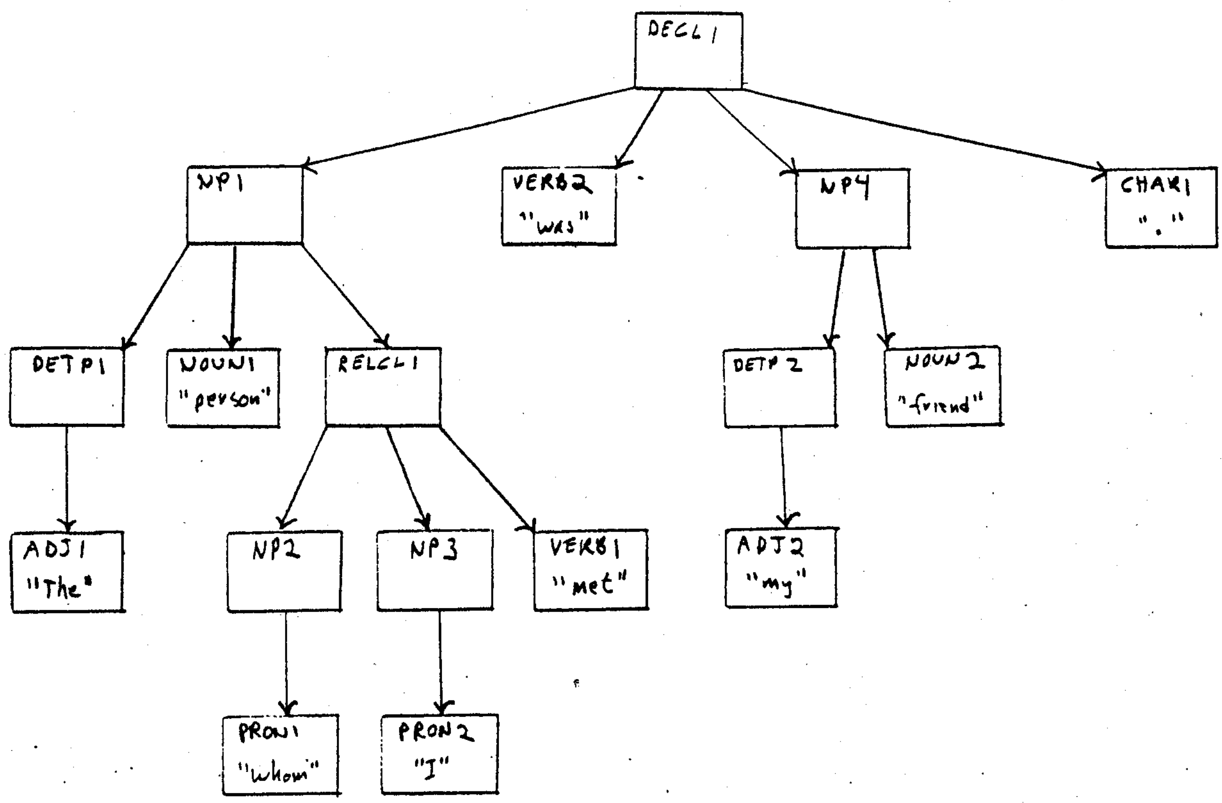
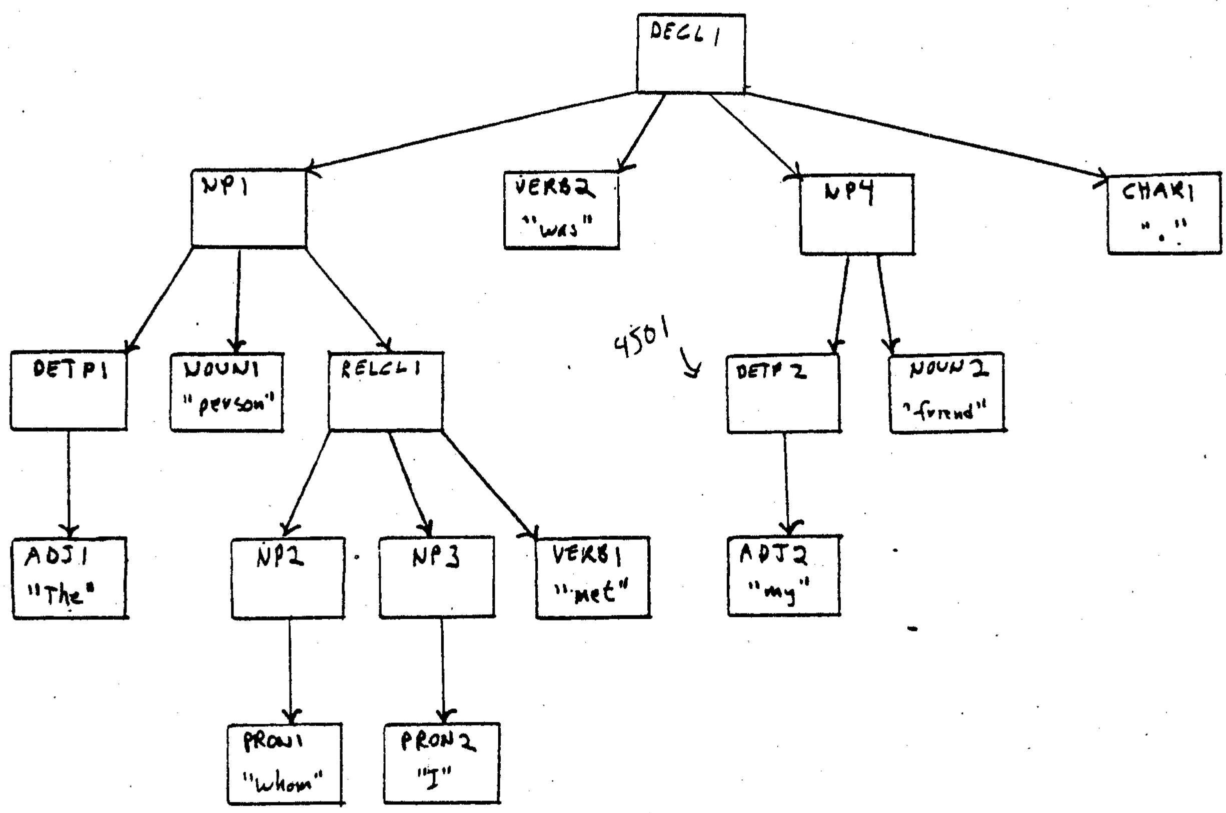

DECL1
NP1
VERB2
"was"
NP4
CHAR1
"."
DETP1
NOUN1
"person"
RELCL1
DETP2
NOUN2
"friend"
ADJ1
"The"
NP2
NP3
VERB1
"met"
ADJ2
"my"
PRON1
"whom"
PRON2
"I"

Figure 45



Rule: SynToSem1 produces logical form graph node from DETP2 ("my")

⇓

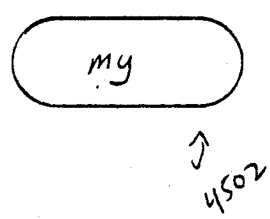
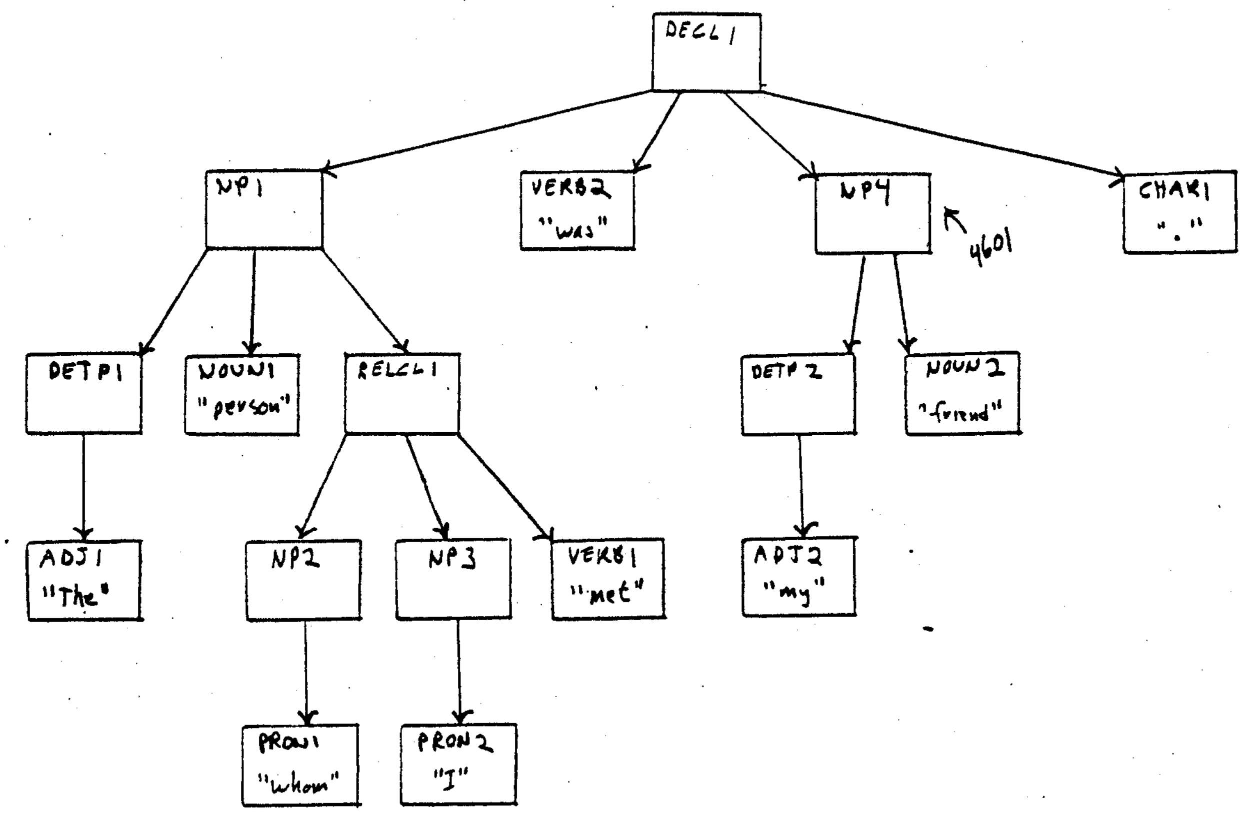


 Figure 46



Rule: SynToSem1 produces logical form graph node "friend" from NP4 ("my friend")

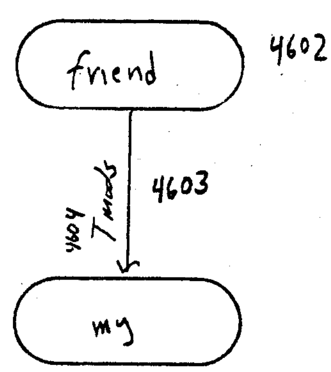
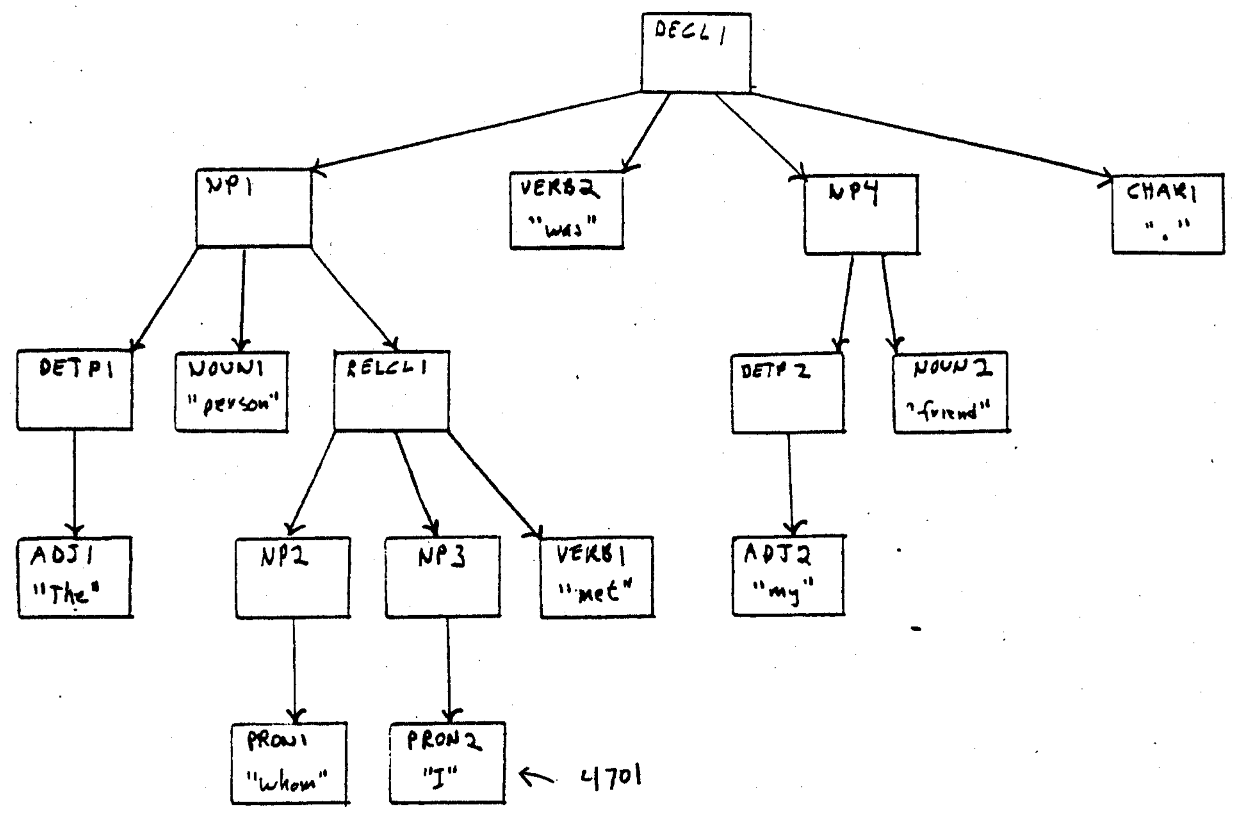
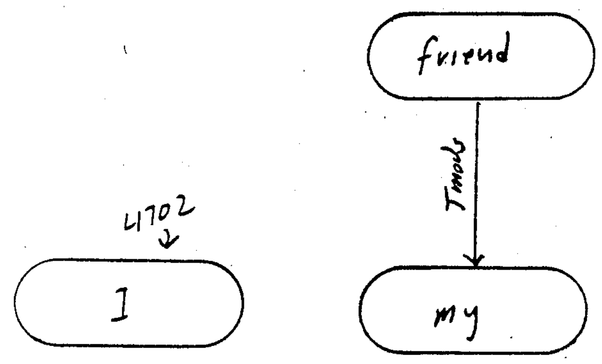
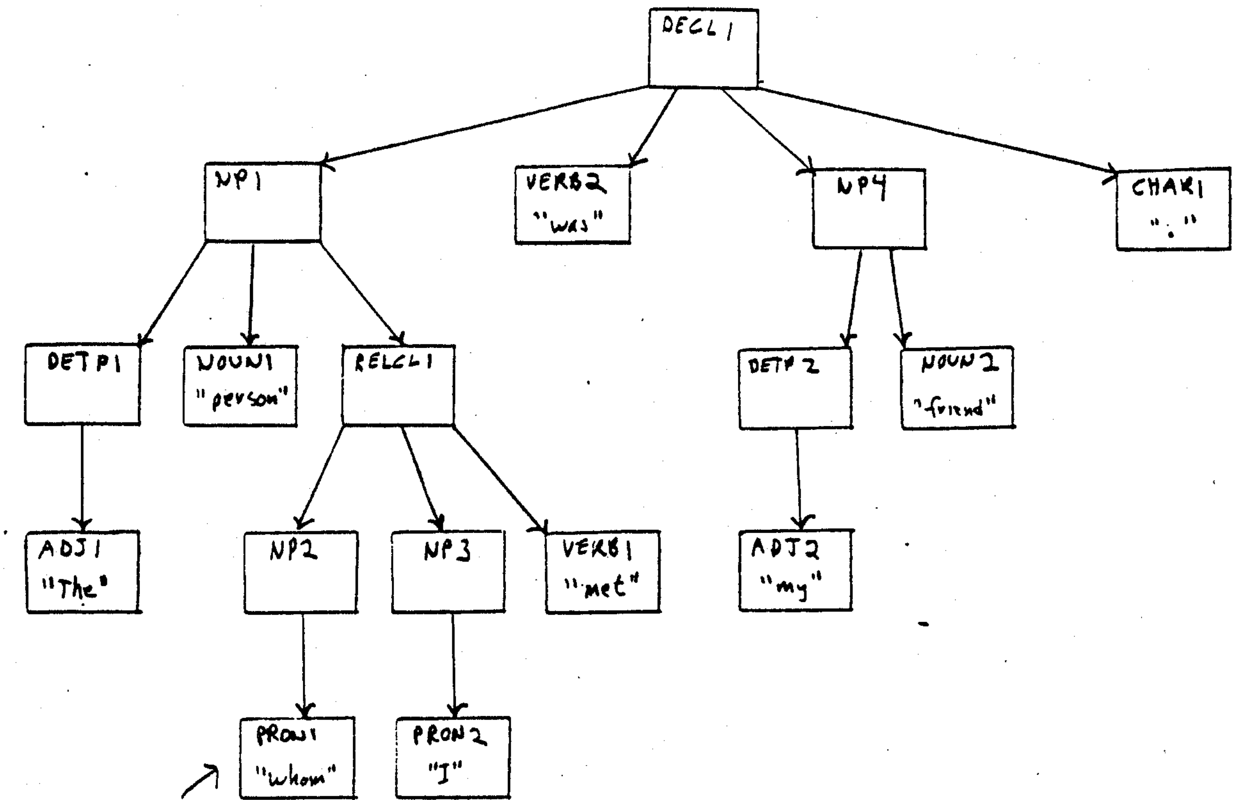


Figure 47



Rule: SynToSem1 produces logical form graph node "I" from NP3("I")





4801

Rule: SyntoSem1 produces logical form graph node "whom" from NP2 ("whom").

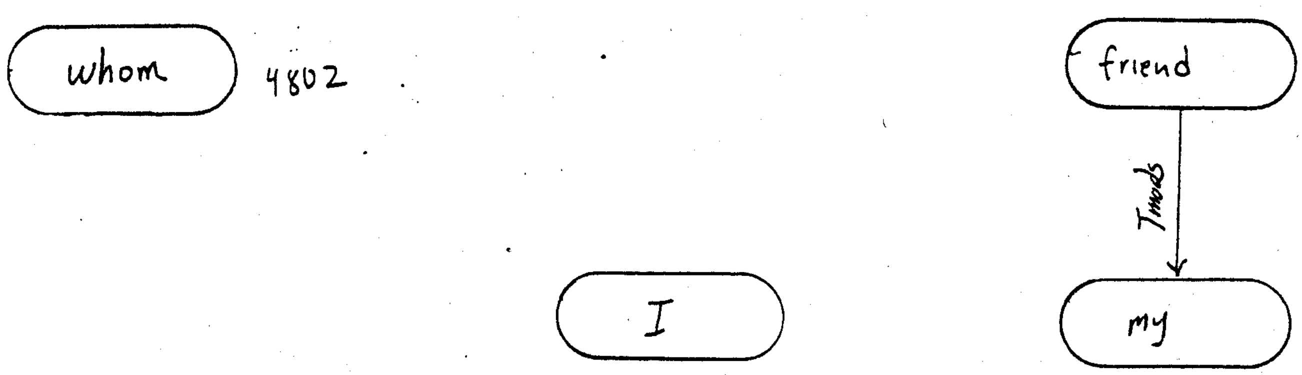
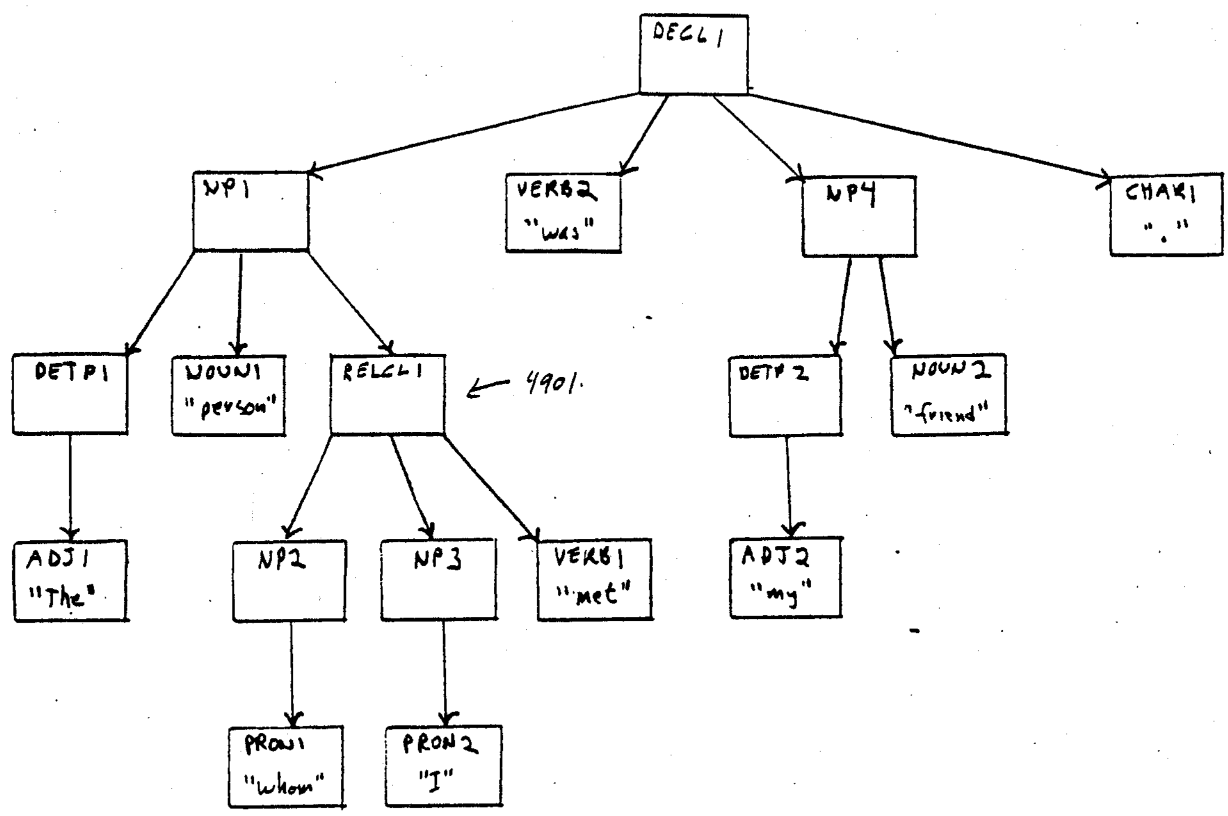


 Figure 49



Rule: SyntoSem1 produces logical form graph node "meet" from RELCL1 ("whom I met")

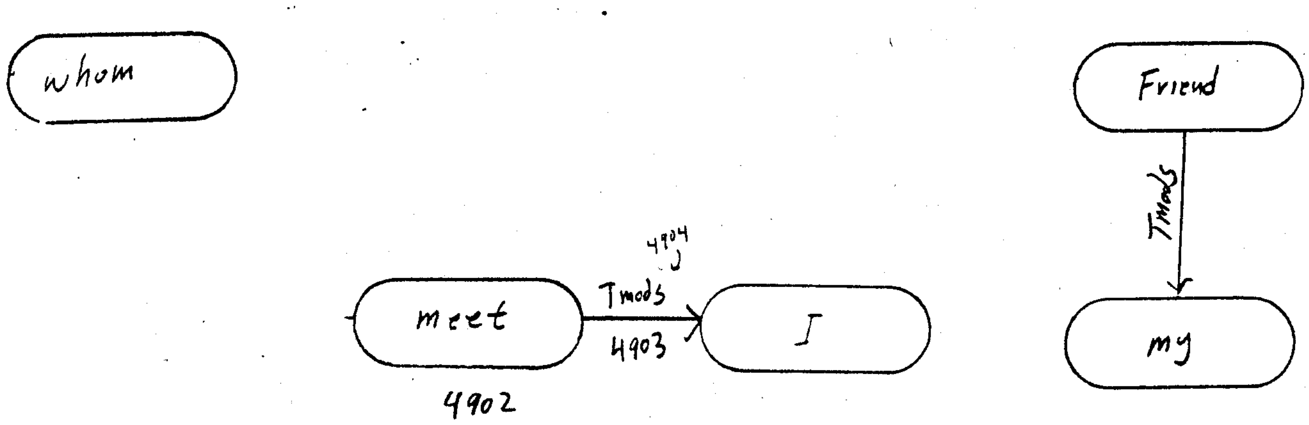
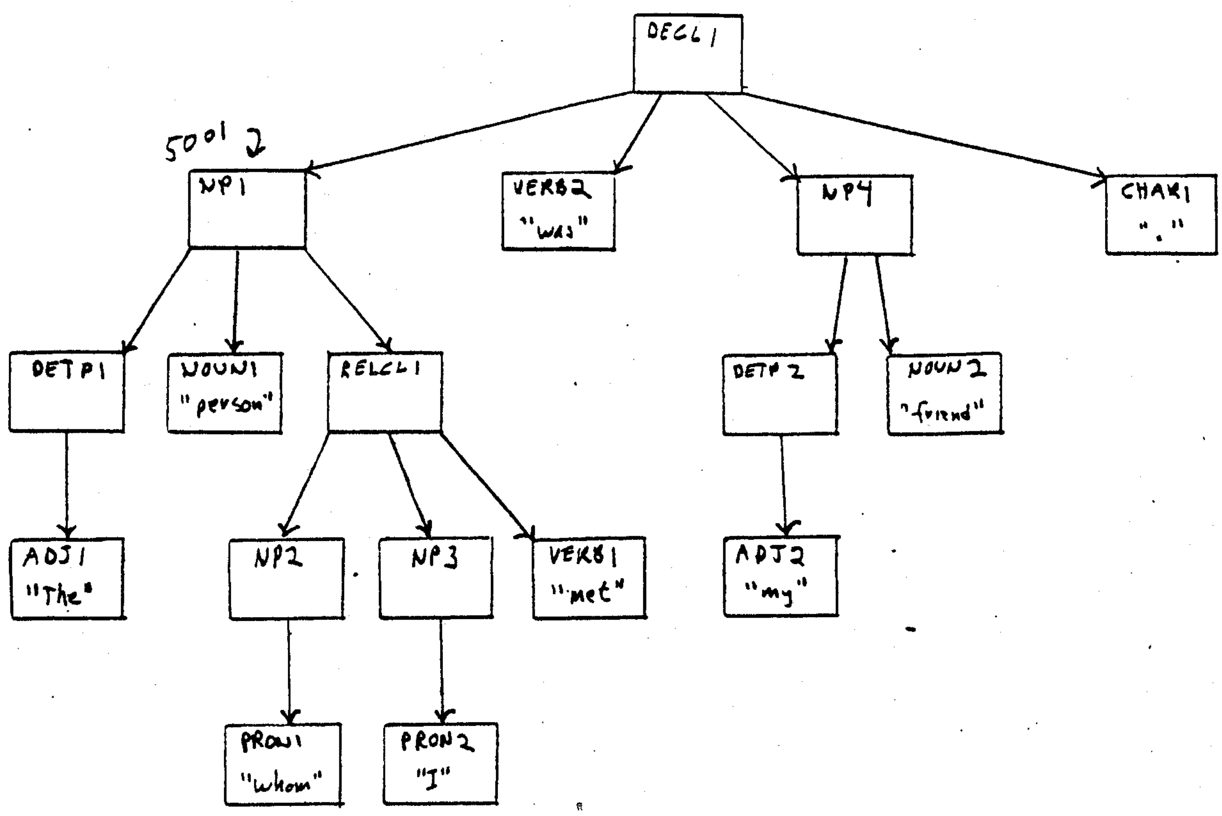
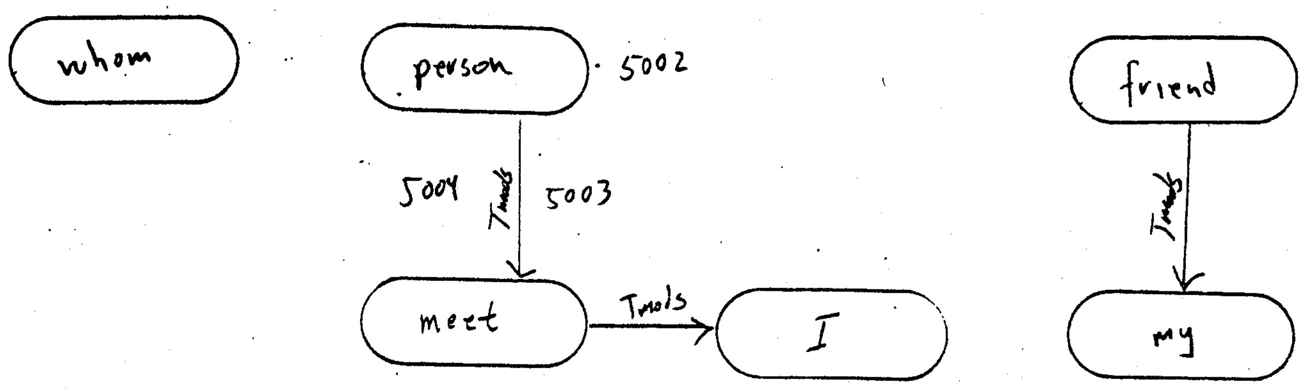


Figure 50

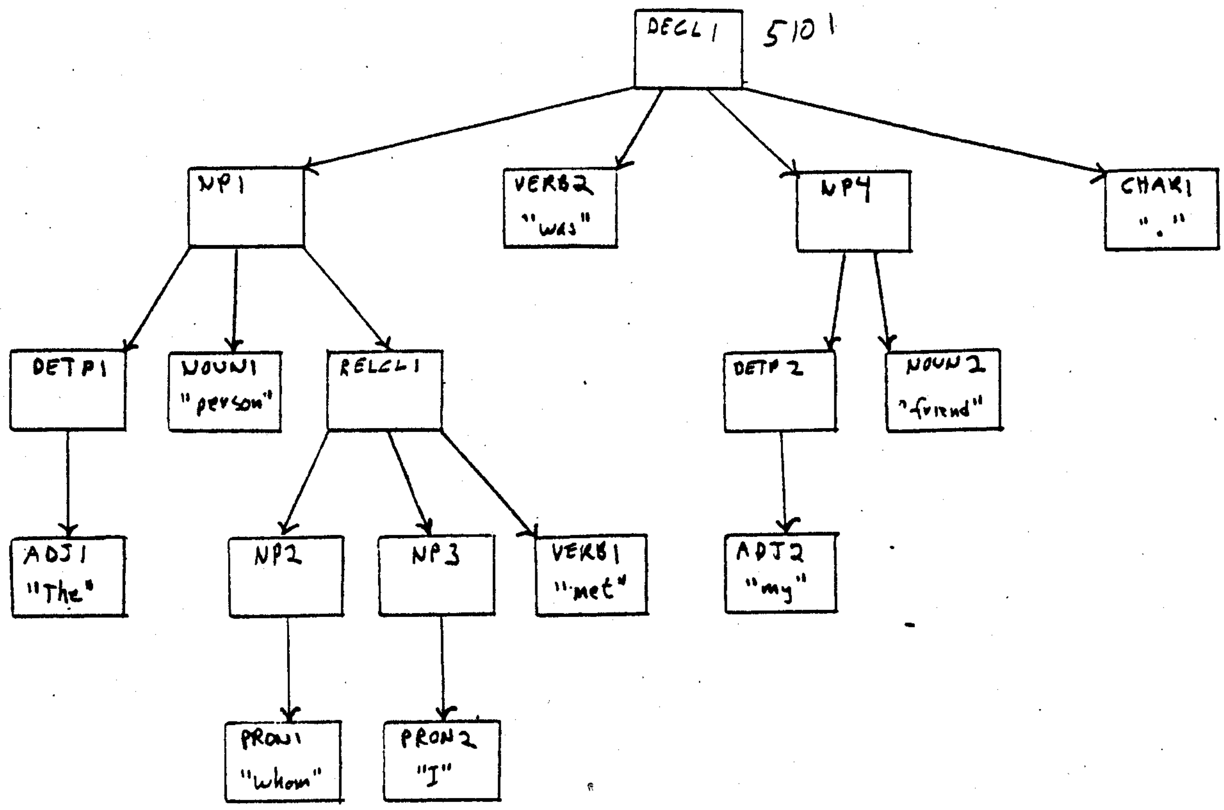


Rule: SyntoSem1 produces logical form graph node "person" from NP1 ("The ..... met",



PRINT OF DRAWINGS
AS ORIGINAL ILED Figure 51

08/674610



Rule: SynToSem1 produces logical form graph node "be" from DECL1

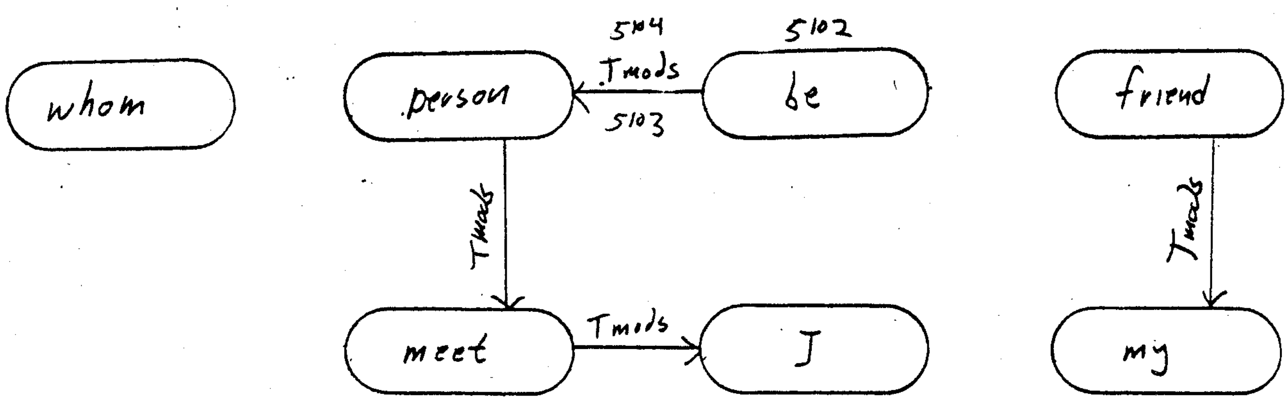
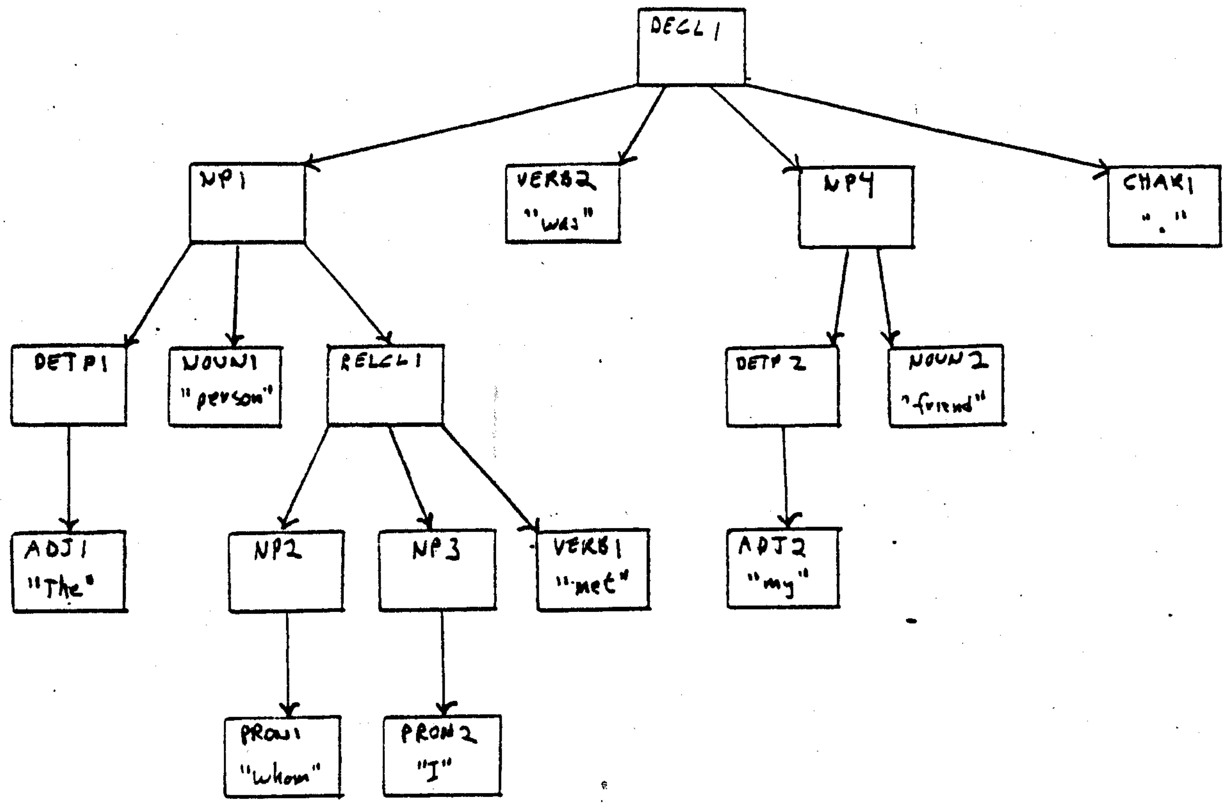


Figure 52



Rule: LF-Dsub1 with node "be" labels link and creates another link

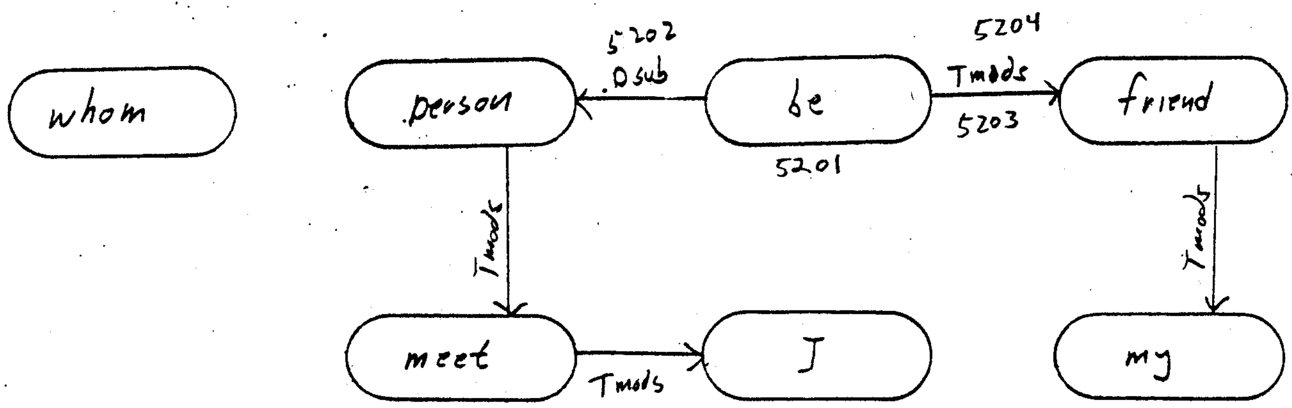
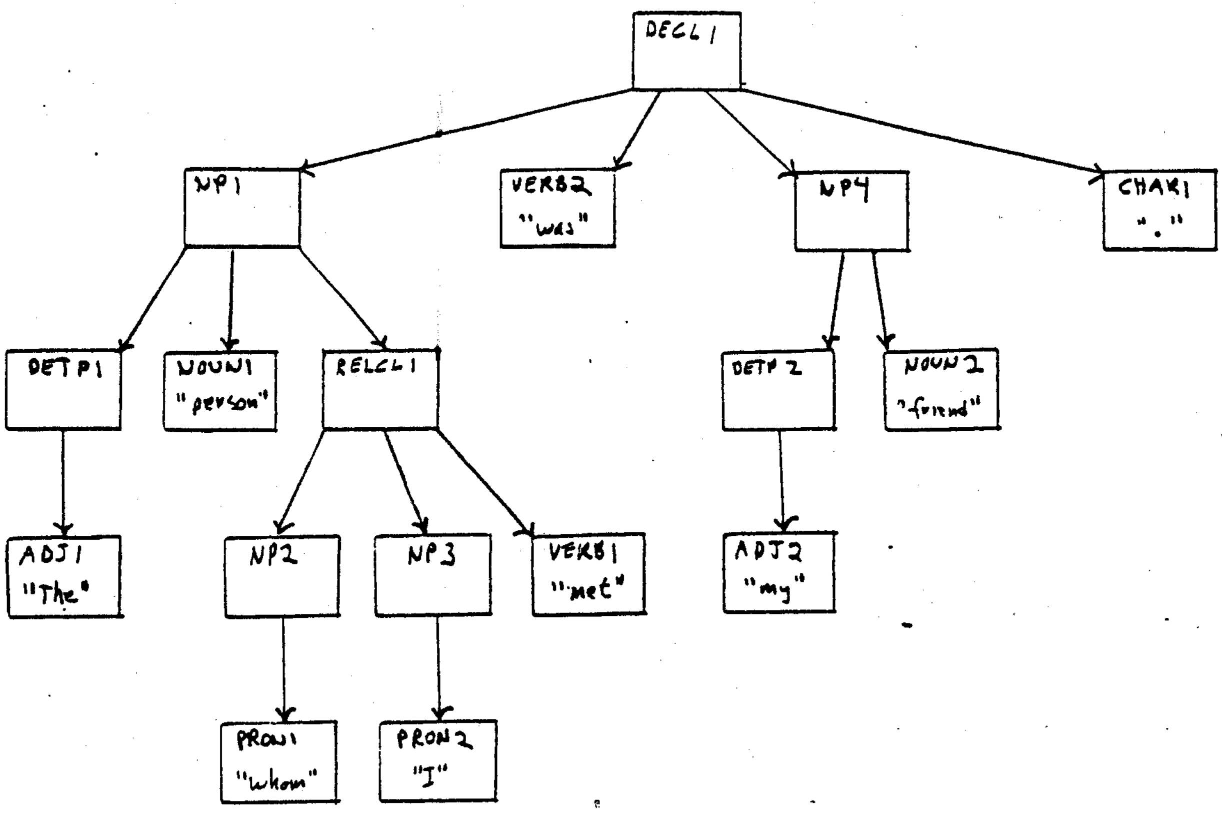
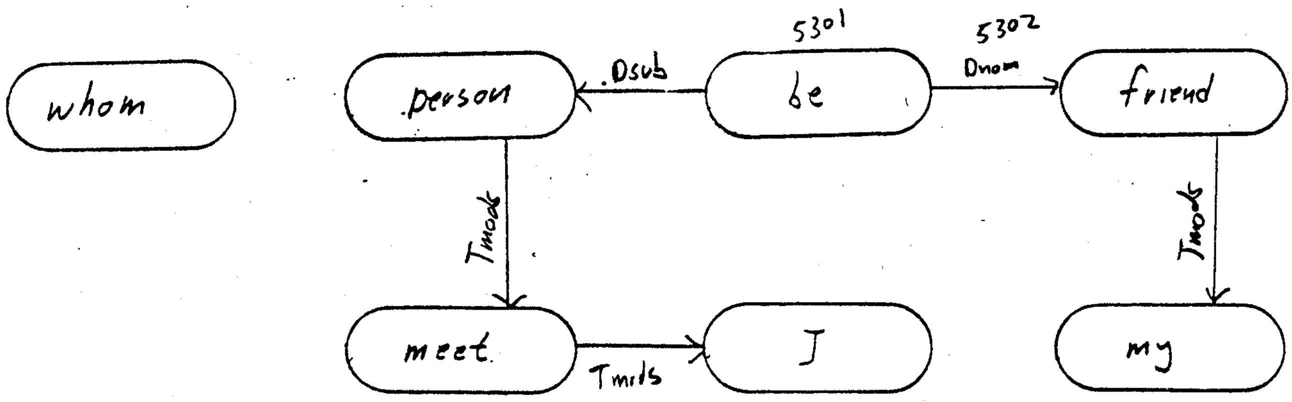


 Figure 53

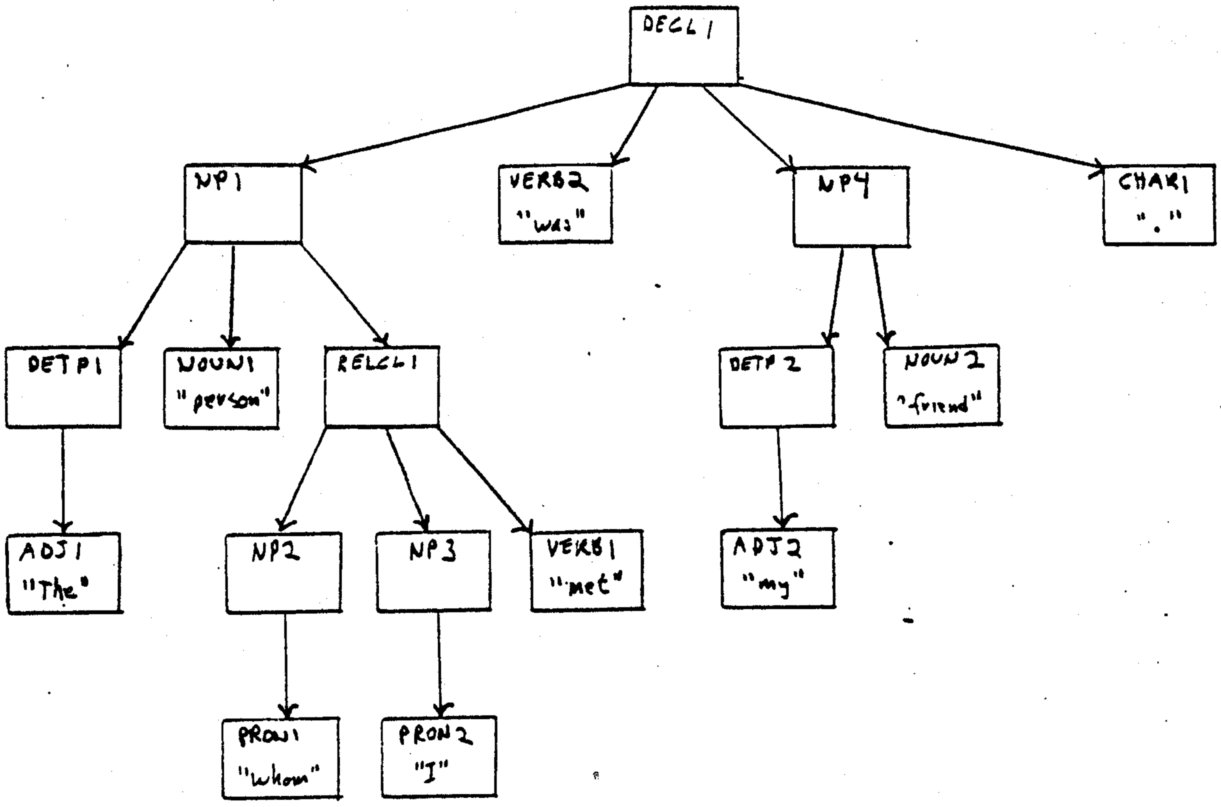


Rule: LF_Dnom with node "be" labels link



08/674610

Figure 54



Rule: LF_Props with node "person" labels link

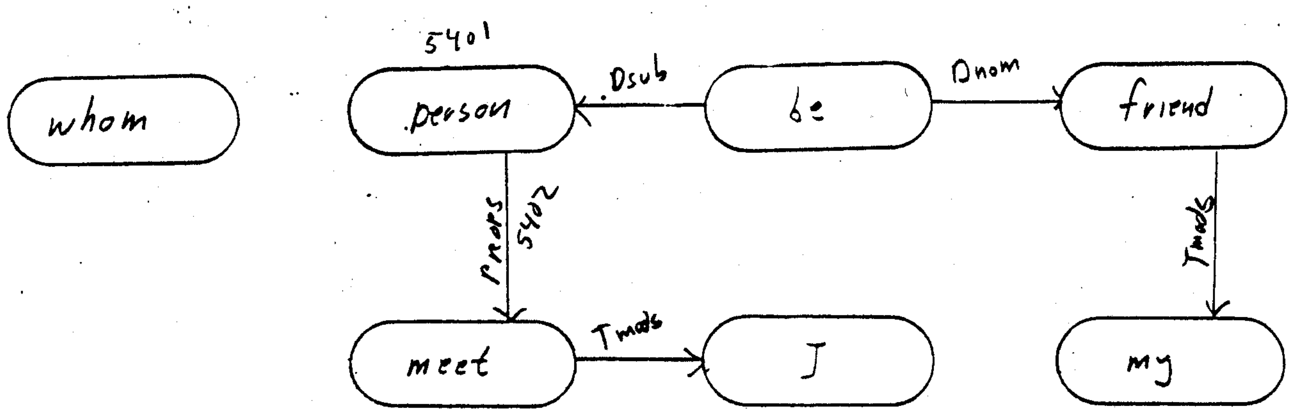
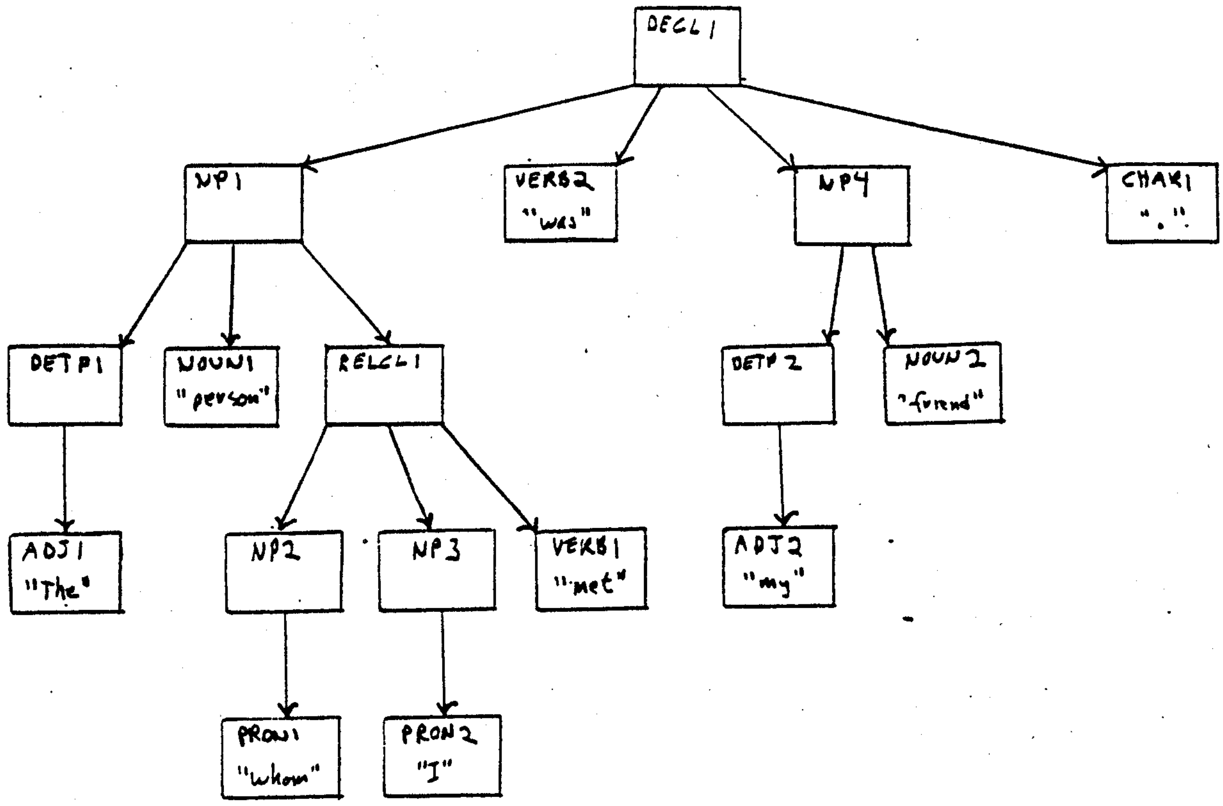


Figure 55



Rule: LF_Dsub1 with node "meet" labels link

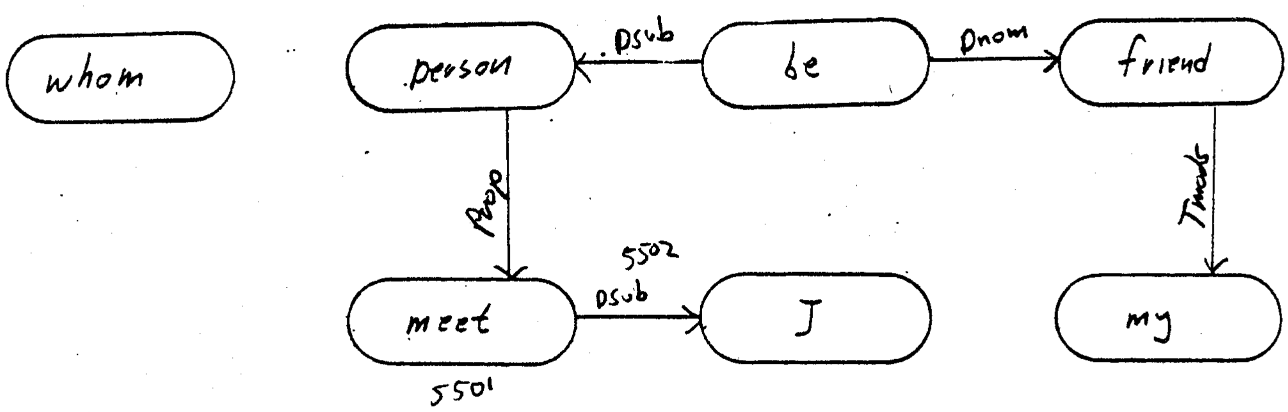
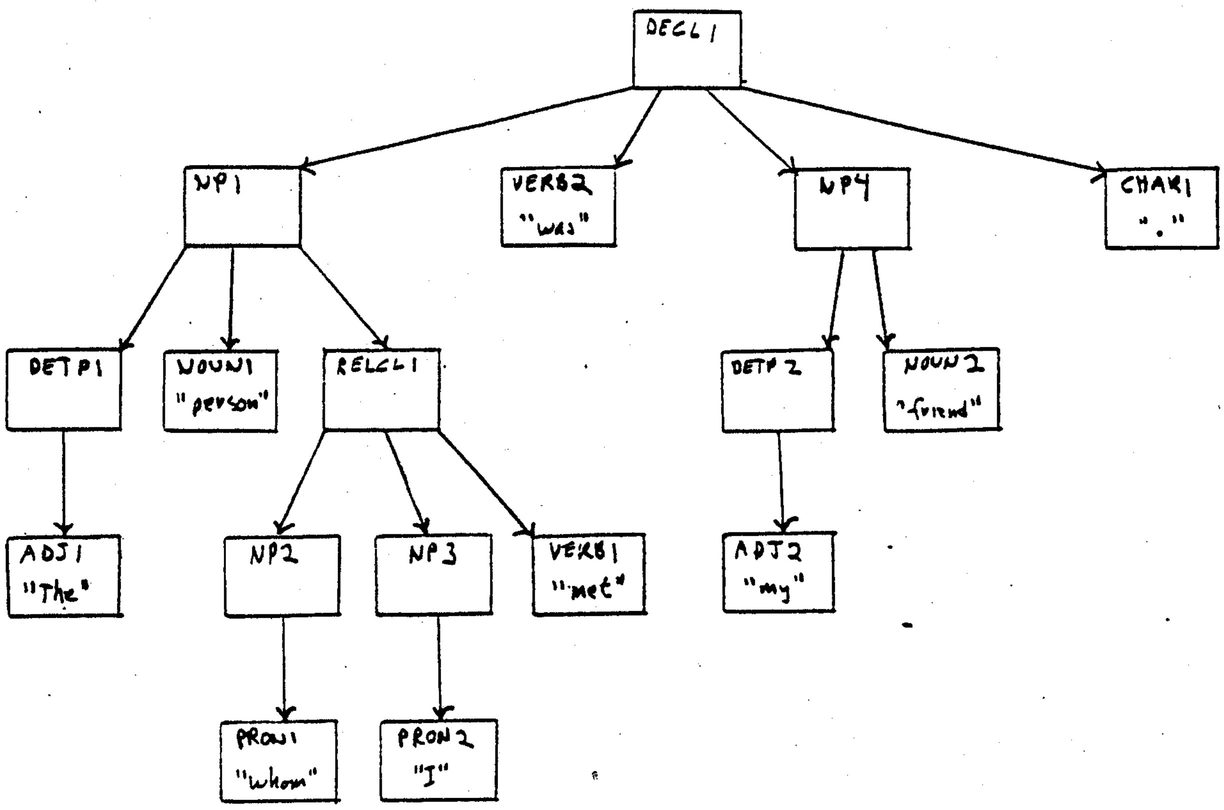


Figure 56



Rule: LF_Dobj1 with node "meet" adds link and labels it

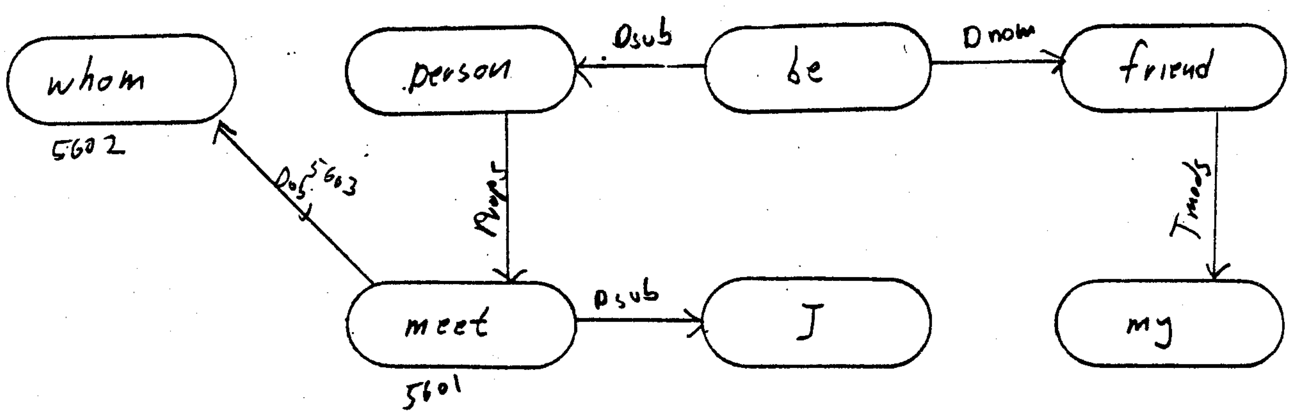
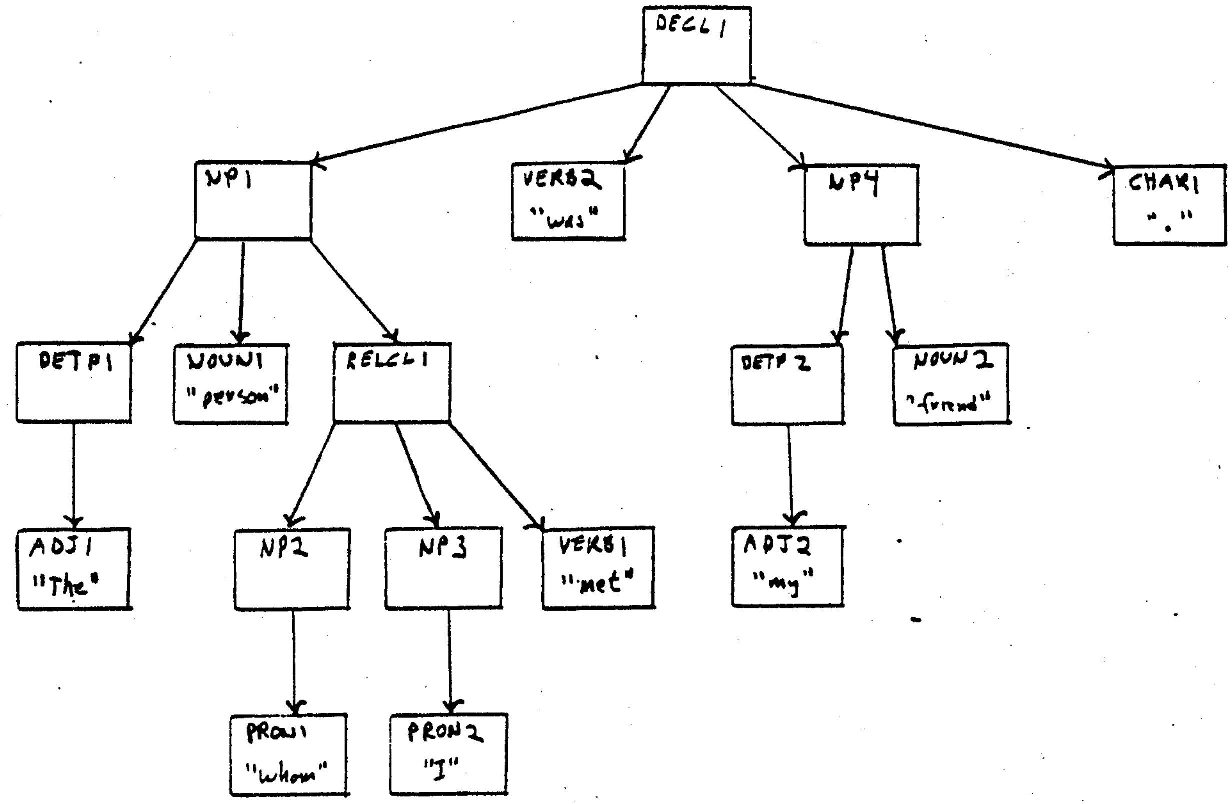


Figure 57



Rule: LF_Ops with node "friend" labels link

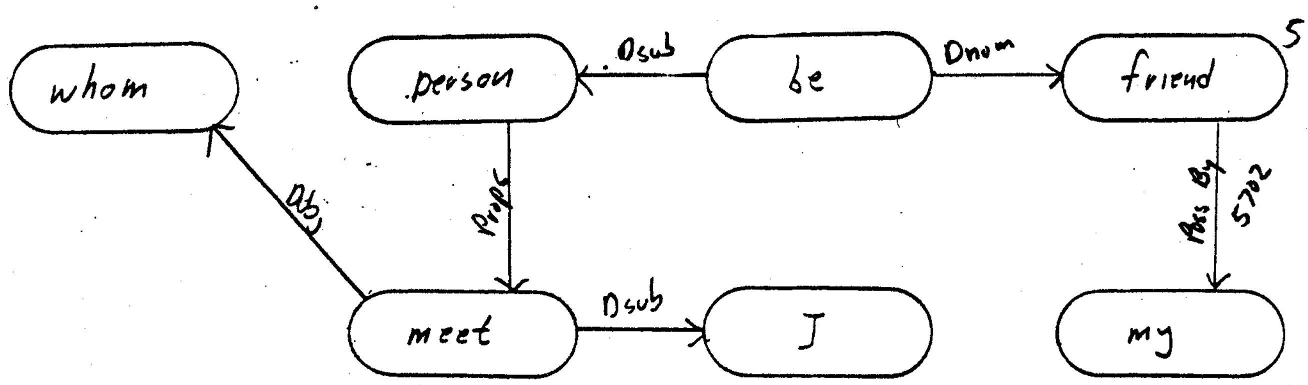
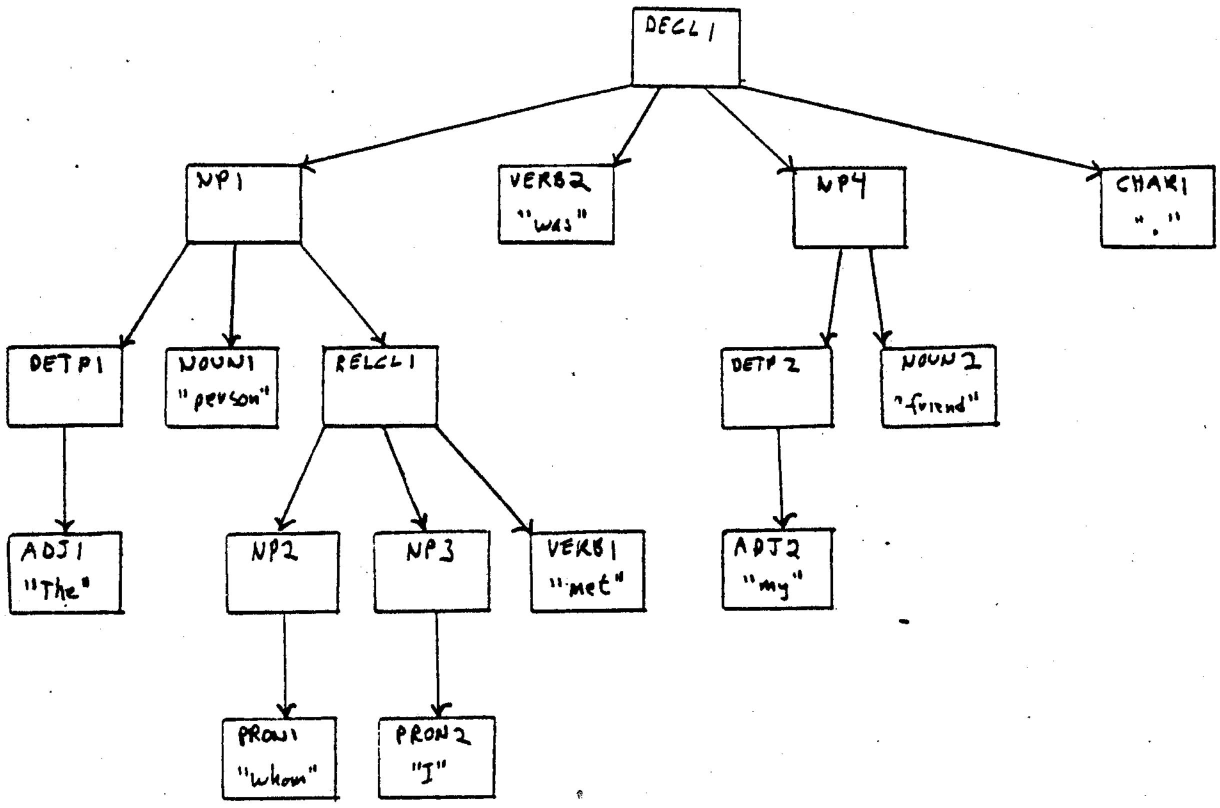


Figure 58

08/674610



Rule: PsLF_RelPro with node "whom" removes node and adds link

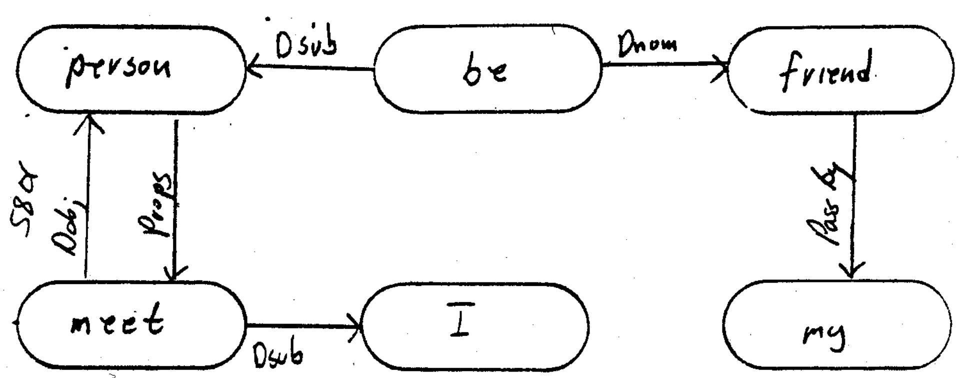
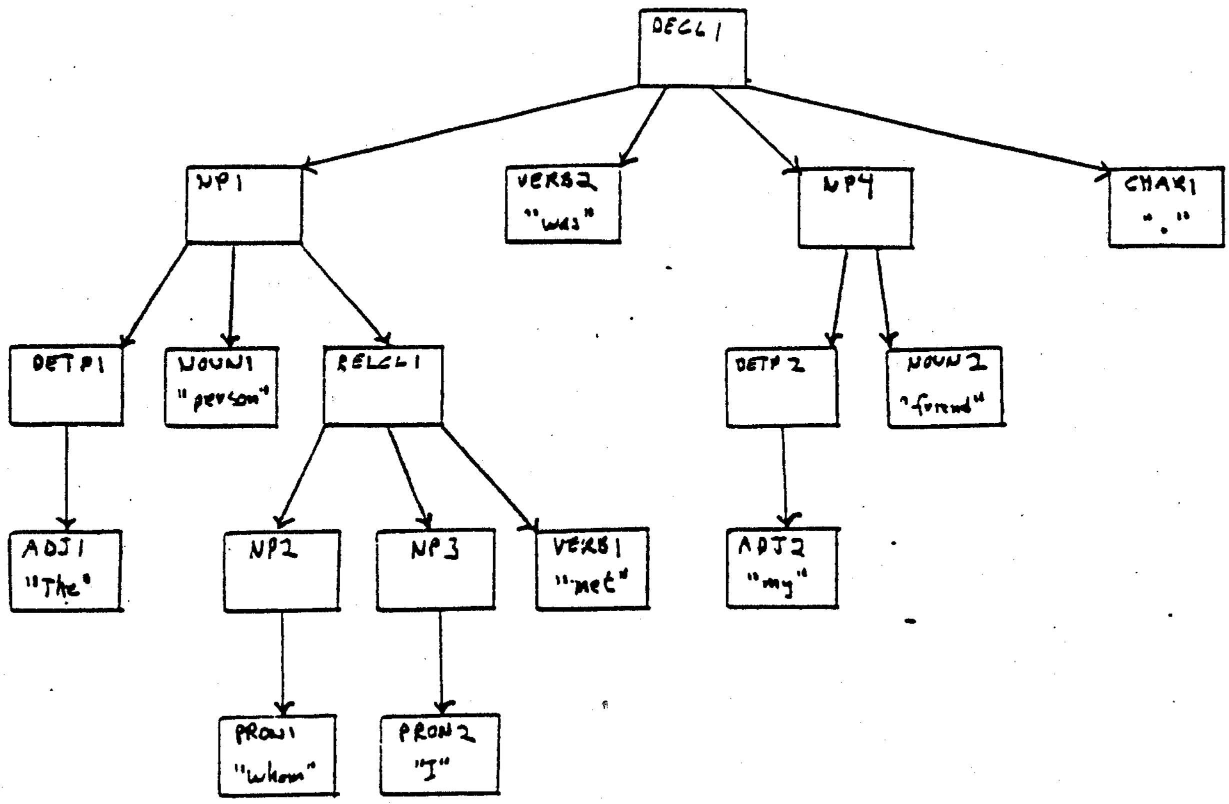
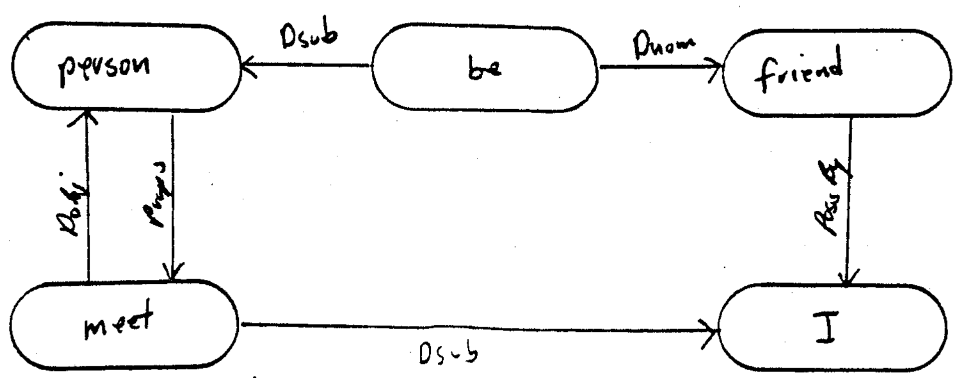


Figure 59



Rule: P3 LF_UnifyProns consolidates nodes "I" and "my" into a single node



UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
08/674,610	06/28/96	HEIDORN G	661005.447

#2

0282/0822

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

0000
DATE MAILED: 08/22/96

NOTICE TO FILE MISSING PARTS OF APPLICATION FILING DATE GRANTED

An Application Number and Filing Date have been assigned to this application. However, the items indicated below are missing. The required items and fees identified below must be timely submitted **ALONG WITH THE PAYMENT OF A SURCHARGE** for items 1 and 3-6 only of $ 130 for large entities or $ 65 for small entities who have filed a verified statement claiming such status. The surcharge is set forth in 37 CFR 1.16(e).

If all required items on this form are filed within the period set below, the total amount owed by applicant as a ☒ large entity, ☐ small entity (verified statement filed), is $ 1544.

Applicant is given ONE MONTH FROM THE DATE OF THIS LETTER, OR TWO MONTHS FROM THE FILING DATE of this application, **WHICHEVER IS LATER,** within which to file all required items and pay any fees required above to avoid abandonment. Extensions of time may be obtained by filing a petition accompanied by the extension fee under the provisions of 37 CFR 1.136(a).

1. ☒ The statutory basic filing fee is: ☒ missing ☐ insufficient. Applicant as a ☒ large entity ☐ small entity, must submit $ 750 to complete the basic filing fee.

2. ☒ Additional claim fees of $ 664.00 as a ☒ large entity, ☐ small entity, including any required multiple dependent claim fee, are required. Applicant must submit the additional claim fees or cancel the additional claims for which fees are due.

3. ☒ The oath or declaration:
   ☒ is missing.
   ☐ does not cover the newly submitted items.

   An oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date is required.

4. ☐ The oath or declaration does not identify the application to which it applies. An oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date, is required.

5. ☐ The signature(s) to the oath or declaration is/are: ☐ missing; ☐ by a person other than the inventor or a person qualified under 37 CFR 1.42, 1.43, or 1.47. A properly signed oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date, is required.

6. ☐ The signature of the following joint inventor(s) is missing from the oath or declaration:

   ____________________ An oath or declaration listing the names of all inventors and signed by the omitted inventor(s), identifying this application by the above Application Number and Filing Date, is required.

7. ☐ The application was filed in a language other than English. Applicant must file a verified English translation of the application and a fee of $__________ under 37 CFR 1.17(k), unless this fee has already been paid.

8. ☐ A $______________ processing fee is required since your check was returned without payment. (37 CFR 1.21(m)).

9. ☐ Your filing receipt was mailed in error because your check was returned without payment.

10. ☐ The application does not comply with the Sequence Rules. See attached Notice to Comply with Sequence Rules 37 CFR 1.821-1.825.

11. ☐ Other.

Direct the response to Box Missing Part and refer any questions to the Customer Service Center at (703) 308-1202.

A copy of this notice MUST be returned with the response.

OFFICE COPY

FORM PTO-1533 (REV. 11-94)

#750 /01
31 /02
352 /03
130 /05 d3 c0

SEP 23 1996

PATENT

#3

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to Box Missing Part, Assistant Commissioner for Patents, 2011 Jefferson Davis Highway, Washington, DC 20231.

September 18, 1996
Date

Maurice J Pirio
Maurice J. Pirio

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Docket No. : 661005.447
Date : September 18, 1996

Box Missing Part
Assistant Commissioner for Patents
2011 Jefferson Davis Highway
Washington, DC 20231

RESPONSE TO NOTICE TO FILE MISSING PARTS OF APPLICATION

Sir:

In response to the Notice to File Missing Parts dated August 22, 1996, please find enclosed a Declaration and Power of Attorney and Form PTO-1533 for the above-identified application.

The fees have been calculated as follows:

Basic Fee	$ 750
Total Claims (36, 16 extra)	352
Independent Claims (7, 4 extra)	312
Missing Parts Surcharge	130
TOTAL	$1,544

Enclosed is a check in the amount of $1,544 for the requisite fees. The Assistant Commissioner is hereby authorized to charge any additional filing fees or to credit

1 102 312.00 CK
1 103 352.00 CK
1 105 130.00 CK

any overpayment to Deposit Account No. 19-1090. A duplicate copy of this response is enclosed.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio
Registration No. 33,273

MJP:jss

Enclosures:
Postcard
Check No. 43791 for $1,544
Copy of this Response
Declaration and Power of Attorney
Copy of Form PTO-1533

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

c:\DVC\L-984

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
08/674,610	06/28/96	HEIDORN G	661005.447

0282/0822

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

DATE MAILED: 0000

08/22/96

NOTICE TO FILE MISSING PARTS OF APPLICATION
FILING DATE GRANTED

An Application Number and Filing Date have been assigned to this application. However, the items indicated below are missing. The required items and fees identified below must be timely submitted **ALONG WITH THE PAYMENT OF A SURCHARGE** for items 1 and 3-6 only of $ 130 for large entities or $ 65 for small entities who have filed a verified statement claiming such status. The surcharge is set forth in 37 CFR 1.16(e).

If all required items on this form are filed within the period set below, the total amount owed by applicant as a ☒ large entity, ☐ small entity (verified statement filed), is $ 1544.

Applicant is given **ONE MONTH FROM THE DATE OF THIS LETTER, OR TWO MONTHS FROM THE FILING DATE** of this application, **WHICHEVER IS LATER,** within which to file all required items and pay any fees required above to avoid abandonment. Extensions of time may be obtained by filing a petition accompanied by the extension fee under the provisions of 37 CFR 1.136(a).

1. ☑ The statutory basic filing fee is: ☑ missing ☐ insufficient. Applicant as a ☑ large entity ☐ small entity, must submit $ 750 to complete the basic filing fee.

2. ☑ Additional claim fees of $ 664 as a ☑ large entity, ☐ small entity, including any required multiple dependent claim fee, are required. Applicant must submit the additional claim fees or cancel the additional claims for which fees are due.

3. ☑ The oath or declaration:
   ☑ is missing.
   ☐ does not cover the newly submitted items.

   An oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date is required.

4. ☐ The oath or declaration does not identify the application to which it applies. An oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date, is required.

5. ☐ The signature(s) to the oath or declaration is/are: ☐ missing; ☐ by a person other than the inventor or a person qualified under 37 CFR 1.42, 1.43, or 1.47. A properly signed oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date, is required.

6. ☐ The signature of the following joint inventor(s) is missing from the oath or declaration:

   ____________________ An oath or declaration listing the names of all inventors and signed by the omitted inventor(s), identifying this application by the above Application Number and Filing Date, is required.

7. ☐ The application was filed in a language other than English. Applicant must file a verified English translation of the application and a fee of $__________ under 37 CFR 1.17(k), unless this fee has already been paid.

8. ☐ A $______________ processing fee is required since your check was returned without payment. (37 CFR 1.21(m)).

9. ☐ Your filing receipt was mailed in error because your check was returned without payment.

10. ☐ The application does not comply with the Sequence Rules. See attached Notice to Comply with Sequence Rules 37 CFR 1.821-1.825.

11. ☐ Other.

Direct the response to Box Missing Part and refer any questions to the Customer Service Center at (703) 308-1202.

A copy of this notice MUST be returned with the response.

FORM PTO-1533 (REV. 11-94)

COPY TO BE RETURNED WITH RESPONSE

#3

DECLARATION AND POWER OF ATTORNEY

As the below-named inventors, we declare that:

Our residences, post office addresses, and citizenships are as stated below under our names.

We believe we are the original, first, and joint inventors of the invention entitled "METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES," which is described and claimed in the specification and claims of Patent Application No. 08/674,610, which we filed in the United States Patent and Trademark Office on June 28, 1996 and for which a patent is sought.

We have reviewed and understand the contents of the above-entitled specification, including the claims, as amended by any amendment specifically referred to herein (if any).

We acknowledge our duty to disclose information of which we are aware which is material to the examination of this application in accordance with 37 C.F.R. § 1.56(a).

We hereby appoint RICHARD W. SEED, Registration No. 16,557; ROBERT J. BAYNHAM, Registration No. 22,846; EDWARD W. BULCHIS, Registration No. 26,847; GEORGE C. RONDEAU, JR., Registration No. 28,893; DAVID H. DEITS, Registration No. 28,066; WILLIAM O. FERRON, JR., Registration No. 30,633; PAUL T. MEIKLEJOHN, Registration No. 26,569; DAVID J. MAKI, Registration No. 31,392; RICHARD G. SHARKEY, Registration No. 32,629; DAVID V. CARLSON, Registration No. 31,153; MAURICE J. PIRIO, Registration No. 33,273; KARL R. HERMANNS, Registration No. 33,507; DAVID D. McMASTERS, Registration No. 33,963; ROBERT IANNUCCI, Registration No. 33,514; JOSHUA KING, Registration No. 35,570; MICHAEL J. DONOHUE, Registration No. 35,859; LORRAINE LINFORD, Registration No. 35,939; KEVIN J. CANNING, Registration No. 35,470; CHRISTOPHER J. DALEY-WATSON, Registration No. 34,807; STEVEN D. LAWRENZ, Registration No. 37,376; ROBERT G. WOOLSTON, Registration No. 37,263; CLARENCE T. TEGREENE, Registration No. 37,951; ELLEN M. BIERMAN, Registration No. 38,079; BRYAN A. SANTARELLI, Registration No. 37,560; MICHAEL L. KIKLIS, Registration No. 38,939; CAROL NOTTENBURG, Registration No. 39,317; CRAIG S. JEPSON, Registration No. 33,517; PAUL T. PARKER, Registration No. 38,264; JOHN C. STEWART, Registration No. 40,188; ROBERT W. BERGSTROM, Registration No. 39,906; HARRY K. AHN, Registration No. 40,243; DAVID W. PARKER, Registration No. 37,414; and ROBERT E. MATES, Registration No. 35,271, comprising the firm of SEED AND BERRY LLP, 6300 Columbia Center, Seattle, WA Washington 98104-7092, as our attorneys to prosecute this application and transact all business in the Patent and Trademark Office connected therewith. Please direct all telephone calls to Maurice J. Pirio at (206) 622-4900 and telecopies to (206) 682-6031.

We further declare that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further, that these statements were made with the knowledge that the making of willfully false statements and the like is punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and may jeopardize the validity of any patent issuing from this patent application.

George E. Heidorn

George Heidorn 1-00

Date Sept, 11, 1996

Residence : City of Bellevue, County of King
State of Washington WA

Citizenship : United States of America

P.O. Address : 3211 - 165th Place N.E.
Bellevue, Washington 98008

Karen Jensen

Karen Jensen 2-00

Date September 11, 1996

Residence : City of Bellevue, County of King
State of Washington WA

Citizenship : United States of America

P.O. Address : 3211 - 165th Place N.E.
Bellevue, Washington 98008

WPN\MJP\661005\447-Declaration (661005.447)

PATENT

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

December 9, 1996 | Maurice J. Pirio

Date · Maurice J. Pirio

#7 / 6/24/97

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES
Art Unit	:	2412
Docket No.	:	661005.447
Date	:	December 9, 1996

Assistant Commissioner for Patents
2011 Jefferson Davis Highway
Washington, DC 20231

INFORMATION DISCLOSURE STATEMENT

Sir:

In accordance with 37 C.F.R. §§ 1.56 and 1.97 through 1.98, applicants wish to make known to the Patent and Trademark Office the references set forth on the attached form PTO-1449 (copies of the cited references are enclosed). Although the aforesaid references are made known to the Patent and Trademark Office in compliance with applicants' duty to disclose all information they are aware of which is believed relevant to the examination of the above-identified application, applicants believe that their invention is patentable.

Please acknowledge receipt of this Information Disclosure Statement and kindly make the cited references of record in the above-identified application.

Respectfully submitted,

George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

MJP:sr
Enclosures:
- Postcard
- Form PTO-1449
- Cited References (3)

6300 Columbia Center
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031
c:\clients\661005\447\IDS cvr

FORM P (REV.7-8)

U.S. DEPARTMENT OF COMMERCE
PATENT AND TRADEMARK OFFICE

ATTY. DOCKET NO.	APPLICATION NO.
661005.447	08/674,610
APPLICANTS	
George Heidorn and Karen Jensen	
FILING DATE	GROUP ART UNIT
June 28, 1996	~~2412~~ 2741

INFORMATION DISCLOSURE STATEMENT

(Use several sheets if necessary)

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
[initials]	AA	5,146,406	9/8/92	Jensen	~~364~~ 704	~~419~~ 9	
	AB						
	AC						
	AD						
	AE						
	AF						
	AG						
	AH						
	AI						
	AJ						
	AK						

FOREIGN PATENT DOCUMENTS

		DOCUMENT NUMBER	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLATION YES	TRANSLATION NO
	AL							
	AM							

OTHER PRIOR ART *(Including Author, Title, Date, Pertinent Pages, Etc.)*

[initials]	AN	Jensen, Karen et al., *Natural Language Processing: The PLNLP Approach*, Kluwer Academic Publishers, Boston, 1993.
[initials]	AO	Garside, Roger et al., *The Computational Analysis of English: A Corpus-Based Approach*, Longman, pp. 97-109, 1987.
	AP	

EXAMINER	DATE CONSIDERED
[signature]	3/18/98

* EXAMINER: Initial if reference considered, whether or not criteria is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant(s).

Form P7-27/V1/1/94
c:\clients\661005\447\1449

NATURAL LANGUAGE PROCESSING: The PLNLP Approach

edited
by

Karen Jensen
George E. Heidorn
Stephen D. Richardson

Microsoft Corporation

KLUWER ACADEMIC PUBLISHERS
Boston/Dordrecht/London

Distributors for North America:
Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061 USA

Distributors for all other countries:
Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS

Library of Congress Cataloging-in-Publication Data
Natural language processing : the PLNLP approach / edited by Karen Jensen, George E. Heidorn, Stephen D. Richardson.
p. cm. -- (The Kluwer international series in engineering and computer science ; 196. Natural language processing and machine translation)
Includes bibliographical references and index.
ISBN: 0-7923-9279-5 (acid free paper)
1. Natural language processing (Computer science)
2. Computational linguistics. 3. PLNLP (Computer program language)
I. Jensen, Karen. II. Heidorn, George E. (George Emil), 1938-
III. Richardson, Stephen D. IV. Series : Kluwer international series in engineering and computer science ; SECS 196. V. Series : Kluwer international series in engineering and computer science. Natural language processing and machine translation.
QA76.9.N38N385 1993
006.3'5--dc20 92-30803
CIP

Printed on acid-free paper.

Printed in the United States of America

Contents

Authors

Jean-Louis Binot
BIM
4, Kwikstraat, Everberg 3078
Belgium
email: jlb@sunbim.be

Jean-Pierre Chanod
Centre Scientifique IBM France
3-5 Place Vendome
75001 Paris
France
email: chanod@fribm11.bitnet

Lisa Braden-Harder
IBM T.J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598
USA
email: harder@watson.ibm.com

Martin Chodorow
Department of Psychology
Hunter College of the City
University of New York
695 Park Avenue
New York, NY 10021
USA
email: mschc@cunyvm.bitnet

Ee Ah Choo
Institute of Systems Science
National University of Singapore
Heng Mui Keng Terrace,
Kent Ridge
Singapore 0511
Republic of Singapore
email: issje@nusvm.bitnet

Jan Engh
INESC
R. Alves Redol, 9
P-1000 Lisboa
Portugal

Koh Mui Koong
Institute of Systems Science
National University of Singapore
Heng Mui Keng Terrace,
Kent Ridge
Singapore 0511
Republic of Singapore
email: isskmk@nusvm.bitnet

Simonetta Montemagni
Dipartimento di Linguistica
Università di Pisa
Via Santa Maria 36
56100 Pisa
Italy
email: grammar@icnucevm.bitnet

Bettina Harriehausen-Mühlbauer
IBM Germany GmbH
Postfach 10 30 68
6900 Heidelberg
Germany
email: harrieha@dhdibm1.bitnet

George Heidorn
Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
email: georgehe@microsoft.com

Chapter 1

Introduction

Karen Jensen, George Heidorn, and Stephen Richardson

Abstract

During the 1980s, a group of dedicated researchers developed a very broad-coverage natural language processing system. This included a programming language (PLNLP: Programming Language for Natural Language Processing), development tools, and analysis and synthesis components. This book presents the first published collection of papers written about this system and its use.

The PLNLP approach can be identified with several important contributions to the field of Natural Language Processing (NLP): (1) Augmented Phrase Structure Grammar (APSG), using exclusively binary rules; (2) practical experience leading toward the linguistic theory of Transductive Grammar; (3) the use of natural language itself as a knowledge representation language, and the resulting exploitation of online text resources as a source of semantic information and as a knowledge base; and (4) an integrated, incremental system design that allows one linguistic level to evolve naturally into the next.

The material in this chapter comes from three previously published sources. Section 1.2 (System components) is taken from: Jensen 1991 and Segond and Jensen 1992. Section 1.3 (PLNLP language and system) is taken from Jensen et al. 1986.

1.1 The starting point

Natural language is easy for people and hard for machines; that much, at least, has been established during the last 40 years, as people have been trying to get computers to handle our native languages in ways that would be interesting and useful to us. This book describes one group's pursuit of that goal.

The birthplace of this group was the International Business Machines Corporation, centrally the IBM Thomas J. Watson Research Center. This work took place mainly during the 1980s, and involved many people at several sites around the world. Many of these people are still affiliated with IBM; some have left to take academic or other posts. The editors of this book joined the Research Division of the Microsoft Corporation in June of 1991.

The programming language and system used for this research is known as PLNLP (Programming Language for Natural Language Processing, often pronounced "Penelope" or "Plenelope"). The PLNLP system is an integrated, incremental system for broad-coverage syntactic and semantic analysis and synthesis of natural language. More work has been done for the English language than for any other, but that is only an historical accident. Significant work has also been done for several European and Asian languages, some of which is described in this book.

Any natural language processing (NLP) system can be conceptually divided into three parts: grammar, dictionary, and the programming system that holds everything together. A division between grammar and dictionary (or "lexicon") is inherited from linguistics, the study of language—a discipline that has been around for a very long time indeed, provably for more than 2000 years. The addition of a computational component is what turns this enterprise into a very current affair. Computational linguistics, or NLP, has only been with us since roughly the late 1950s.

Traditionally, grammars are systems of rules that mediate between symbols and meanings. The rules have a dynamic nature, and are supposed to embody generalizations that hold true for many symbols and combinations of symbols—the more general, the better. Lexicons are repositories for particular units like words or phrases, and for information about those units. Lexical information is prototypically static and specific in nature. But this comparison provides only the slimmest of guidelines for designing a real system. A key question is what should be the proper distribution of work between grammar rules and lexicon?

If there is a definitive answer to that question, it has not emerged yet. One predominant tendency, for example in some systems based on versions of Chomskyan theory, is to account for linguistic phenomena primarily in the rules.

Another tendency, found typically in systems that derive from Lexical Functional Grammar, is to pack a lot of information into the lexical entries, and simplify the rules as much as possible. Regardless of the tendency, the actual situation is best seen as a continuum. A certain amount of information is necessary to produce the analysis, and we have two poles for the distribution of that information: rules and lexicon. Different systems choose different ranges along the continuum. Now we can rephrase the central question in engineering terms: which distribution will be most efficient in the long run?

1.2 System components

To position the various PLNLP language processing systems along that continuum is not easy. The systems consist of different components, and the distribution of lexical versus rule information is different for each component. Traditional components of linguistic theory include phonetics, phonology, morphology, syntax, semantics, discourse, and pragmatics. Since the PLNLP system restricts its input to typed text, it does not deal with phonetics and phonology. Morphology is included as part of the initial lexicon. To date, most activity has been in syntax and semantics; only beginnings (but significant beginnings) have been achieved in discourse and pragmatics.

Gradual evolution during the eighties suggested the following components for our English analysis system:

1. *Syntax*, consisting of the broad-coverage English sentence analysis grammar PEG (the PLNLP English Grammar), coupled with a large lexicon that is basically a list of English word stems with fairly simple associated feature information. The lexicon started with entries from the full online *Webster's Seventh New Collegiate Dictionary*. Although the number of words covered is great, the amount of information per word is small, compared to what is described for many other syntactic grammars. Linguistic information is distributed much more heavily over the rules than over the lexicon in this component.

2. *Corrected syntax (reassignment)*, which takes the output from PEG and resolves many ambiguous syntactic analyses, based on semantic information from online dictionary definitions. It recursively calls PEG to retrieve and analyze dictionary information, applying heuristic rules to that information in order to "bootstrap" its way from syntax into semantics. During this process, some word sense disambiguation falls out automatically as a result of the attachment disambiguation. Since the lexicon associated with this component actually contains entire online dictionaries, the amount of information per word is huge, much larger than what is described for other NLP systems; and the distribution

of linguistic information in this component is heavily skewed toward the lexicon.

3. *Derivation of logical form (PEGASUS),* which takes the corrected sentence parse and produces a graph that is the basis for further semantic processing. In so doing, it determines: (a) the structure of arguments and adjuncts; (b) pronoun reference; and (c) verb phrase anaphora (the semantic structure of elided VPs). These steps are accomplished by a set of procedures that operate strictly on the output of the reassignment component, without consulting any additional lexical information.

4. *Sense disambiguation,* which narrows down the possible senses of verbs and nouns in the sentence. It operates on the output from the previous component, mapping target words, in their sentential context, to relevant online dictionary entries. Taking advantage of all available information—from the parsed analysis, from the dictionary, and from other sources—the most likely possible senses of words are identified through a strategy that weights various types of evidence and ranks senses according to a similarity measure. The balance of information in this component is again weighted toward the lexicon, because of the significant use that is made of online dictionary resources.

5. *Normalization of semantic relations.* The first step in constructing a discourse model is to refine the semantic graph, with the goal of creating a common or normalized representation for all inputs that mean the same thing. The notion "mean the same thing" is still fairly intuitive, and this component has been only partially implemented. Normalization routines are intended to inspect nodes in the graph and the relations between those nodes, and identify rule-governed paraphrases across a wide variety of syntactic domains. Of course, the process of normalization has already been started by PEGASUS; for example, equivalent argument structures are produced for active and passive variants of an English sentence. Although the routines are semantically oriented, they do not lose access to the surface syntactic differences.

6. *Paragraph (discourse) model.* After all possible sentential normalizations have been made, the system must join sentence graphs to build a formal model of those discourse chunks which, in written text, are typically called paragraphs. Much remains tentative here, because this component has also been only partially implemented. However, at this point it seems likely that the distribution of activity between rules and lexicon will be fairly even for this component and for the preceding one.

The papers collected in this volume discuss all of these components, some in more detail than others. Although the architecture described here is sequential, with each component building on the output of the preceding one, this is just for development purposes. It should be stressed that, in the PLNLP approach, "broad-coverage" means having the goal of processing literally *any* input string in the designated natural language. This is a non-trivial goal, and entails that each component requires a non-trivial amount of work and time to produce; a sequential architecture makes it easier to concentrate on certain tasks in the beginning stages of development. However, the control structure could be non-sequential (i.e., similar to agenda-controlled systems), taking advantage of the parallel processing facilities afforded by PLNLP.

More has naturally been done on the first parts of the system than on later parts, and this state of affairs is reflected in the number of papers collected here for each component. Most of the chapters are reprints of previously published papers, with minor editing changes. Some are shortened versions. Two chapters (including this one) are constructed from parts of previously published papers. Also there is one paper that has not appeared before. This book brings this material together for the first time, presenting a coherent picture of the evolution and structure of a natural language analysis system that has been credited with providing linguistic coverage among the broadest of any in existence.

Since this is a book of collected works, the chapters are somewhat independent from each other. They also contain more redundancy than would otherwise be the case. We have tried to minimize redundancy and maximize cohesiveness wherever possible, and we beg our readers' indulgence for whatever discontinuities might remain.

1.3 PLNLP (the Programming Language for Natural Language Processing)

What is presented in this book is a blueprint for an integrated NLP analysis system, in which the traditional theoretical modules of syntax, semantics, and discourse are linked to form a unified whole. A major key to this linking is the fact that the entire system can be written in a single formalism, PLNLP (Heidorn 1972), which provides an efficient and smooth joining of information across the modules. PLNLP is intended for natural language and knowledge base applications. It can be used, by linguists or by anyone else who is interested in the structure of human languages, to write computational grammars that both describe the language and perform tasks associated with language use.

Both rule-based and procedural programming facilities are available in PLNLP. The basic units of the language are rules and records. The records are collections of attributes and values, where the values can be pointers to other records, thereby creating a complex network of information. In addition to attribute-value records, PLNLP also supports lists, strings, etc. Furthermore, it allows "loose" data typing, with implicit declarations of variables and run-time type handling (like LISP). Procedures and production rules can be intermixed. PLNLP provides for both determinism and non-determinism, and features a concise notation, with essentially no reserved words.

PLNLP's *augmented phrase structure grammar* (APSG) rules (Heidorn 1975) are divided into two types: decoding (parsing or analyzing) and encoding (generating or synthesizing). Associated with each of the rule types is a separate algorithm: in the decoding case, processing is done bottom-up and parallel; and in the encoding case, processing is top-down and serial. The basic structure of the rules, however, is the same in either case. There is a left-hand side, where the constituent(s) is/are identified and where conditions are tested which must be true before the rule can be activated; there is the rule arrow; and there is a right-hand side, where the new constituent(s) is/are identified and new structure is specified:

```
CONSTITUENT1(conditions)
CONSTITUENT2(conditions)
->    CONSTITUENT3(structure-building actions)
```

Figure 1. General form for PLNLP decoding (parsing) rule

The PLNLP system supports interactive program development and efficient program execution. The system itself is written in PLNLP, and bootstrapped. It is portable to many target programming languages and to many computer families. The PLNLP system is an outgrowth of the Natural Language Processor (NLP), which was first described in Heidorn 1972. It is not bound to, and therefore can be used by, any linguistic theory.

To minimize the effort required in writing a computational grammar, the PLNLP runtime environment provides a shell into which the user, typically a linguist, loads a grammar definition as a set of PLNLP rules. Having loaded a grammar, the user can then choose to decode a sentence according to the rules in that grammar. Details of the process of decoding can be displayed or suppressed, by selecting from a variety of tracing options.

Whenever a stretch of input text ending with full stop (typically a sentence) is processed, the tree for that parse is displayed. After the decoding process is completed, the user can perform a "post-mortem analysis," to see what non-terminal symbols of the grammar were discovered at various positions in the input, and what attribute values were associated with their instances. In addition, sophisticated debugging functions allow the grammarian to pinpoint the exact place in a rule where a parse failed to proceed to completion, or the exact differences between two ambiguous parses of the same input. With these tracing and analyzing facilities, the user can easily locate a problem in the grammar, instead of having to infer it by elaborate deduction.

One of the most important aspects of PLNLP is its ability to express complex relationships by means of interconnecting networks of records. Permanent or enduring knowledge structures can be constructed as part of the grammar loading process; during decoding or encoding, these structures can be modified, or new transient structures created. The user can call for a record to be displayed by giving its name, or can call for the records which are the roots of parse trees resulting from the last decode operation, and can then follow pointer links to other records in the network. At any time, the values of attributes in the displayed record can be viewed, and changed if desired. The user can also call for the displayed record to be encoded (generated) as an instance of a syntactic category. All of these features of PLNLP have been used successfully by grammar developers, who have built systems for such diverse languages as Norwegian, Italian, Arabic, Korean, and English.

1.4 A guide to the chapters of this book

The remaining chapters mirror the major system components as follows:

Syntax: chapters 2–10
Reassignment: chapters 11–15
Logical form: chapters 16–18
Sense disambiguation: chapters 19–20
Normalization of semantic relations: chapter 21
Paragraph model: chapter 22

Chapters 2–10 are associated most closely with the first system component, the initial syntactic sketch. Chapter 2, by Alexis Manaster Ramer, paves the way for a new theoretical linguistic orientation that would explain the evolution and architecture of a system such as this one, which we may call a *transductive* grammar system. Manaster Ramer contrasts PEG-style grammars with traditional generative grammars. In a generative grammar a string is either well-formed or not, and if not then it has, strictly speaking, no structural analysis. A transductive grammar, on the other hand, analyzes any input whatsoever, thus making no *initial* distinction between well-formed and ill-formed input. Some such distinction may still be made as a part of the structural analysis, but it is not the case that the way you determine if a string is well-formed is to check whether it *has* an analysis (as you do in a generative grammar). Instead you check *what kind* of analysis it has.

The PLNLP system was built empirically, driven by the demands of textual data as they presented themselves, but always with the hope that the text corpora would suggest a theoretical model, and that the system itself could serve as data for a more explanatory theory of language. This, in fact, has always seemed to be the great promise that computational linguistics holds for linguistics proper. The theory of Transductive Grammar is a large step toward the fulfillment of that promise.

The next three chapters (chapters 3–5) describe the initial syntactic component, consisting of three sub-sections: the English analysis grammar PEG, which tries to produce a single reasonable parse for every input sentence (or sentence fragment); the parse metric, which ranks them in case PEG produces more than one parse; and the parse fitting procedures, which handle those cases where PEG fails to produce any parse covering the whole input string. PEG is discussed in chapter 3; the parse metric is explained in chapter 4, and parse fitting in chapter 5. Chapter 4, by George Heidorn, gives the original (1982) published statement of the metric. Additional, unpublished work has been done to enhance it since that time. Chapter 5, originally published in 1983, lays out the purposes and early strategies for parse fitting—a technique that guarantees robustness in a computational grammar since it produces some reasonable parse for *any* input. This robustness is a necessary characteristic of a transductive grammar.

Yael Ravin's chapter on "Grammar Errors and Style Weaknesses in a Text-Critiquing System" (chapter 6) straddles the boundary between theory and application. From the theoretical point of view, an error-detecting capability is a salient characteristic of a transductive grammar. Both generative and transductive approaches agree that a grammar should be able to identify ill-formed input. But the transductive insight is that this judgment need not result in parse failure, and, in fact, does not even have to be made by the same parsing rules that describe constituent structure. Chapter 6 details those filtering aspects of the PLNLP system that make grammaticality judgments. But from the application point of view, the main thrust of Ravin's chapter is to explain how those judgments are used in a text-critiquing system, to offer suggestions and corrections to users in a word-processing environment.

Chapters 7–10 present examples of applications that make use of the initial syntactic sketch. The flagship application is the text critiquing system introduced in chapter 6, first known as "Epistle" (until 1984) and then called "Critique." Critique is described by Stephen Richardson and Lisa Braden-Harder in chapter 7. The initial analysis component of this system has also been used as a front end for machine translation systems from English to several diverse languages. The most developed of such systems is the English-Japanese SHALT, built at IBM-Japan's Tokyo Research Laboratory under the direction of Taijiro Tsutsumi, who describes that work in chapter 8. SHALT is used regularly within IBM-Japan, at this time, to translate English computer manuals into Japanese. From a very different environment, Diana Santos and her group developed the PORTUGA system, entirely written in PLNLP, to handle English-Portuguese translation. In the process, they suggested solutions to several interesting MT problems. This work is detailed in chapter 9. Another use for the syntactic sketch is described in chapter 10, by Judith Klavans, Martin Chodorow and Nina Wachholder. They used PEG to parse dictionary definitions, then analyzed the syntactic and semantic impact of certain head nouns in the definitions, demonstrating how relationships and semantic networks might be automatically inferred from the dictionary.

The next five chapters (chapters 11–15) center on the second analysis component, reassignment, which was first proposed in 1986. The basic problem was this: we had a syntactic grammar with a reasonable promise of true broad coverage; where would the broad-coverage semantics come from that was needed to match the syntax?

The problem manifested itself urgently first as a need to correct those syntactic attachments, such as prepositional phrase attachments, that cannot be successfully resolved without semantic information. The accepted way of providing such information at the time was to hand-code it in some knowledge representation, like scripts or frames or graphs, often using a specially-designed knowledge representation language. But if hand-coding were necessary, then true broad coverage would be very difficult to attain.

We discovered in 1986 that we could get a lot of the requisite information from a good dictionary of English. By invoking PEG on dictionary definitions, we could produce parses from which, with some heuristic rules, semantic data could be extracted and used to correct prepositional phrase attachments in a number of interesting cases. From there came the realization that natural language itself is a knowledge representation language. Every text that has been written is a knowledge representation. Much of the information that we call semantic, pragmatic, or common-sense does not have to be coded in stylized forms (although it may be useful to do so in some cases); once we have a broad-coverage syntax, we can access the knowledge in NL text and exploit it for the purpose of bootstrapping the system to higher levels of understanding.

Chapter 11, by Jean-Louis Binot and Karen Jensen, presents the early results of the experimentation with prepositional phrases. Simonetta Montemagni and Lucy Vanderwende, in Chapter 12, explore how to extract semantic information from dictionary definitions. There are two main parts to such definitions: the genus (syntactically and semantically central) term, and the differentiae (everything else of interest). In contrast to most other work in the area, which concentrates on genus terms, Montemagni and Vanderwende scrutinize the differentiae, using patterns found in the syntactic structural analyses to identify important semantic relations. Chapter 13, by Vanderwende, extends the prepositional phrase reattachment strategy to other problematic constructions, such as the definition of relationships between nouns in a phrase like "vegetable market." All text corpora are possible and promising sources of knowledge; chapters 14 and 15 focus on natural language dictionaries, which are information repositories with their own particular characteristics. In chapter 14, Ravin applies disambiguation techniques to the definitions themselves. Montemagni has written an initial syntactic grammar for Italian and, in chapter 15, describes how she uses it within the framework of the Esprit BRA Acquilex project, tailoring the output from her general-purpose grammar to facilitate the parsing of dictionary definitions, with the goal of extracting semantic information that will then be fed into a formal knowledge base.

The logical form component (also called PEGASUS) is discussed in chapters 16–18. The basic purpose, structure and results of PEGASUS are explained in chapter 16. Chapter 17, by Jean-Pierre Chanod, Bettina Harriehausen, and Montemagni, presents an example of computational *comparative* linguistics: post-processing techniques for deriving logical forms are applied to the syntactic analyses of three languages—French, German, and Italian—and are shown to produce, automatically, similar or identical semantic predicate-argument structures. Chapter 18, by Ee Ah Choo et al., describes a machine translation system, under construction at the Institute for Systems Science in Singapore, that uses the argument-structure outputs from PEGASUS as intermediate structures for English-to-Chinese machine translation. This system may be compared with the MT systems presented in chapters 8 and 9, which use output from the initial syntax only.

Sense disambiguation is a critical and difficult task for machine understanding. This task is distributed to some extent throughout the analysis system, but comes into high focus in the fourth component. Braden-Harder, making use of techniques drawn from information retrieval, shows in chapter 19 that multiple sources of information, including both explicit and implicit dictionary cues, can be exploited to help the system determine relevant senses of words. In chapter 20, Tsutsumi demonstrates a case-based approach, using disambiguated example sentences along with hierarchies of synonyms and taxonyms.

The last two chapters move beyond semantics and into conceptual structure. As Fredérique Segond demonstrates in chapter 21, normalization involves taking information (including as much word sense disambiguation as possible) from the preceding components, and providing a foundation for the next stage of analysis, discourse. By joining sentence structures, we arrive at the paragraph, the next grammatical unit beyond the sentence. In the final chapter, Wlodek Zadrozny and Karen Jensen examine the nature of this linguistic construct, demonstrating a correspondence between paragraphs and certain types of logical models, and suggesting how to formalize the notional definition of a paragraph as a "unit of thought." The authors conclude that background knowledge, as exemplified in online reference works, can be used automatically to build a discourse model.

We can summarize some of the major contributions of the PLNLP approach to Natural Language Processing as follows:

(1) the Augmented Phrase Structure Grammar (APSG) formalism with binary rules, which provides an efficient and comprehensive tool for NLP;

(2) practical experience leading to the theory of Transductive Grammar, which presents a new formal perspective on the discipline of linguistics, and provides a mathematical framework for NLP;

(3) the idea that natural language is a knowledge representation language and can be computationally (and efficiently) exploited as such. This idea manifested itself first in the use of online dictionaries as a source of semantic information (a major theme in this book);

(4) an integrated and incremental system design, which moves smoothly from syntax through semantics into discourse.

Chapter 16

PEGASUS: Deriving Argument Structures after Syntax

Karen Jensen*

Abstract

PEGASUS is the third component in the PLNLP analysis system, following syntax and reassignment. Its purpose is to produce a semantic representation, or logical form, for each input sentence or sentence fragment. To do this it computes: (a) the structure of arguments and adjuncts for each clause; (b) NP (pronoun)-anaphora; and (c) VP-anaphora (for elided VPs). While doing this, PEGASUS must maintain broad coverage (that is, accept and analyze unrestricted input text). More commonly in NLP systems, the computation of such meaning structures is considered impossible unless a particular domain is specified. This chapter explains these steps and then compares the PLNLP approach with other current approaches to defining predicate-argument structures.

This chapter is excerpted from Segond and Jensen 1992.

16.1 Introduction

PEGASUS, the third component in the PLNLP analysis system after syntax and reassignment, provides a definitive move from syntax to semantics, where "semantics" is understood to involve, minimally, the definition of case frames or thematic roles (i.e., predicate-argument relations). The most obvious display of this is in its input and output representations. Input is shown as a parse tree; output, as a labeled, directed graph. (The underlying information is in the form of attribute-value record structures throughout.) A tree is primarily a syntactic representation, where linear ordering and categorial dominance are significant. A graph is a semantic representation; linear ordering is no longer relevant because whatever information it provided has been assigned to arc labels or features in the graph. This output can also be called a *logical form.*

In order to derive the logical form, PEGASUS must correctly make many challenging argument assignments, such as long-distance dependencies (e.g., assigning the right object for "ate" in "What did Mary say that John ate?"); functional control (e.g., assigning the proper subjects and objects to infinitives); the active/passive relationship (making sure that active and passive variants have the same underlying arguments); and so forth. The program must also identify meaningful relationships between head words of phrases and their modifiers or adjuncts. In addition, NP-anaphora (including pronoun and definite noun phrase referents) and VP-anaphora (assigning the correct arguments and adjuncts within elided VPs) must be completed; and the entire input string must be properly quantified. Currently PEGASUS does not handle definite NP reference or quantification, but does handle the other phenomena mentioned here.

16.2 Arguments and adjuncts

Consider the sentence, "After dinner, Mary gave a cake to John." Figure 1 shows the syntactic (tree) representation for that sentence after it has been processed by the first two analysis components, and figure 2 shows the semantic graph produced by PEGASUS for the same sentence.

A graph is produced by displaying only those attributes and values that are defined to be semantic. However, the underlying record structure contains all attributes resulting from the parse. In this fashion, all levels and types of information, from morphological to syntactic to semantic and beyond, are constantly available. This principle of accountability holds throughout the PLNLP system.

```
DECL1|----PP1|-------PREP1-----"After"
     |       |-------NOUN1*----"dinner"
     |       |-------PUNC1-----","
     |----NP1--------NOUN2*----"Mary"
     |----VERB1*-----"gave"
     |----NP2|-------DETP1------ADJ1*----"a"
     |       |-------NOUN3*----"cake"
     |----PP2|-------PREP2-----"to"
     |       |-------NOUN4*----"John"
     |----PUNC2------"."
```

Figure 1. Syntactic parse tree

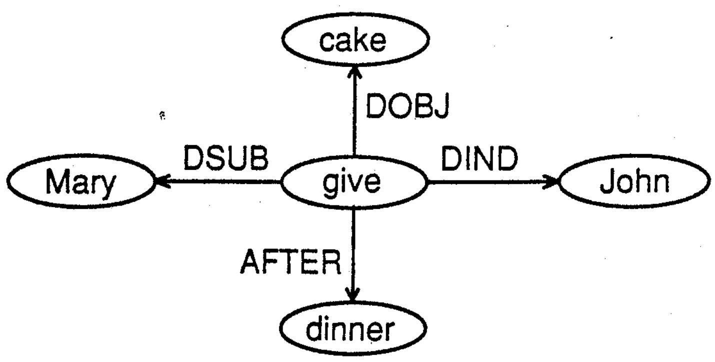


Figure 2. Semantic graph for the sentence in figure 1

In an NLP system that uses attribute-value pairs, argument structures can be produced (a) by defining, for each node, attribute names that correspond to the desired argument or adjunct types, and (b) by assigning values to those attributes. It is customary to think of argument names like AGENT, PATIENT, etc. However, although these labels are tantalizingly semantic in nature, there is as yet no uniformly acceptable way of relating syntactic structure to them. Therefore we avoid such labels, at least for the time being. We adopt, instead, the notion of "deep" cases or functional roles:

DSUB: deep subject
DIND: deep indirect object
DOBJ: deep object
DNOM: deep predicate nominative
DCMP: deep object complement

All deep argument attributes are added to the analysis record structure by PEGASUS. For very simple clauses, deep arguments correspond exactly to the surface syntactic arguments. For example, in "John ate the cake," the NP "John" fills the roles of both surface and deep subject; "the cake" fills the roles of both surface and deep object. In such simple cases, the deep argument attributes could as well have been assigned by the syntax rules; they are assigned by PEGASUS just to simplify the overall system architecture.

Each major class node is examined, and, if it contains more than just one single (head) word, each associated word is evaluated for possible assignment to some deep-structure attribute. In addition to the deep case labels, the following non-syntactic, non-argument attributes define the fully elaborated structure:

- PRED: predicate (basic term) label
- PTCL: particle in two-part verbs
- OPS: operator, like demonstratives and quantifiers
- NADJ: adjective modifying a noun
- PADJ: predicate adjective
- PROP: otherwise unspecified modifier that is a clause
- MODS: otherwise unspecified modifier that is not a clause; also, members of a coordinated structure, whether clausal or not.

And in addition to these, attributes are defined to point to prepositional phrases and subordinate clauses. The names of these attributes are actually the lemmas of those prepositions and conjunctions that begin their phrases and clauses. In this fashion, a step is taken toward a more semantic analysis of these constituents, without the necessity of going all the way to case labels like "locative" and "durative."

The procedure starts by renaming the surface arguments in all cases, as described previously. Then it calls a set of sub-procedures, each one of which is designed to solve a particular piece of the argument puzzle. Here is an outline of the flow of control taken for the specification of arguments and adjuncts:

1. Assign arguments and modifiers to all VP nodes:
   a. Assign arguments, in this order:
      1) Unbounded dependencies, e.g., in "What did Mary say that John ate?" the DOBJ of "ate" is "What."
      2) Functional control, e.g., in "John wanted to eat the cake," the DSUB of "eat" is "John."
      3) Passives, e.g., in "The cake was eaten by John," the DSUB is "John" and the DOBJ is "the cake."
      4) Indirect object paraphrases, e.g., the structure for "Mary gave a surprise to John" must be identical to the structure for "Mary gave John a surprise."

5) Indirect object special cases, e.g., in "I told the story," the syntactic object "the story" is the DOBJ; but in "I told the woman," the syntactic object "the woman" is the DIND.

6) Extraposition, e.g., "John ate the cake" is the clausal DSUB of the sentence "It appears that John ate the cake."

b. Assign modifiers (all adjuncts): prepositional, adjective, and adverb phrases; adverbial noun phrases; subordinate clauses; infinitives; comment clauses; participial modifiers; sentential relative clauses; etc.

2. Assign modifiers (and arguments, when possible) to all NP nodes.
3. Assign modifiers to all AJP (adjective phrase) nodes.
4. Assign modifiers to all AVP (adverb phrase) nodes.
5. For fitted parses (chapter 5), assign the pieces as modifiers of the entire analysis.
6. Clean up the attribute-value structure by deleting some unwanted features. For example, the verb "give" can be either transitive or ditransitive, and therefore brings both features with it from the dictionary into the parse. If the given sentence turns out to be a simple transitive (e.g., "They gave $10"), leave the transitive feature but erase the ditransitive feature, which is no longer true at the clause level.

The focus of linguistic interest here is on the assignment of arguments to VP nodes. Ordering of the sub-procedures is important. Long-distance dependencies must be resolved before functional control is assigned, and both of these maneuvers must be performed before passives are handled. The ordering presented here was experimentally determined by parsing sentences that contain more than one of the phenomena noted. Figure 3 shows the graph for a sentence that combines passivization with a long-distance dependency: "Who did John say was kissed by Mary?"

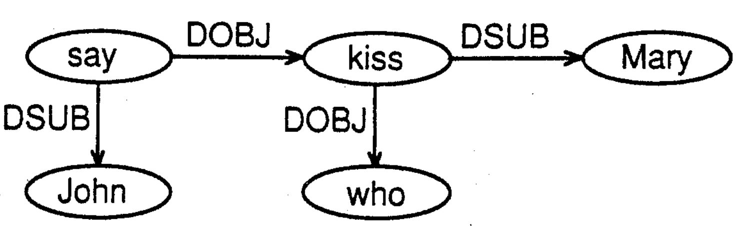


Figure 3. Graph for the sentence, "Who did John say was kissed by Mary?"

Subcategorization features on verbs are used more strictly here than they are used in the first component, the broad-coverage syntactic sketch. Also, although selectional features were not found to be useful in constructing the syntactic sketch, they are both useful and necessary for defining deep arguments in PEGASUS. With unbounded dependencies, it is important to distinguish the probable subcategorization types of verbs in the sentence, and also some selectional ("semantic") features on nouns, since the argument structure will vary depending on the interplay between these two pieces of information. Consider the difference between arguments of the two verbs "kiss" and "write." "Kiss" is usually a simple transitive verb, not ditransitive; but "write" is very often a ditransitive. These facts affect the default interpretation of their surface syntactic objects:

Who did John kiss?	"Who" is DOBJ of the verb "kiss."
Who did John write?	"Who" is DIND of the verb "write."

"Who" carries an animate (selectional) feature. If we change "who" to "what" (non-animate), however, the same ditransitive verb no longer has a deep indirect object:

What did John write?	"What" is DOBJ of the verb "write."

The sub-procedure for functional control handles not only infinitive clauses, but also participial clauses, both present and past. These constructions often require argument assignment over long intervening stretches of text. In the sentence "Mary, just as you predicted, arrived excitedly waving her hands," "Mary" is DSUB of the present participle "excitedly waving her hands." In the sentence "Bolstered by an outpouring of public confidence, John accepted the post," "John" is DOBJ of the past participle "Bolstered by an outpouring..."

All of the other sub-procedures for argument assignment are linguistically interesting to various degrees, but none of them is quite so complex as the procedures for unbounded dependency and functional control.

16.3 Anaphora

The resolution of NP-anaphora is done by assigning a REFerent attribute, which has as its value a pointer to the NP that is being referred to, as shown in figure 4. Currently only pronouns are handled; definite NPs can be added.

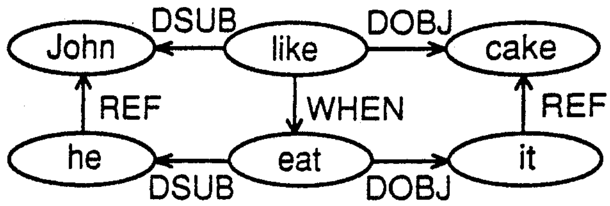


Figure 4. Graph (including pronoun reference) for the sentence, "John liked the cake when he ate it."

The resolution of VP-anaphora is defined by filling in the missing arguments and adjuncts for elided VPs. The sentence "John ate the cake and Peter did too," for example, shows ellipsis in the second conjunct. The program must infer the missing information, and display it in the graph, as in figure 5:

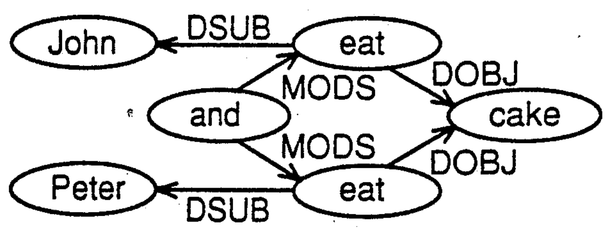


Figure 5. Graph (including reference for VP ellipsis) for the sentence, "John ate the cake and Peter did too."

16.3.1 NP (pronoun)-anaphora

The assignment of referents to pronouns proceeds in two steps. First, relative pronouns within relative clauses are assigned a REF attribute that points to the head noun governing the clause. The simplest subcase of this assignment is displayed in figure 6:

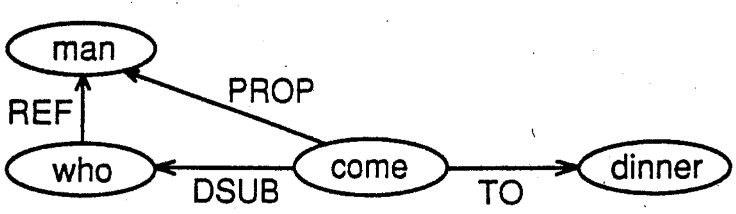


Figure 6. Graph showing REFerent for the relative pronoun "who" in the phrase "the man *who* came to dinner."

A slightly more complicated functional assignment is involved when right-shifted relatives are encountered. In the sentence, "The man came *who* was your friend," the REFerent of "who" is the subject of the main clause, "man." In the sentence "Which friend did you bring *who* likes chocolate?" the REFerent of "who" is the fronted object of the main clause, "which friend."

Second, after handling relative pronouns, all other pronouns are considered, within sentence boundaries. The following constraints are enforced:

1. *Agreement:* the NP must agree with the pronoun in number, person, and gender.
2. *Non-clausemates:* the NP cannot be an argument within the same clause as the pronoun, or else the pronoun must be reflexive.
3. *Reflexive* (the inverse of non-clausemates): a reflexive pronoun can have as its referent only an NP that is an argument within the same clause. (But this will have to be modified; see Zribi-Hertz 1989.)
4. *Command*: basically, the pronoun cannot command the NP or else the NP must come before the pronoun in the input string.

These constraints are applied to a pronoun-NP pair within the sentence. If all of the conditions succeed, the list attribute REF is created on the pronoun record, pointing to the NP. Then all other eligible NPs in the sentence are tested, and the ones that pass the constraints are added as members of the REF list.

The appearance of a noun as a member of a pronoun's REF list means only that this noun is a possible referent for the pronoun, within sentence boundaries. The notion of possible referent is important for two reasons. First, at this point the system has access only to the information carried within the analysis record; and this information is mostly of a limited syntactic nature. However, it is well known that referents cannot be assigned properly, in all situations, on the basis of syntactic information alone (Hobbs 1977). In case more NPs than one should pass the syntactic constraints, a choice will have to be made later in the processing, on the basis of background knowledge or broader contextual information. The REF list carries the candidates forward for future consideration. Given only the sentence "The cake was a surprise; John liked it," it is not possible to know whether "it" refers to the cake or to the surprise.

Second, every sentential assignment of reference can always be modified by extrasentential context. In the single sentence, "John talked while he ate," the only referent for "he" is "John." But in the sentence pair "John was not hungry, so Peter started eating by himself. John talked while he ate," the most likely referent for "he" is "Peter." This potential for modification includes even the assignment of referents to reflexives (Zribi-Hertz 1989).

16.3.2 VP-anaphora

The elaboration of VP-anaphora is done after pronoun referents are assigned. First the program identifies the nodes where VP elision has occurred. Then it constructs a list of propositions within the sentence, and passes these two arguments—node list and proposition list—to a sub-procedure.

The eligible nodes include, for example:

1. coordinate and subordinate clauses with VP-anaphora, e.g.,
   John ate some cake and Peter *did too*. (…*so did* Peter)
   John ate his cake, although Peter *didn't*.
2. "do so" anaphora, e.g., "If asked to *do so*, they will comply."
3. S-pronominalization with "it" or "so," e.g.,
   They want him, but John doesn't know *it*.
   Because no one told them *so*, they had to discover the fact.
4. bare infinitivals, e.g., "Those who can afford *to*, live in large houses."

Gapping, which involves the total absence of any verbal element, is not yet handled: "John ate a cookie and Peter, some cake."

For each eligible node, each eligible proposition is considered, and subjected to several constraints. These constraints bear some interesting resemblances to the constraints enforced for pronoun anaphora. By exercising them, the program identifies the most likely antecedent to fill the empty VP node. It then fills that node by cycling through the semantic structure of the antecedent. For each of the antecedent's semantic attributes ("semantic attributes" are those listed in section 16.2 above), if no corresponding attribute exists in the empty node, the attribute of the antecedent is copied into the empty node. Consider the sentence "If asked to do so, they will comply." First PEGASUS supplies the missing deep subject for "asked," and then determines that "do so" involves an anaphoric reference to the main predicate, "They will comply." The resulting filled structure is shown as a graph in figure 7. From this graph paraphrases can be generated, such as "They will comply, if someone ('xx') asks them to comply."

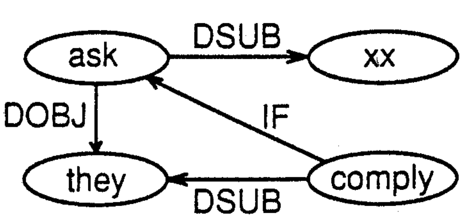


Figure 7. Graph for the sentence, "If asked to do so, they will comply."

16.4 Comparison with other approaches

16.4.1 Empty categories and functional uncertainty

Of particular interest here is the fact that argument structures for English are being defined *after* the syntactic parse is complete. PEGASUS operates on the output of the analysis grammar, by using a post-syntactic, procedural processor, to fill in all missing arguments and to produce the completed predicate-argument picture. In computational systems motivated by current linguistic theories, however, argument structures are computed during the operation of the initial analysis grammar.

The problem of filling predicate-argument structures—and, in particular, of correctly assigning long-distance dependencies as in "Who did Mary think that Peter said that John kissed?"—is well known in the literature of linguistics and computational linguistics. Two chief methods have been described for accomplishing this:

- the "empty category" (EC) approach;
- the "functional uncertainty" (FU) approach.

The EC approach is advocated, for instance, by linguists of the Government and Binding (GB) and Generalized Phrase Structure Grammar (GPSG) schools (Sells 1985). This approach uses parse structures that contain empty slots in the places where the dislocated long-distance constituents might be, if the sentence were in its most neutral form. For example, the sentence "Alice, Peter said that John kissed" is supposed to have an *empty category*, or *trace*, right after the verb "kissed," because that is where the noun phrase "Alice" would go if the sentence were in its default, or neutral, declarative form. Computational grammars that are built along these lines actually specify empty slots in their parse trees (see chapter 3, section 3.3.2).

The FU approach is advocated by Lexical Functional Grammar (LFG). This approach bases its solution not on empty slots in a parse tree, but rather on the incremental evaluation of the characteristics of all the verbs ("characteristics" chiefly refers to the required number and kind of arguments that a verb must have), from left to right in a sentence, in order to find out where the displaced constituent best fits. A new notational device has been added to the LFG formalism, for the purpose of computing the properly filled argument structures (Kaplan and Zaenen 1987). Computational grammars that are built along these lines use this device, in their grammar rules, to specify where the missing argument should be assigned.

The present method differs from both of these approaches. It differs from the EC approach in that:

1. It does not use empty categories or traces of any kind.
2. It does not rely so heavily on the constituent, or tree, structure, but uses both constituent and functional information, along with any other kind of information available from the dictionary and the syntactic parse (e.g., morphology, selectional and subcategorization features, etc.).

It differs from the FU approach in that:

1. It does not use any special notational devices other than those already provided by the programming language used.
2. It does not rely so completely on characteristics of verbs in the sentence (functional information), but uses both constituent and functional information, along with all other available information.

It differs from *both* of the above approaches in that it performs the argument-filling *after* the syntactic parse has been completed. It uses a post-processor, and not the initial syntactic grammar itself, to manipulate the full range of attribute-value information, in order to derive the most reasonable argument structure.

16.4.2 Why use a post-processor?

At this stage of the development of NLP, there seems to be no disagreement that a convenient place to do anaphora resolution is in a post-syntactic processor. After all, a possible NP or VP antecedent might be found in any part of the sentence, so it makes sense to assemble all parts, and their likely relationships, before starting the search. But there is definitely no agreement on the idea that deep arguments should be assigned post-syntactically. In fact, the optimal strategy for building an argument structure might conceivably vary from language to language. For English, there are some advantages to the post-processing approach.

Using a bottom-up parser, as in the PLNLP system (and many other computational analysis systems), work may have to be re-done in the syntax anyway, if the argument structures were assigned during that initial syntactic parse. Sentences with topicalization illustrate this fact. Consider "This husband, Peter said that the preacher gave Mary." "Mary" may be construed as the deep object of "give" in all partial parses, right up until the final moment when the topicalized NP, "this husband," is added; then "Mary" would have to be changed to DIND, and the DOBJ attribute of "give" would have to be set to point to "this husband." Another type of complication may occur in sentences like:

Who did John kiss who loves him?
Who did John write who loves him?

Suppose that we parse these sentences into two main parts: a matrix clause ("Who did John kiss/write") and a relative ("who loves him"). "Who" may be assigned as DOBJ of "kiss" and DIND of "write" (as discussed in section 16.2). Then, to what would the relative clause be linked? Would we search for some deep argument (DOBJ or DIND or something else) in the matrix? Or would we link the relative clause to the fronted "Who" and then re-assign this modified NP to be DOBJ of "kiss" and DIND of "write"? Either solution would require extra effort. It may be easier to postpone building the semantic structure until after the syntactic pieces have been assembled.

For some languages, e.g., Italian (see chapter 15), many argument structures cannot be assigned correctly until after background knowledge has been added to the analysis. This should not be done during the initial syntactic sketch; it defeats the purpose of the sketch, and damages the potential for broad coverage. That being the case (for these languages), since some argument assignments will have to be made after the syntax anyway, it makes sense to group them all after the syntax.

No single one of these complications, by itself, is decisive. But taken together, these (and other similar) situations suggest the desirability of assigning deep functional roles in a post-syntactic component such as PEGASUS.

16.5 Conclusion

Within the framework of the PLNLP analysis system, PEGASUS is significant because it makes the definitive transition from syntax to semantics, where "semantics" is understood to involve, minimally, the definition of case frames or thematic roles (i.e., predicate-argument relations). Within the general framework of NLP, PEGASUS is interesting because it suggests a way of computing argument structures, in a post-syntactic processing stage, that is different from methods being used in other current analysis systems.

Chapter 21

Normalization of Semantic Graphs

Frédérique Segond

Abstract

At the level of semantic relations, we are interested in finding the semantic links hidden in the syntax of a sentence. This involves, among other things, normalizing semantic structures across a wide range of paraphrases. The goal is achieved by taking the output of the preceding analysis components and modifying it with a "concept grammar," written in PLNLP. The rules of this grammar are similar in form to the rules of preceding components; but they operate on different aspects of the common information structure, analyzing the relations between nodes in the sentence graph, and normalizing semantic structures and lexical relationships in a variety of syntactic domains, without losing access to the surface syntactic differences. This chapter shows how, starting from the argument structure output from PEGASUS, the concept grammar produces semantic graphs that preserve the broad-coverage, broad-domain characteristics of the entire system.

This chapter is excerpted from Segond and Jensen 1992.

21.1 Introduction

Semantic relations may be represented by a graph. The nodes of the graph contain words; but, since these are linked with dictionary definitions, synonyms, and other related words, it is possible to say that these nodes represent concepts.[1] It is the job of the concept grammar to construct a well-motivated network in which semantic relations are properly drawn among concept nodes. This grammar consists of PLNLP procedures that perform certain operations on a graph under certain conditions. The arcs of the graph are labeled with relation names, which are derived in a principled fashion from the combined syntax and semantics of the input text.[2]

In order to do this job, one of the important problems that has to be addressed is the problem of showing equivalences between paraphrases. This problem is first approached by PEGASUS (chapter 16), where, for example, both active and passive forms of a clause are provided with the same argument structure. The work is continued by the concept grammar, and expanded to handle a much wider set of paraphrase situations. The basic intuition remains the same, however: different sentences that have essentially the same truth-value will have the same semantic graph. And the same principle of accountability applies here as there: the system will always have access to the original surface syntactic variability, so that no nuances of meaning need ever be lost.

As an example, all of the following sentences have the same essential meaning, and therefore should be associated with the same semantic graph:

(1) (a) There is a blue block.
(b) The block is blue.
(c) The block is a blue block.
(d) The block is a blue one.

These are not classical syntactic variants, like active and passive; but they are variants of the same semantic facts: a block exists, and it is blue.

The sentences are analyzed by the syntax (PEG) and by PEGASUS. (Because our descriptive sentences are purposely kept very simple, we do not need to use the reassignment and the sense disambiguation components.) The result is a graph for each sentence, corresponding to the basic arguments and adjuncts of that sentence. The concept grammar examines each sentence graph, checking

[1]See Sowa 1984 for an introduction to conceptual graph structures.
[2] There are some interesting parallels that can be made between the principles that guide the construction of these graphs, and the principles stated in Jackendoff 1983, for defining major phrasal constituents (S, NP, VP, etc.) as concept nodes that belong to major "ontological" categories.

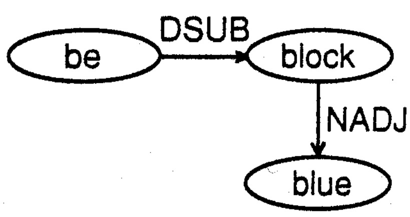
for certain configurations that signal the presence of common underlying conceptual categories. Here is where the remaining variations will be normalized.

The operation of the concept grammar can be compared to the operation of a syntactic grammar: syntax takes words and phrases, and links them, via common morpho-syntactic relationships, into a structural whole; the concept grammar takes arguments and adjuncts, and links them, via common semantic relationships, into a conceptual whole. Syntax works with syntactic category labels; the concept grammar works with semantic arc labels.

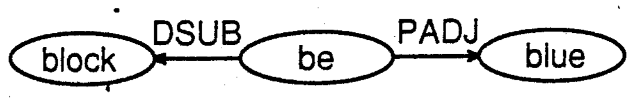
21.2 Normalizing the "block" sentences

Consider the four sentences given in (1). The argument and adjunct structures (sentential graphs) provided by PEGASUS for these sentences, and shown in figure 1, use just four semantic arc labels (see chapter 16):[3]

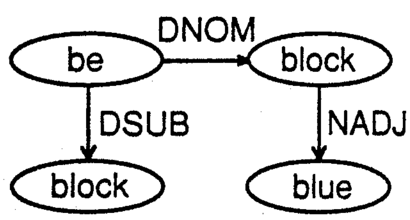
DSUB: deep subject
PADJ: predicate adjective
NADJ: adjective modifying a noun
DNOM: deep predicate nominative



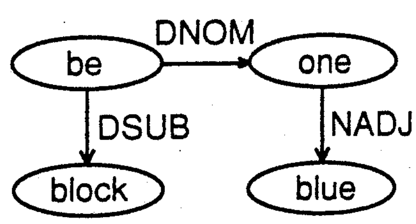
"There is a blue block" (1a)



"The block is blue" (1b)



"The block is a blue block" (1c)



"The block is a blue one" (1d)

Figure 1. Sentential graphs for the sentences in (1)

These four sentential graphs are quite different; but, since the sentences have the same meaning, there should be just one conceptual graph for all of them:

[3] Although only the lemmas are displayed in the graph nodes, the underlying record structure keeps access to all syntactic details, such as determiners, tense, etc.

Figure 2. Canonical semantic graph for the sentences in (1)

This is a case of paraphrase that requires normalization. In order to achieve it, first we delete the node "be" in all graphs. The English copula "be" generally carries very little semantic weight.

RULE 1: Delete the copula "be."

Second, if an adjective carries a lexical feature that marks it as a "color" word, then we change the arc label NADJ to the label COLOR. The effect is to change the name of the relation between the noun and the adjective.

RULE 2: Change NADJ from node with "color" word to COLOR.

To achieve the desired semantic graph for sentence (1a), we apply Rule 1 and Rule 2, deleting the node "be" and changing the name of the relation between the node "block" and the adjective "blue."

When the predicate is an adjective (PADJ), there is, in the argument structure, no direct relation between the subject (DSUB) and the adjective (PADJ). Both of them are attributes of the node "be." In this case, we create a new relation, NADJ, between the subject and the adjective, and delete the relation PADJ. (We will deal later with the difference between predicative (PADJ) and attributive (NADJ) adjectives.)

RULE 3: Create NADJ arc between subject and predicate adjective.

Once this new arc is created, rules 1 and 2 will recognize that the adjective is a "color" word, change the name of the relation NADJ to COLOR, and delete the node "be." These operations will turn the sentential graph for (1b) into the desired semantic graph in figure 2.

When the predicate is a noun or a noun phrase (DNOM), as in sentences (1c) and (1d), we have to ask if that predicate nominative is the same term as the subject (or is an equivalent empty anaphoric term, like "one"), or if it is different from the subject, and not empty. In the first case we "unify" the subject and the predicate NPs. All the nodes which point to the first are made to point to the second, and vice versa. Once this is done, the problems of the color adjective and of the empty copula are automatically handled by existing rules, and the sentential graphs for (1c) and (1d) are transformed into the canonical graph in figure 2.

RULE 4: Unify subject and predicate under appropriate conditions.

In the second case, when there is a DNOM that is different from the subject NP, we create a new relation between the subject and the predicate. In the simplest case, we give this relation the IS-A label (but see chapter 10, section 10.2, for complications):

RULE 5: Create IS-A link under appropriate conditions.

Hence the sentence "The block is an object" has the following semantic graph:

Figure 3. Semantic graph for "The block is an object"

The reader should not conclude from the previous examples that dealing with paraphrases requires a lot of ad hoc solutions. On the contrary, the rules (or procedures) of the concept grammar are general in nature. They identify and represent typical semantic relations in a formal way. A syntactic grammar does the same thing, but at a different level of structure. The concept grammar tries to catch what might be called "the semantics of the syntax." These operations are straightforward, just as the operations that build constituent structure in a syntactic grammar are straightforward. But this simplicity should not obscure the elegance of what is going on here. With minimal effort, using easily accessible parse information, we are automating the creation of a conceptual structure. This conceptual structure will ultimately have a high degree of abstractness and generality.

21.3 Locative prepositional phrases

Consider the following set of sentences (and cf. sentences 12–27 in the appendix), which should all have the same semantic graph (figure 4):

(2) (a) There is a blue block on the red block.
(b) There is a red block under the blue block.
(c) The blue block is on the red block.
(d) The red block is under the blue block.

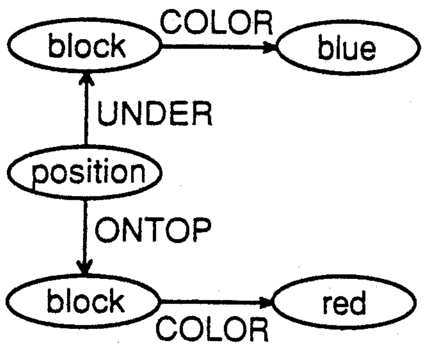


Figure 4. Canonical semantic graph for the locative sentences in (2)

Note the graph node labeled "position." This word was never used in the paraphrase sentences, but the concept was implicit in all of them. (The link between preposition names and the word/concept "position" can be validated in dictionaries and thesauri.) One interesting and significant result of setting out to normalize these paraphrases is the emergence of what might be called the essential meaning of the expressions, namely, a statement of the relative position of two objects. In this fashion, the writing of a concept grammar results naturally, and pragmatically, in the emergence of terms that we might want to consider as "semantic primitives."

It should be emphasized, however, that we are not committed beforehand to any basic conceptual or semantic primitives. In this example, the relations ONTOP and UNDER appear in the canonical graph of the sentence, but this is just for purposes of the present exposition. What we are interested in is to establish an appropriate link between the two blocks. Instead of ONTOP and UNDER we could have ABOVE (or ON) and BELOW, etc.

It is not necessary to discuss the treatment of each of the paraphrases. The first sentence in (2) will serve as an example. Figure 5 shows its sentential graph.

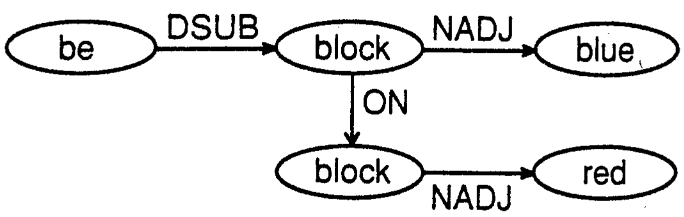


Figure 5. Sentential graph for "There is a blue block on the red block" (2a)

What we want to do is to link the deep subject ("blue block") with the object of the preposition ("red block") by using the relation names ONTOP and UNDER, which spring from the concept POSITION. We delete the copula "be," and create the new node POSITION, motivated by dictionary definitions for locative prepositions. Then we add two attributes, UNDER and ONTOP, to this node (pointing respectively to the subject and the noun phrase object of the preposition), and delete the attribute ON in the list of attributes of the subject. Notice that if the sentence read "above" instead of "on," the treatment would be the same.

Of course, this does not mean that looking at the syntactic relations between words is enough; the semantics of the words themselves are also important. For instance, the kind of relation involved between a subject NP and the NP object of a PP in the case of a locative prepositional phrase (e.g., the cat is *in* the garden, the cat is *under* the table), is not the same as the one involved with the PP which is a part of the sentence "The cat is *in* love." But still, in all these three sentences, what we are interested in is building the relation between "the cat" and the NP object of the PP (garden, table, love). Giving a name to the relation (and, for that purpose, knowing that love is a concept, garden is a place, and table is an object) is the task of the sense disambiguation component (chapter 19), which consults dictionary definitions to find the necessary semantic information.

21.4 Relative clauses

One way of combining propositions (the block is blue, is on the table, etc.) into one sentence is to use a relative clause. We can say:

(3) (a) The block that is blue is on the table.
(b) On the table is the block that is blue.
(c) The block, which is on the table, is blue.

Figure 6 shows the sentential graph for (3a). The attribute PROP points to the semantic structure of the relative clause "that is blue," and the attribute REF identifies the referent of the relative pronoun "that":

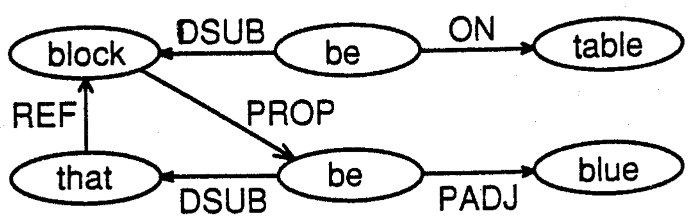


Figure 6. Sentential graph for "The block that is blue is on the table" (3a)

In the sentences of (3), we want to relate the deep subjects of the relative clauses with their predicates. All we have to do, in this case, is to unify the DSUB of the PROP with the REF of the DSUB of the PROP, deleting the REF attribute. The result is a record, pointed to by PROP, which has a DSUB identical to the DSUB of the whole sentence, and therefore possesses both the attributes that it gains from the relative clause, and the attributes of the DSUB of the whole sentence. Now the system is able to handle recursively all the other problems (copula, predicate adjective, and spatial prepositional relationships), and we obtain the same graph as is obtained for sentences such as "The blue block is on the table" or "There is a blue block on the table":

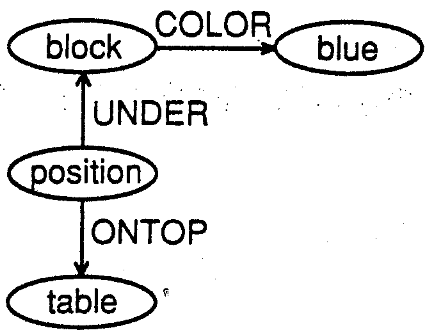


Figure 7. Canonical semantic graph for the sentences in (3)

21.5 Toward the discourse model

Our work also involves normalizing across sentence boundaries. For instance, from (4a–b):

(4) (a) The blue block is on the red block.
(b) The red block is on the black block.

we want to be able to infer (4c–d):

(4) (c) The blue block is above the black block.
(d) The black block is below the blue block.

Inference across sentence boundaries does not differ, in essence, from inference within a single sentence; after all, two sentences may become one sentence, under coordination:

(4) (a AND b) The blue block is on the red block AND the red block is on the black block.

From an implementation point of view, the strategy is the same. We consider all nodes called "position." There is one such node in the graph for (4a), and another in the graph for (4b). We look at the records for both "position" nodes and obtain two lists: one, a list of all ONTOP attributes; and the other, a list of all UNDER attributes. We look at the intersection of those lists. If they have an element in common (for instance, in the previous example, "red block" will appear in both of them), then we know that we can infer the graph in figure 8:

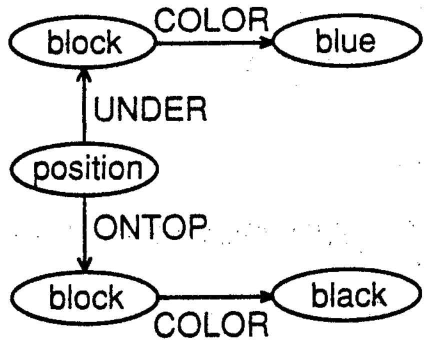


Figure 8. Inferential graph for (4c–d)

Figure 8 displays only the inferences in (4c–d), derived from (4a–b). But the system does not lose access to information about the existence and placement of the red block mentioned in (4a–b).

All the examples given in this chapter involve sentences with the verb "be." "Be" and other state verbs comprise a complicated and interesting class. They accept a lot of different constructions (adjectival predicates, nominal predicates, prepositional phrase complements, etc.), and provide a convenient and convincing field for preliminary investigations. At the same time, much of the work done for state verbs (coordination, PP relationships, etc.) can be applied to other verb classes.

21.6 Conclusion

Dealing with the above phenomena is not the same as dealing with the whole of natural language. However, we have tried to avoid specific or ad hoc solutions. The rules of the concept grammar are generic in nature. They express semantic facts about English (and, in some cases, about language in general), just as a morpho-syntactic grammar expresses syntactic facts about English. Therefore they are in no way restricted to a semantic subdomain.

We hope to have made one substantial contribution in this chapter: to show the birth of a conceptual grammar, which receives syntactic and semantic information from earlier stages of the system, and automatically provides a grammatical foundation for the next stage, discourse. We have dealt with some linguistic problems, including different kinds of paraphrases. We have also suggested methods for handling logical properties of natural language, such as the spatial properties of prepositions. (See Segond and Jensen 1991 for additional constructions handled by the concept grammar. The appendix to this chapter gives example sentences of the sort that are normalized by the initial version of the grammar.)

This structure of very general relations is one of the steps leading to an ideal semantic representation of sentences. It provides a universal representation, independent from the surface structure, but without losing the information contained in the surface structure. (See Fagan 1990 for a discussion of related ideas.)

Another contribution of this chapter is to illustrate some advantages of this approach to an articulated architecture for a natural language understanding system. The architecture provides both modularity and integration of NLP tasks, and allows for a smooth flow from syntax through semantics to discourse. Starting with an initial syntactic sketch, we obtain a conceptual graph step by step, without adding a lot of hand-coded semantic information in the lexicon.

The next step is to join the normalized sentence graphs that are the output of the semantic normalization component, and build a formal model of those discourse chunks which, in written text, are typically called paragraphs (chapter 22).

Appendix to Chapter 21: Some sentences handled by the concept grammar

1. There is a blue block.
2. The block is blue.
3. The block is a blue block.
4. The block is a blue one.
5. The block is an object.
6. The blue block is a nice block.
7. The blue block is a nice one.
8. The blue block is a nice object.
9. The black and blue block is small.
10. The big block is blue and the small block is red.
11. The big block and the small block are blue.
12. There is a blue block on the red block.
13. There is a red block under the blue block.
14. The blue block is on the red block.
15. The red block is under the blue block.
16. There is a blue block on the red one.
17. There is a red block under the blue one.
18. The blue block is on the red one.
19. The red block is under the blue one.
20. There is a blue block above the red block.
21. There is a red block below the blue block.
22. The blue block is above the red block.
23. The red block is below the blue block.
24. There is a blue block above the red one.
25. There is a red block below the blue one.
26. The blue block is above the red one.
27. The red block is below the blue one.
28. The big red and blue block is on the small green one.
29. The block that is blue is on the table.
30. There is a block that is blue.
31. The helmet, which is a red piece, is round.
32. The block, which is on the table, is blue.
33. The round helmet is a red piece.
34. There is a round helmet which is a red piece.
35. The blocks are blue and red.
36. There are blue and red blocks.
37. There are eight blue blocks.
38. There are two red blocks.
39. The two red blocks are the mufflers.
40. One of the eight blue blocks is the steering wheel.

41. There are four wheels.
42. There are two pairs of wheels.
43. There are blue blocks.
44. There are red blocks.
45. There is a black steering wheel.
46. The steering wheel is black.
47. There are two identical blue blocks.
48. The biggest block is blue.
49. The red blocks are on a blue one.
50. The steering wheel is above a pair of wheels.
51. The mufflers are on a pair of wheels.
52. The number that is on one blue block is 3.
53. The car is red and blue.
54. There are three pieces for the driver.
55. The red helmet is on the head of the driver.
56. The head and the body are on the driver's legs.
57. The driver who has a yellow face is on the car.
58. A blue block connects two red blocks.
59. A blue black connects eight red blocks.
60. A blue block connects ten blocks.
61. A blue block connects eight blue blocks and two red blocks.
62. The number that is on the blue block is white.
63. The driver is on the car.

Garside, Roger et al., *The Computational Analysis of English: A Corpus-Based Approach*, Longman, pp. 97-109, 1987.

same category in deep structure and different categories in surface structure, or that categories are not atomic symbols but sets of features which may display partial resemblances with other sets. But this would not affect the claim that subjective analytic styles are irrelevant.

The generativist's view may be correct; clearly, the fact that we made decisions about lumping or splitting "core" constructions such as *for-to* clauses in a partly subjective fashion is no argument at all against the possibility that such questions can be resolved in terms of objective evidence that we were not perceptive enough to notice. What might be an argument, though, is that questions of just the same type arose equally often in connexion with "peripheral" grammatical issues where the generative writ scarcely runs.

For instance, should a Harvard-style reference, e.g. the *(1960)* in:

Knighting and Hinds (1960) showed .. (J10 94)

be lumped under the hypertag Si, along with bracketed interpolations such as *(iv)* or *(it is claimed)*? The fact of enclosure in brackets is an important formal similarity, and the slots where such sequences can occur overlap heavily; but there are also some consistent differences. In the event we "split" in this case, and chose to link Harvard-style references with the preceding names as a special class of appositional elements. But in an area like this it is difficult to believe that one choice would be an accurate reflection of the psychologically-real linguistic competence determined by our innate language-acquisition device, with other choices simply "incorrect". Once one becomes used to the idea that only judicious weighing of the balance of convenience can settle issues of this kind, it seems implausible that comparable questions have a qualitatively different status when one happens to be dealing with an aspect of grammar falling within the generativists' purview. Where is the borderline, and how would one know that one had crossed it?

It may be that theorists of linguistic competence have answers for such questions. My purpose here is not to claim that their style of language description is necessarily mistaken in its own terms. What seems undeniable, though, is that, for the practical purpose of building natural-language processing systems which achieve usable results with authentic input, the approach to linguistic description associated with the phrase "linguistic competence" is irrelevant, and will remain so at least for the foreseeable future.

A sociologist may spin webs of deep theory by focusing on just those aspects of human behaviour which express individuals' class affiliation, or their family relationships. A judge, on the other hand, has to confront human behaviour in all its rich and unpredictable diversity, and the legal structure he helps to build is a developing filigree of detailed rules rather than a geometrical construct of elegant, eternal axioms. In computational lingistics, likewise, the researcher who aims to cope successfully with real-life data cannot pick and choose just theoretically-interesting bits of language. His approach must be catholic and pragmatic, not absolutist. But to be pragmatic is not to be unprincipled. The opposite of God's-truth need not be "hocus-pocus"; it may be jurisprudence.

CHAPTER 8

Text input and pre-processing: Dealing with the orthographic form of texts

Barbara Booth

1. Introduction

The problem of analysing the orthographic features of texts is an important but neglected area in natural-language text processing, and current enhancement work to CLAWS has brought this problem area very much into focus. Previous chapters have emphasized that the aim of our work is the automatic analysis of large quantities of unrestricted text. But if we are to make credible claims about the practical usefulness of our systems it is clearly essential that all the orthographic features of texts, especially such ambiguous features as capitalization, full stops, and abbreviations, should be as far as possible correctly processed without manual intervention. CLAWS1 was developed specifically for the tagging of the LOB Corpus, from which the orthographic ambiguities had been virtually eliminated at the coding stage, with a few remaining problems forestalled during manual pre-editing. With CLAWS2 we are aiming for a more general and ambitious system, which will run over any machine-readable text without recoding or pre-editing. This chapter explores the problems of dealing automatically with the orthographic form of texts, first, by describing how the automatic process in CLAWS1 used information made available by manual coding and pre-editing, and secondly by examining how the process was elaborated so that CLAWS2 could function from the text alone, once this manual intervention was eliminated.

2. The LOB orthographic coding scheme

The anticipated uses of the LOB Corpus influenced the format in which the Corpus was held, as well as the initial selection of material. The "guiding principle" was to produce "a faithful representation of the text with as little loss of information as possible" (Johansson *et al.* 1978: 7).

The compositors of the original printed texts were able to exploit the options of a very large symbol system: the letters of the Roman alphabet in a variety of sizes and styles; alternative alphabets and diacritic markings for non-English language material; a vast assortment of symbols for mathematical data or information from other specialized fields.

But computer character sets are far more restricted in size. The LOB editors limited themselves to the 95 printable symbols of the international standard 128-member ASCII character set. So only a certain number of the symbols in the original text could be represented on a one-for-one basis in the Corpus coding scheme. In addition to "simple" coding symbols, where one character in the Corpus represented a single, usually the same, character in the original text, there were also "compound" coding symbols, where strings of more than one character represented a single character in the original. For example, a percent sign, %, in the Corpus represented the same character in the text, whereas the string *?22 was used to represent the square root sign, √. And because the character * was a prefix distinguishing compound coding symbols, when an asterisk actually occurred in a text it also had to be represented by a compound symbol, in this case */.

The Corpus was to provide a representative sample of written English at a particular moment in history, so the coders needed to distinguish data within the Corpus which might be unrepresentative or non-contemporary. It was particularly important to mark quotations, which might be taken from earlier texts, or might be direct speech, with the rather special linguistic features of dialogue. Although opening and closing inverted commas are usually differentiated in print, in computer character sets they are not normally distinguished. Opening and closing single and double inverted commas (', ', ", ") were all allocated unique compound coding symbols (*', **', *", **"), so that quotations could be unambiguously identified within the Corpus.

Besides such "designator" symbols, both simple and compound, corresponding to the symbols of the original text, the Corpus also contained "marker" symbols (see Johansson *et al.* 1978: 28). A marker symbol gave information about one or more of the preceding or following designators, and had no direct correlate in the original. Again, markers might be simple or compound. For instance, the compound marker symbol *1 indicated a typographical shift and showed that the following characters (up to the next typographical shift marker) appeared in italics in the original text. The simple symbol " indicated that there was an umlaut or diaeresis on the preceding character in the original. Again, because of the need to distinguish possible unrepresentative material, mathematical formulae and headings were delimited by markers, as were foreign-language excerpts, and all forms of non-standard English.

The coding scheme as described so far was an attempt to prevent loss of information when representing a printed text within the far more limited constraints of a machine-readable format. But the devisers of the scheme went beyond this in anticipating problems which might be met when processing printed data automatically. Despite the wide range of symbols available to the printer, some symbols are still conventionally used for more than one purpose. A full stop, for instance, may mark the end of a sentence, or an abbreviation, or both, or might appear as a decimal point within a numerical expression. Similarly, the same printed symbol is used for an apostrophe and for a closing single quotation mark, and an apostrophe may indicate a possessive case form (*John's father*) or a contraction (*don't*, *he's*). These local ambiguities would pose no problem for the competent reader, and would in all probability be unconsciously resolved. But for the linguistically incompetent machine they remain problematic.

Where the coders felt that an ambiguity would cause processing problems they again usually introduced separate coding symbols to resolve the ambiguity. Since the closing single quotation mark had been allocated a compound coding symbol, *', the simple symbol ' was reserved for apostrophe. Similarly, the compound symbol *— was allocated to the dash punctuation mark and the simple symbol — was reserved for hyphen or minus.

The full-stop symbol remained ambiguous, but all abbreviations were overtly marked as such in the Corpus, so the ambiguity could be automatically resolved from the context. For instance, the abbreviation for *inch* would be coded as \0in. and could be distinguished from the preposition *in* even at the end of a sentence.

Various other coding conventions were adopted in relation to spacing and ordering of punctuation, to standardize the Corpus, and to simplify later processing by researchers. For instance, where words in the original text were linked by a solidus (/), the symbol would be followed by a space in the Corpus to facilitate the automatic division of the Corpus into its minimal grammatical units. Similarly, the Corpus coders introduced a special marker symbol, ^, to indicate the start of each sentence, since the sentence is the maximal grammatical structure usually analysed by linguists.

A coding scheme which represents a single original symbol by a variable- length string of characters will necessarily involve more sophisticated processing than a one-for-one scheme at the character-handling level. But the recoding of ambiguous symbols, the inclusion of interpretative codings, and other explicit marking of covert information in the LOB Corpus, all helped to simplify the language processing tasks to be performed automatically.

3. CLAWS1: The automatic pre-editing phase

The first phase of the automatic processing was known as the "pre-edit" phase (see Chapter 3, pp. 33–4). At this stage, the text was segmented into the units which were to be tagged, and at the same time various other amendments were made to simplify the later tagging procedures.

The conventions adopted for spacing and ordering of punctuation made the automatic segmentation of the text relatively straightforward. Each sequence of characters delimited by spaces was treated as a word, and any punctuation marks at the start or end of the word were stripped off. A separate line was generated in the "verticalized" output text for each word or punctuation symbol, a line which would eventually also hold the correct tag. In general, because of the coding scheme, the recognition of punctuation was unproblematic, and it was decided that punctuation tagging could also be done at this early verticalization stage, each line created for a punctuation symbol being generated with an unambiguous tag. A full stop was treated as a punctuation symbol unless it fell within the scope of an abbreviation marker or followed a digit. The special marker for start of sentence was also put on a separate line, but recoded as a line of dashes so that sentence divisions would be clearly visible in the output text (see the example on p. 32).

Because the later tagging procedures would process the "verticalized" text output from this procedure with each word on its own separate line, marker symbols in the Corpus with a scope of more than one word were recoded as flags on every word within their scope. So, for instance, the delimiters for foreign-language excerpts were removed, and the line generated for each word within the excerpt was written with the character F in a certain column to indicate a foreign word. Abbreviations were flagged in another column, but the abbreviation marker symbol was left on the front of each abbreviated form and the final full stop was removed, to simplify lexical look-up at the next stage. Besides generating flags for words within their scope, certain other marker symbols – heading delimiters, for instance – were also recoded as start-of-sentence lines; and where there was no terminating punctuation before such markers, a full-stop record was inserted in the output text, to standardize sentence divisions in the verticalized Corpus. Because the automatic pre-editing involved this recognition and recoding of marker symbols, any tagging which could be carried out by reference to the marker symbols alone was also done at this verticalization stage. Thus, foreign words and formulae explicitly marked in the text were tagged as &FW and &FO respectively.

The segmentation task also involved the recognition of certain contracted forms. The automatic procedure looked for the words *I'm* and *cannot* and forms ending *'re*, *'ve*, *'d*, *'ll*, or *n't*. These forms were split into two separate lines and were flagged to show that they were originally part of the same orthographic unit.

The other major amendment made to the text at this automatic pre-editing phase was the conversion of letters from upper to lower case in certain contexts. The procedure assumed that, where a word occurred with a capital letter immediately after a start-of-sentence marker, the capitalization was irrelevant for tagging purposes, and changed the capital to its lower-case equivalent. But, if a word started with a capital letter in the middle of a sentence, this was assumed to signal some sort of proper name, and the automatic pre-editor left the word unchanged because the word-initial capital expressed information which was needed for tagging. However, where a whole word appeared in upper case, the assumption was that this was for typographical emphasis only, and the whole word was changed to lower case.

4. CLAWS1: The manual pre-editing phase

Obviously there were exceptions to the kinds of general rules used by the automatic pre-editing procedure. Proper names might occur at the start of sentences as well as in mid-sentence, and in such cases the automatic pre-editor would throw away the word-initial capital, leaving the tagging procedures with insufficient information to make the correct tagging decision. In CLAWS1 it was thought necessary to have some additional manual pre-editing between the automatic pre- editing and the tagging procedures proper, to deal with problems like this.

In fact, the major task of the human pre-editor was dealing with the problem of capitalization. A list was produced of all the words in upper case or with word-initial capitals which had been automatically changed to lower case. The editor then checked these to see if any of the words were actually proper names, and, if so, the word-initial capital was restored. In the sentence *'Gaunt was compelled to return to England* .. (G01 21–2), for example, the first word was changed to *gaunt* by the automatic pre-editor and then restored as *Gaunt* by the manual editor, to ensure that it would be tagged as a proper noun rather than an adjective. Similarly, a list was also produced of all words with word-initial capitals which had *not* been changed automatically, and this was checked for any words which were *not* proper names. Consider the example .. *the handling of the *1Lady Chatterley's Lover *0case* .. (A11 163). After automatic pre-editing the typographical shift indicators would be removed, but the word-initial capitals of the title would be left unchanged. The manual editor would then flag each word of the title, and reduce *Lover* to lower case, so that it would then be tagged as a singular common noun, while *Lady* and *Chatterley's* would be tagged as titular noun and proper noun, respectively:

A11 163 070 Lady	A11 163 070 Lady	T
A11 163 080 Chatterley's	A11 163 080 Chatterley's	T
A11 163 090 Lover	A11 163 090 lover	T
(after automatic pre-editing)	(after manual pre-editing)	

This last example also shows up another failing of the automatic pre-editing procedure requiring manual intervention, this time in the area of text segmentation. Enclitic forms of *is* and *has*, as in *It's the spring* .. (E15 157) and .. *it's got to be* .. (A33 36), and enclitic forms of *us*, as in *Let's face it* .. (A19 213), all needed to be split off from their preceding words. But, in CLAWS1, genitive forms, like *Chatterley's* above, were tagged as a single grammatical unit. The automatic procedure had no way of distinguishing between these various uses of *'s* without access to grammatical information. So, in the automatic phase, all *'s* forms were left attached, and the manual pre-editor had to put the enclitic *is*, *has*, and *us* forms onto separate lines later.

Because the manual pre-editing phase already involved some very detailed checking of problematic data, some predictable failures of the automatic tagging system could also be detected at this stage, and a very small number of words (approximately 0.2%) were tagged by hand to prevent these failures. More details are given below.

5. CLAWS1: Tagging

Both the initial coding of the texts in LOB coding scheme format, and the amendments made to the texts during the pre-editing phase, allowed certain generalizations to be made which simplified the task of the automatic tagging procedures proper.

WORDTAG, the automatic tag assignment program, contained special procedures for dealing with words starting with a capital. These relied on the fact that, after pre-editing, only proper names or words habitually written with word-initial capital would remain capitalized in the verticalized text. The lexicon contained both lower-case entries and words with initial capital; for example, there were entries both for *may*, tagged as modal verb, and for *May*, tagged as adverbial noun. At tag assignment, a word would be looked up in the form in which it occurred in the text, and, if found, would receive the appropriate set of tags. If the word was not in the lexicon, a special suffixlist was consulted for words with initial capitals; and, if all else failed, capitalized words would be tagged by default as proper nouns. Approximately 35 000 words were tagged as proper nouns by this default procedure, and of these only 700 had to be corrected during post-editing. Virtually all of these errors were because the manual pre-editor had failed to reduce capitalized words to lower case where necessary.

Since the manual pre-editing phase had involved looking through all capitalized words in the Corpus, any which would predictably be tagged incorrectly by WORDTAG were manually tagged at the same time. So about 300 plural proper nouns were manually tagged, as were about 200 proper names which would not correctly drop through to the default procedure. For example, *May* in *Dora May rose to the bait* (L05 127), and *Cyprian* in .. *Chief of the Zulus, Cyprian Dinizulu* (A28 63), were both manually tagged as proper nouns, because otherwise they would have been tagged incorrectly from the lexicon and the special suffixlist, respectively.

Similarly, WORDTAG relied on explicit marking, inserted when the corpus was constructed, for tagging abbreviations. The lexicon contained both full word entries and entries for abbreviated forms with the same marker symbol used in the Corpus; there were entries both for *in*, tagged as a preposition or adverbial particle, and for *\0in*, tagged as a unit of measurement. All abbreviated forms also appeared in the lexicon without a final full stop. So abbreviations could be looked up exactly as they occurred in the verticalized text, with a preceding marker symbol but without any final full stop. This avoided the necessity of having two entries in the lexicon for, say, *\0Mr* and *\0Mr.*, since many common abbreviations are written both with and without final full stops. (Any forms which might occur with or without medial full stops had two entries, e.g. *\0m.p.h* and *\0mph*, both with identical sets of tags). Abbreviations not tagged from the lexicon were tagged as units of measurement by default. Again some manual tagging had been done at the pre-edit phase, for rare abbreviations which would predictably be tagged ~~in~~correctly. Compass bearings, for instance, like *\0ESE*, were manually tagged as adverbial nouns. This combination of explicit coding and manual pre-tagging of problematic cases again ensured a very high accuracy rate for abbreviated forms ~~under~~ CLAWS1.

The procedures in WORDTAG dealing with words ending in *'s* could also rely on the fact that all contracted forms of *is*, *has*, and *us* had been split off manually and any remaining *'s* words could be assumed to be genitive forms. Where the *'s* was a contraction of *us* or *has*, the form was tagged manually at the pre-editing stage. The entry *'s* itself appeared in the lexicon tagged as a contraction of *is*. WORDTAG tagged all other words ending in *'s* by ignoring the *'s* ending, assigning tags to the rest of the word through the usual procedures, then changing these tags to equivalent genitive tags, discarding any tags with no corresponding genitive form. For example, *board's* in *the board's share* would first be assigned the tags associated with *board* in the lexicon – singular common noun and verb – then the verb tag would be discarded, and the noun tag changed to the tag for singular common noun plus genitive. In the entire Corpus only 86 words ending in *'s* were mistagged, and again most of these errors were because the manual pre-editor had failed to deal correctly with some capitalized forms.

As explained above, before the texts were submitted to WORDTAG some tagging had already been done at the pre-editing stage. In fact, this amounted to just over 12% of the Corpus – 149 603 syntactic units. The vast majority of these were punctuation symbols, plus the small number of foreign words and formulae tagged by the automatic pre-editor, and the even smaller number of words tagged by the human pre-editor (approximately 2500). This unambiguous pre-tagging together with the generalizations made possible because of the explicit marking of abbreviated forms, and the manual checking of enclitics and capitalized forms, all helped to reduce the amount of tag disambiguation left for CHAINPROBS and must have improved the overall performance of CLAWS1.

6. The motivation for CLAWS2

Although those involved in the development of CLAWS1 were delighted with the accuracy rate they had achieved, they were well aware how time-consuming the initial coding and pre-editing phases had been and were interested to discover how much of this manual work could be avoided if the tagging system were to be made available to other users.

The LOB coding scheme was complex, with multi-character variable-length symbols introduced by a number of different prefixes. Some of the initial coding was valuable for CLAWS1 but other information was not used at all. Markings for typographical distinctions other than capitalization were removed before tagging took place, and heading delimiters were recoded as flags but otherwise ignored. When coding new texts, those unfamiliar with the tagging system would find it difficult to work out exactly what was necessary and what was not, and it would also be far easier for a new user to run the CLAWS system and then correct any errors at the post-editing stage, than to try to anticipate possible errors at the manual pre-editing stage. Moreover, many texts already exist in some sort of machine-readable format, and improvements to optical character readers over recent years have made it possible to input and archive printed texts fairly speedily. Who would want to use an automatic tagging system that required all such texts to be recoded?

It was therefore decided to enhance the tagging system to avoid the need for special coding and manual pre-editing. This entailed modifying the system to deal automatically with ambiguous punctuation marks, all capitalized words, abbreviations without explicit markers, and all the consequential problems of text segmentation.

7. CLAWS2: A simplified input format

The first stage of this enhancement work was the specification of a new simplified input format. The recognized elements of this new format are either characters from a standard computer character set, or special coding symbols from a limited set used to represent non-standard characters such as foreign orthographic symbols.

The standard characters are those in the ISO/ASCII 95 printable character set, omitting "_" underline, "‾" overline, "|" vertical bar, "\" backslash, "`" grave, and "^" circumflex: 89 characters in all. But it would be a relatively simple matter to alter the composition of this character set if required. The system assumes that all standard characters are used in the text as they would be in normal English orthography and punctuation, except that there is no distinction between opening and closing inverted commas. The character - might represent dash, hyphen, or minus, depending on context, and the character ' might represent apostrophe, or opening or closing single quotation mark.

All special coding symbols have the same format, one or two digits prefixed by *?. This allows 99 special symbols in all. Certain of these special symbols are fixed in meaning: those appearing in the lexicon, or those specifically tested within the tag assignment program. So the symbol *?2 should always be used to represent an acute accent, because it occurs in the lexicon in such entries as *applique*?2* and *cafe*?2*; and the symbols *?7 and *?8 should be used for single and double prime marks respectively because WORDTAG uses them to recognize some non-word strings. But other special coding symbols have been left unallocated for the user to assign as required.

With the new input format there are no "marker" symbols, only "designator" symbols. So sentence divisions and abbreviations are no longer explicitly marked in the text. The system has been amended to allow for this. Also there is no coding to pick out typographical shifts, non-English expressions, headings or comments. If this information is required in the output, it would be best to use the existing LOB coding scheme.

8. CLAWS2: The pre-editing phase

In CLAWS2 there is no manual pre-editing phase between the initial verticalization stage and the tagging procedures proper, and the tasks of the manual pre-editor are divided between a new automatic pre-editing program and a revised version of WORDTAG, with any residual errors left for correction at the post-editing stage.

Essentially, the new pre-editing program concerns itself with the verticalization task alone; that is, segmenting the text into syntactic units for tagging. The program puts each word and punctuation symbol on a separate line, and splits contracted forms over two lines, as did the CLAWS1 pre-editor. In addition it recognizes sentence-boundaries, which are no longer explicitly marked in the text. But the pre-editor does not change any capital letters to lower case, nor does it tag punctuation. All tagging decisions are now left to the tagging procedures proper. The problems of capitalization and abbreviations, except with respect to segmenting the text, are handled by WORDTAG; the other amendments made to the text by the CLAWS1 pre-editor are omitted because they operate on information no longer encoded in the simplified input format.

The recognition of sentence-boundaries is not entirely straightforward because it depends on distinguishing abbreviatory from sentence-terminating full stops, another problematic task now that abbreviations are unmarked. The program recognizes potential sentence-boundaries between words that end with sentence-terminating punctuation and words that start with a capital letter, optionally preceded by an inverted comma or bracket. Question marks and exclamation marks, again optionally followed by inverted comma or bracket, are always regarded as sentence-terminating, and so are full stops if followed by inverted comma, bracket, or more than one space. The problem arises when a word ends with a full stop followed by a single space. Certain strings that end with a full stop – abbreviations or initials, for instance – can occur in mid-sentence, and might be followed by other strings starting with a capital, such as proper nouns or initials. The sentence-boundary recognition procedure must try to exclude these cases.

At present a full stop is not considered to be sentence-terminating in the following cases:

(a) After a single capital letter (assumed to be an initial).

(b) After an abbreviated title (*e.g. Mr., Mrs., Dr.*) preceding a name. (These abbreviations are tagged NNSB in the lexicon.)

(c) After an abbreviation for a qualification or award following a name (*e.g. M.A., Ph.D., M.B.E.*) when the next string is also such an abbreviation. (These abbreviations are tagged NNSA in the lexicon.)

In all other circumstances, a full stop is considered to be sentence-terminating punctuation and, if the following word starts with a capital letter, a sentence-boundary is assumed to occur after the full stop. Obviously this procedure is not foolproof. It will fail to recognize a sentence boundary after a single capital letter, and will wrongly insert sentence-boundaries when abbreviations are followed by proper nouns in mid-sentence, if these abbreviations are not in the lexicon tagged NNSA or NNSB. But it is thought that such cases will be so rare that few errors will result.

As before, each string of characters preceded and followed by space is treated as a word, any punctuation stripped off, and a separate line generated for each word or punctuation mark. The following characters are always treated as punctuation at the start or end of character strings:

' " ( ) [ ] , ; : ? ! ...

But the character – is treated as a punctuation mark (i.e. dash) only when at the end of a character string, followed by a space; otherwise it is assumed to be a hyphen or a minus sign and is left as part of the word. Full stops are stripped off only if they occur at a recognized sentence-boundary point; all other full stops are treated as abbreviatory and left attached. A line of dashes is inserted in the verticalized file at the start of the text and at sentence-boundary points.

The new pre-editor handles contracted forms just like the CLAWS1 pre-editor, but it also deals with *'s* in the same way. It is left to the rest of the tagging procedures to decide whether the form is a genitive, reduced auxiliary, or enclitic *us*. Words ending in *s'* get the same treatment, with the character ' put on a separate line and flagged as detached. Because in the simplified input format there is now no coding distinction between apostrophe and single quotation mark, this also implies a tagging ambiguity later. (It may be necessary to try to distinguish quotation marks from apostrophes at verticalization – by pairing quotation marks, for instance – if it proves difficult to resolve this ambiguity later.)

9. CLAWS2: Tagging

CLAWS2 is designed to work with the new "Lancaster" wordtag set (see Appendix B), which makes more distinctions between different classes of words than the LOB tagset used by CLAWS1. (Because of this, tags quoted for various words in this chapter will not in general be identical to the tags quoted for the same words in other chapters.) In the Lancaster tagset, the tags for punctuation marks are generally identical to the punctuation marks themselves, except that both single and double inverted commas share the same tag. There is also now a separate tag for the genitive suffix, namely the symbol $. All punctuation is tagged unambiguously from the lexicon, except that when the new WORDTAG encounters ' on a separate line of text, flagged as detached from a preceding word ending in *s*, then WORDTAG overrides the unambiguous tagging in the lexicon and substitutes an ambiguity between genitive and quotation mark. The lexicon also contains an entry for *'s* with the tags VBZ VHZ $ PPIO2 (i.e. *is*, *has*, genitive, or *us*).

The problems of capitalization and abbreviation are also now handled at this stage. In CLAWS1, lower-case words, initial-capital words, and abbreviated forms all appeared in the lexicon, and, because of the unambiguous input format and manual

pre-editing procedures, a string of characters in the verticalized text could be matched exactly to the entries in the lexicon and receive the relevant set of tags. In CLAWS2, all capital letters are left unchanged in the verticalized text, whether at the start of a sentence or in mid-sentence; initial capital letters might or might not indicate some sort of proper name; abbreviations might or might not be marked with full stops. So the principle in CLAWS2 is to have a single entry in the lexicon or suffixlist for each string of letters, rather than having a separate entry for each possible orthographic variation on that string. But the single entry has a complete list of all possible tags the string could receive, each marked in some way to indicate the conditions under which it is appropriate. All tags compatible with the input orthographic form are selected at the tag-assignment stage and it is left to CHAINPROBS to choose between the options.

Entries in the lexicon are now entirely in lower case, but may have both the normal "lower case" set of tags, and also an "upper case" set, members of which are marked with a colon, and are relevant only if a word starts with a capital. So the lexicon entry:

may VM NPM1:

indicates that the form *may* always gets a modal verb tag (VM), but the singular month noun tag (NPM1) is assigned only to the form *May*. For some words, one or the other of these sets might be empty:

enquiry NN1
england NP1:

Similarly, the suffixes in the suffixlist may also have two sorts of tag. The suffixlist entry:

aul NN1 VV0 NP1:

indicates that the form *baul* would get singular noun and verb tags (NN1 VV0), but the proper noun tag (NP1) would apply only to capitalized forms like *Saul*.

WORDTAG reduces all words in the verticalized text entirely to lower case before matching them with the lexicon or suffixlist, but remembers whether or not the words occurred at the start of a sentence or with an initial capital. These factors determine which tags are assigned, and with what probability weightings.

So, for instance, if a word that was originally in lower case is found in the lexicon, then the word is assigned only the lower-case tags in the lexicon entry; if the word has an initial capital at the start of a sentence, it will be given the upper-case tags as well but with reduced probability weightings; and a word with an initial capital in mid-sentence will get the upper-case tags first with the lower-case tags added with reduced probabilities. The word *may* would receive three different tag assignments in these three different contexts:

I may go ..	VM
May I go ..	VM NPM1@
In May ..	NPM1 VM@

The "at" sign, @, is a low-probability marker – see p. 35.

If an initial-capital word has only lower-case tags in the lexicon then that set is used without any reduction in probability. For example, *enquiry* is always tagged NN1 even if capitalized. But if there are no lower-case tags in the lexicon for a lower case word, then WORDTAG behaves as though the word had not been found in the lexicon and tries out its other rules for tag assignment.

The same sort of procedure for selecting appropriate tags applies when words are tagged from the suffixlist; and the final default for initial-capital words is now singular proper noun, NP1, together with the lower case default tagset, NN1 VV0 JJ (singular common noun, verb, adjective). The relative probability of these tags will depend on whether the word occurred at the start or middle of a sentence.

Abbreviations are handled in much the same way. In the simplified input format abbreviations are no longer explicitly marked, and the revised system is not always certain whether a particular form is abbreviatory or not. Some abbreviations, *NATO* for example, will occur without any full stops. Some abbreviations written with final full stops will occur at the end of sentences, in which case it will not be clear to the automatic system whether the full stop is abbreviatory or not; if the new pre-editor has recognized a sentence boundary, the full stop will have been stripped off and treated as punctuation.

All entries now appear in the lexicon without any full stops, final or medial. So a form which might occur with or without medial full stops will now have only one entry; and a word and an abbreviated form which share the same letters (for example *in* and *in.*) will also share the same entry, with any tags relevant to abbreviatory forms marked with a preceding full-stop, thus:

in II RL .NNU

(II = preposition; RL = locative adverb; .NNU = abbreviated unit of measurement.)

WORDTAG removes all full stops from forms in the verticalized text before matching them against the lexicon, but remembers if the form occurred with medial or final full stops, in order to determine which tags are appropriate. Forms which occur without stops will be given all the tags which would be assigned by the upper/lower-case selection procedure described above, whether the tags are marked with a stop or not, but forms with stops will be given only those tags marked with a stop which are appropriate for their case. So *in* will receive all three tags, but *in.* will be assigned only the unit of measurement tag, NNU, unless it occurs at the end of a sentence and the full stop has been stripped off by the pre-editor. If a form with stops matches a lexicon entry which has no tags marked with a stop, then WORDTAG behaves as though the form had not been found in the lexicon, and assigns the default tags for abbreviated forms, namely NNU (unit of measurement) for lower-case forms and NP1 (singular proper noun) for forms with an initial capital.

Some entries in the lexicon also have tags which apply only to forms which are entirely in capitals; these tags are marked with a double colon. Again WORDTAG will throw away these tags when processing a lower-case word, but will give them the highest probability if processing a form all in capitals.

Once these changes are implemented it will be possible to run CLAWS without

the time-consuming and error-prone tasks of manual coding and pre-editing. There may be a small loss of accuracy when dealing automatically with orthographic ambiguities, but any tagging errors caused by the new procedures can be picked up in the later post-editing phase. Possibly other changes might be made in the future to improve the accuracy of the new system even further: the system might treat as a lower-case word any word with an initial capital that occurs in a cluster of capitalized words; or it could look to see whether a capitalized word at the start of a sentence occurred elsewhere in the text with a capital; or, as suggested above, it could disambiguate a final *s'* by pairing quotation marks. These approaches all involve processing the text in "chunks" rather than single word units. But even without these more sophisticated techniques to improve accuracy, we feel that CLAWS2 will still be a great improvement for an unfamiliar user who wishes to tag new texts.

If the canonical form in the entry retrieved matches the actual form of the input word, the entry is handed over to the calling function with the indication that there was an exact match; this will be the case in the first instance, where the input form is *McDonald* and the canonical form in the entry is also *McDonald.* In the second case, where the input form is *MCDONALD*, the entry is also returned to the calling function, but with an indication that there was a mismatch on the level of the canonical form. The calling function can then determine how entries marked as mismatches are to be handled: that is, whether or not they are to be rejected.

A more interesting example is the treatment of the word *Polish.* Lookup will successfully retrieve the entry stored under the normalized key *polish.* In accordance to the scheme laid out in (1) above, that entry will contain data for *polish* and *Polish.* Because only the records for the latter will produce an exact match between the input form *Polish* and the stored canonical form *Polish,* only those records will be marked as having matched exactly. The records for the lowercase form *polish* will be marked as not having matched exactly. The calling application can then determine whether to treat the two equally. For example, if the word occurs sentence-initially, the calling application may reasonably decide to give both words equal standing; if the word occurs in any other position in the sentence, the records for the lowercase form may be penalized or rejected altogether, or they may be given equal standing if the sentence is in title case.

Related Writings

Descriptions of the treatment of capitalization are not easy to find in the literature. However, a detailed description of the related algorithm and dictionary representation in the CLAWS automatic tagging system, developed at the University of Lancaster and the University of Leeds, can be found in *The Computational Analysis of English: A Corpus-Based Approach*, edited by Roger Garside, Geoffrey Leech and Geoffrey Sampson, and published by Longman in 1987. See pp. 106-108. Chapter 8 in that work contains an excellent overview of the problems introduced by capitalization.

The invention does have some points in common with the approach described in that work. Specifically, our key normalization scheme is very close to the one implemented in the CLAWS system. It remains to be determined, I suppose, whether the ways in which the invention differs would be obvious to someone skilled in the art!

Microsoft Products

The text critiquing application currently under development, code-named Noah, includes an implementation of this invention.

MAIL ROOM
DEC 6 1996
55
PAT. & TRADEMARK OFF.

Receipt

#5
C FR 1/12/97 mdf

PATENT

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Application Processing Division, Customer Correction Branch, Assistant Commissioner for Patents, 2011 Jefferson Davis Highway, Washington, DC 20231.

December 3, 1996
Date

Maurice J. Pirio
Maurice J. Pirio

RECEIVED
96 DEC 26 AM 11: 18
GROUP 240

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Art Unit	:	2412
Docket No.	:	661005.447
Date	:	December 3, 1996

Application Processing Division
Customer Correction Branch
Assistant Commissioner for Patents
2011 Jefferson Davis Highway
Washington, DC 20231

REQUEST FOR CORRECTED FILING RECEIPT

Sir:

Attached is a copy of the official filing receipt received from the PTO in the above-identified application, for which issuance of a corrected filing receipt is respectfully requested.

There is an error with respect to the following data, which is incorrectly entered. There is an error in filing date, which should read June 28, 1996.

The correction to be made has been marked in red on the copy of the enclosed filing receipt. Also enclosed is a copy of the Express Mail label and of the postcard submitted with the application.

The correction is not due to any error by applicants and no fee is due.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J Pirio

Maurice J. Pirio
Registration No. 33,273

MJP:sr
Enclosures:

Postcard
Copy of Filing Receipt
Copy of Express Mail Label
Copy of Postcard submitted with application

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

c:\clients\661005\447\file rcpt crctn

MJP

PTO-103X
(Rev. 8-95)

FILING RECEIPT

MAIL R
DEC 6 1996 55
PAT. & TRADEMARK OFF.

UNITED STA... S ... ARTMENT OF COMMERCE
Patent and Trademark Office
ASSISTANT SECRETARY AND COMMISSIONER
OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NUMBER	FILING DATE	GRP ART UNIT	FIL FEE REC'D	ATTORNEY DOCKET NO.	DRWGS	TOT CL	IND CL
08/674,610	06/22/96	2412	$1,544.00	661005.447	69	36	7

06/28/96

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

RECEIVED

NOV 2 6 1996

SEED & BERRY

Receipt is acknowledged of this nonprovisional Patent Application. It will be considered in its order and you will be notified as to the results of the examination. Be sure to provide the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION when inquiring about this application. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. If an error is noted on this Filing Receipt, please write to the Application Processing Division's Customer Correction Branch within 10 days of receipt. Please provide a copy of the Filing Receipt with the changes noted thereon.

Applicant(s)

GEORGE HEIDORN, BELLEVUE, WA; KAREN JENSEN, BELLEVUE, WA.

FOREIGN FILING LICENSE GRANTED 11/20/96

TITLE

METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

PRELIMINARY CLASS: 395

RECEIVED
96 DEC 26 AM 11:18
GROUP 240

(see reverse)

POST OFFICE TO ADDRESSEE

EXPRESS MAIL
UNITED STATES POSTAL SERVICE

EMS

EM417213544US

EM417213544US

ORIGIN (POSTAL USE ONLY)

INTERNATIONAL SHIPMENTS ONLY

Business Papers

Merchandise

Customs forms and commercial invoice may be required. See Pub. 273 and International Mail Manual.

98104

6/28/96

15 00

1500

1 6

15 00

No Delivery

SEE REVERSE SIDE FOR THE SERVICE GUARANTEE AND LIMITS ON THE INSURANCE COVERAGE

CUSTOMER COPY

CUSTOMER USE ONLY

METHOD OF PAYMENT:

Express Mail Corporate Acct. No. X980232

Federal Agency Acct. No. or Postal Service Acct. No.

WAIVER OF SIGNATURE (Domestic Only): I wish delivery to be made without obtaining the signature of the addressee or the addressee's agent (if in the judgment of the delivery employee, the article can be left in a secure location) and I authorize the delivery employee to sign that the shipment was delivered and understand that the signature of the delivery employee will constitute valid proof of delivery.

NO DELIVERY

WEEKEND HOLIDAY

Customer Signature

FROM: (PLEASE PRINT)

Robert W. Bergstrom
SEED AND BERRY
6300 COLUMBIA CENTER
701 FIFTH AVE
SEATTLE WA 98104-7092
661005.447 6/28/96 jlc

SEATTLE WA COLUMBIA CENTER STA. JUN 28 1996

TO: (PLEASE PRINT)

BOX PATENT APPLICATION
ASSISTANT COMMISSIONER FOR
PATENTS
WASHINGTON DC 20231-0001

FOR PICKUP OR TRACKING CALL 1-800-222-1811

LABEL 11-F 5/93

Express Mail No. EM417213544

661005.447

RWB:jlc

Box Patent Application
Assistant Commissioner for Patents
Washington, DC 20231

SENT: June 28, 1996
NEW APPLICATION

Filing Date Stamp

MAIL ROOM
JUN 28 1996

PLEASE STAMP WITH APPLICATION NO. AND RETURN

Kindly acknowledge receipt of the below-listed documents by placing your receiving stamp hereon and mailing:

Cert. of Mailing by Ex. Mail; Form PTO-1082 (+ copy); Spec., Claims, Abstract (41 pages); 69 Sheets of Informal Drawings (Figs. 1-59); in re: George Heidorn and Karen Jensen, for METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES.

Application Number

08/674610

NO FEE OR FORMAL PAPERS BEING SUBMITTED*

SEED AND BERRY LLP

OIPE JCT88 JAN 20 1998 PATENT & TRADEMARK OFFICE

Cm #17 #6 27/RA 2-2-98

PATENT

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Assistant Commissioner for Patents, 2011 Jefferson Davis Highway, Washington, DC 20231.

January 16, 1998 — Date

Maurice Pirio — Maurice Pirio

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Art Unit	:	2412
Docket No.	:	661005.447
Date	:	January 16, 1998

Assistant Commissioner for Patents
2011 Jefferson Davis Highway
Washington, DC 20231

INFORMATION DISCLOSURE STATEMENT

Sir:

In accordance with 37 C.F.R. §§ 1.56 and 1.97 through 1.98, applicants wish to make known to the Patent and Trademark Office the references set forth on the attached form PTO-1449 (copies of the cited references, as required under 37 C.F.R. § 1.98, are enclosed). Although the aforesaid references are made known to the Patent and Trademark Office in compliance with applicants' duty to disclose all information they are aware of which is believed relevant to the examination of the above-identified application, applicants believe that their invention is patentable.

We hereby certify that each of the references set forth on the attached form PTO-1449 was cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of this Information Disclosure Statement.

Please acknowledge receipt of this Information Disclosure Statement and kindly make the cited references of record in the above-identified application.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

Enclosures:
Postcard
Form PTO-1449
Cited References (6)

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

FORM PTO-1449 (REV.7-80) | U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE

ATTY. DOCKET NO.	APPLICATION NO.
661005.447	08/674,610
APPLICANTS	
George Heidorn and Karen Jensen	
FILING DATE	GROUP ART UNIT
June 28, 1996	~~2412~~ 2741

INFORMATION DISCLOSURE STATEMENT

(Use several sheets if necessary)

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
NH	AA	5,406,480	4/11/95	Kanno	~~364~~ 704	~~419.08~~ 10	
	AB						
	AC						
	AD						
	AE						
	AF						
	AG						
	AH						
	AI						

FOREIGN PATENT DOCUMENTS

		DOCUMENT NUMBER	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLATION YES	TRANSLATION NO
NH	AJ	0413132A2	7/11/90	Europe	~~G06F~~	~~15/38~~		
NH	AK	0413132A3	7/11/90	Europe	~~G06F~~	~~15/20~~		
	AL							
	AM							
	AN							

OTHER PRIOR ART *(Including Author, Title, Date, Pertinent Pages, Etc.)*

NH	AO	Geetha, T.V. and Subramanian, R.K., "Natural Language Representation - a Connectionist Approach", Computer and Communication, New Delhi, August 28, 1991, volume 3, pgs. 294-298.
NH	AP	Isahara, Hitoshi and Ishizaki, Shun, "Context Analysis System for Japanese Text", 11th International Conference on Computational Linguistics. Proceedings of Coling '86, Bonn, West Germany, August 25-29, 1986, pgs. 244-246.
NH	AQ	Winograd, Terry, "Computer Software for Working with Language", *Scientific American*, September 1984, New York, U.S.A., vol. 251, no. 3, pp. 90-101.

EXAMINER	DATE CONSIDERED
Kent [illegible] Smith	3/18/98

*EXAMINER: Initial if reference considered, whether or not criteria is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant(s).

Form P7-27/V1/1/94

International Conference on

EC³-ENERGY, COMPUTER, COMMUNICATION AND CONTROL SYSTEMS

August 28-30, 1991

PRE-CONFERENCE PROCEEDINGS - Volume 3

COMPUTER AND COMMUNICATION

ORGANISED BY IEEE DELHI SECTION AND INDIA COUNCIL

In Co-operation With :
IEEE Power Engineering Society
IEEE Computer Society
IEEE Communication Society
IEEE Control Society
IEEE Robotics and Automation Society
Computer Society of India

Venue
Hotel Taj Palace Inter. Continental, New Delhi

G06F15/20F4G

NATURAL LANGUAGE REPRESENTATION - A CONNECTIONIST APPROACH

T.V.Geetha

R.K.Subramanian

PUBLICATION DATE: 28. 08. 91
(further bibliographic data on next page).

School of Computer Science and Engineering
Anna University
Madras- 6000 25
India

p-294-298

Abstract: The work discusses a connectionist approach to the construction of syntactic and semantic representations. The need to handle symbol processing and heuristic search to process natural language in real time, has prompted the search for an alternative approach to tackle these issues. This has led to the cycle-intensive approach using a large number of very simple processors to get around the limitations of conventional symbolic processing. The system described in this work uses a sort and scan approach to assign lexical details to words. During the training phase of the syntactic module an incremental learning method to establish weak connections to represent the syntax of the natural language by exposing the system to example syntactic constituent patterns and sentence patterns. A bottom-up parsing strategy is used to progressively fire word nodes, syntactic constituent nodes, using a marker propagation and constraint decision technique to maintain the relative ordering. The approach creates a distributed representation of the parse tree and allows selective retrieval of parts of the structure. The semantic representation phase also works from the word nodes and utilizes projection rules to propagate markers and establish semantic connections after satisfying the semantic selectional restrictions imposed by the semantic descriptions. This marker propagation continues until a semantic structure has been associated with the sentence node.

Introduction

The application of a connectionist approach to higher-level cognitive tasks such as natural language processing has been one of the interesting problems that attracted the attention of both scientists working in the area of connectionist systems as well as computational linguists. Natural language processing involves the interpretation of natural language text bringing into play a large amount of linguistic and extra-linguistic pragmatic and world knowledge. Natural language understanding is considered a complex task because the understanding of language by human-beings is influenced not only by definable linguistic knowledge but by his environment, his social interaction, his intuitive knowledge and the conventions followed by discourse participants. The cognitive task of understanding can be explained as the application of an enormous amount of interdependent knowledge, acknowledgement of the multiple constraints imposed by linguistic theories and the construction of a mental model of the information conveyed by the text using symbol processing and heuristic search methods. Natural language understanding can be viewed as a mapping from the source representation (the natural language) to a target representation (the internal representation within the computer). The internal representation will be a parsed, interpreted, deep representation conveying the meaning intended by the producer of the text [1]. Thus in essence one of the major concerns of natural language understanding can be put in a nutshell as - the representation, manipulation and retrieval of a vast amount of input and processed knowledge. A connectionist approach is well suited to perform this cognitive task because of its ability to simultaneously satisfy multiple constraints [2] and the possibility of interpreting neural-network connection weights to model the criteria used to classify the input [3], in this case differentiate sentences from non-sentences. This work discusses a connectionist approach that uses a massively parallel architectural approach to deal with symbolic processing required for natural language process and representation. Such an approach allows the large number of simple virtual processors to perform the analysis simultaneously and also allows the selective retrieval from the comprehensive representation constructed as per the demands of the ensuing stages.

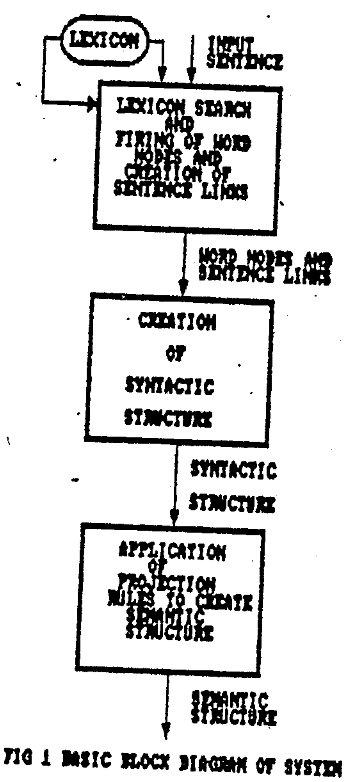


FIG 1 BASIC BLOCK DIAGRAM OF SYSTEM

The work described here outlines the application of a connectionist approach to parsing and the semantic interpretation phase required by most natural language systems. This work was attempted on the basis of successes achieved in using connectionist learning procedures to learn internal representations [4,5]. The major phases to be discussed are the lexicon organization and search mechanisms, the syntactic parser and the semantic interpreter all implemented using a connectionist approach where the application of a large number of simple processors to tackle the processing and the possibility of the system learning the regularities in the syntactic structure and the feature clustering of concepts for semantic interpretation. This system combines both the advantages of having independent systems to tackle syntactic analysis and semantic interpretation while allowing the connectionist approach to yank out the regularity present in language. The basic system block diagram is shown in Fig.1.

Connectionist Approach - An Introduction

Connectionist systems may be described as massively parallel architectures designed to tackle artificial intelligence problems. A connectionist system uses a large number of simple processing units. The units have limited storage capability. The long-term storage of knowledge is in the pattern of interconnections among the units and in the strengths of the connections between the processors. If the connections or weights between the units are allowed to learn, the network can build its own internal representations by forming an interleaved pattern of connections between multiple units. The network generally learns to use distributed representation in which each input vector is represented by activity in many different hidden units and each hidden unit may contribute in representing many different input vectors [6]. In this work we make use of the representation created as a result of the training procedure to represent the syntactic and semantic knowledge extracted from processing the text. In other words the processors modify their behavior in response to their environment that is shown a set of appropriate examples (perhaps with desired outputs) they self-adjust to produce consistent responses. These learning procedures allow the system to generalize (that is ignore slight variations) and to abstract (that is recognize some inputs it had never seen before). We have employed an incremental learning procedure where the system is fed first with rudimentary patterns and gradually the complexity of the patterns are increased as the training phase progresses. As patterns are fed to the system it is able to construct relationships between the elementary patterns and evolve a common structure. The structure is refined and made more accurate as more sentences are provided to the system.

The Lexicon Search Module

The lexicon is the core of any natural language processing system and is a vast storehouse of information. The lexical details are stored one root word per each virtual processor. The basic syntactic and semantic details are attached to each word. Each word of the text is also stored one per each virtual processor. Here we assume that the words have been morphologically pre-processed and the the root word and any information conveyed by the derivative is extracted before the commencement of the lexicon search procedure. The assigning of lexical words to all the words of the text has been performed using the scan and sort procedure based on associative or memory-based search [7]. In this method all the words of both the lexicon and the text are sorted together. Since the lexical items are subscripted with a zero and the text items with a one, the result of the sorting operation is that a word appears from the lexicon, followed by all its occurrences in the text, followed by the lexicographically next word and so on. Then the lexicon definitions associated with the word are spread to all the words in the text.

The virtual processor representing a word of the text called the word node in turn consists of four sub-nodes representing the syntactic, semantic, temporal and cohesive aspects of the word [Fig.2]. This independent association allows the extraction of any combination of details of a word by adjusting the connections between the word node and its sub-nodes. During this phase weak connections (with weights greater than zero but less than a full-fledged connection) are established [8]. Thus in the absence of any other processing, the structural links called the sentence links between the word nodes with associated l xical details are available.

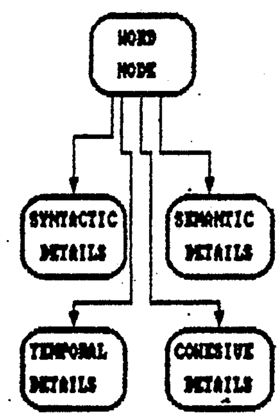


FIG 2 WORD NODE AND ASSOCIATED SUB-NODES

The Syntactic Parser

The grammar of any natural language can be expressed using a finite set of primitives which combine into patterns whose formation is constrained by a set of syntax rules. Since the number of syntactic categories and primitive syntactic constituents are limited, during the preprocessing stage a virtual processor is attached to each such syntactic category and constituent. By virtual processor here we mean a logical unit with capabilities for pattern matching and retrieval of the structure that it holds. Furthermore each such processor can operate asynchronously and in parallel with other similar processors. During the training phase the system is provided with simple examples of primitive syntactic patterns and then with examples of complete sentences representing complex syntactic structures. These examples are utilized by the system to learn the syntactic regularity presented by the sentences [Fig.3]. However the training phase is used only to establish weak links [Fig.4.].

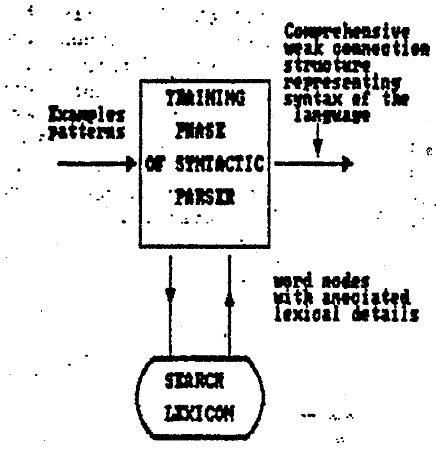


FIG 3 TRAINING PHASE OF THE PARSER MODULE

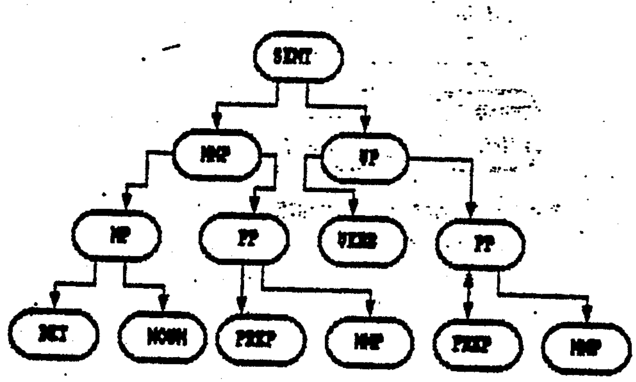


FIG 4 PART OF LEARNT SYNTACTIC STRUCTURE

Processor Allocation During Training:

When an example with the new syntactic phrase enters the system, the syntactic category associated with each word in the phrase is identified. The syntactic category of the first word is then broadcast to each processor which attempts to match it with the associated structure. A successful match would result in that particular processor being marked. Similarly the syntactic category of the next word is transmitted to the processors and the pattern matching process is repeated, and the syntactic category processor marked. This process is continued until the input phrase is exhausted. A failure to find a pattern indicates that this pattern has not been previously encountered by the system and a new pattern is allocated to this structure.

The Parsing Procedure:

After the completion of the lexicon search phase, physical connections called category links exist between the word nodes and the syntactic category nodes. During the processing, after the words have been attached with the lexical details, the word nodes will fire the particular syntactic category node to which it belongs, by strengthening the corresponding connection. It is possible that two category nodes are fired by the same word since a particular form of the word can belong to more than one syntactic category. Once the syntactic category nodes have been fired the corresponding syntactic constituent node. The system thus essentially employs a bottom-up parsing strategy where the word nodes are initially fired, then the corresponding syntactic category nodes in combination fire the syntactic constituent nodes until eventually the sentence nodes are fired [Fig.5]. However in addition to the presence of the syntactic constituent nodes fired as a result of the activation of the

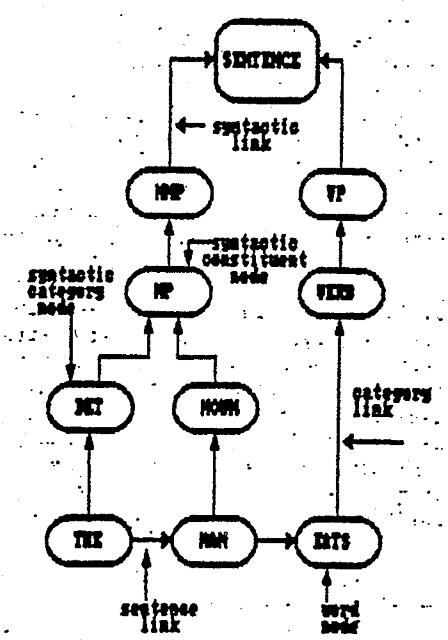


FIG 5 SYNTACTIC CONNECTIONS FOR AN EXAMPLE SENTENCE

syntactic category nodes, the relative order of the components is also of significance.

Marker propagation and the ordering constraint:

A marker propagation and constraint decision technique is used to deal with maintaining the relative order among the constituents. When a sentence link exists from node A to node B a marker is passed from syntactic category node of A to the syntactic category node of B. The presence of the marker enhances the connection weight of corresponding syntactic links. Although the weights of both the syntactic links are enhanced it is so designed that the syntactic link of A is enhanced less than the syntactic link of B. A syntactic constituent node is fired only if all its syntactic category nodes are fired and the decision constraint stating that the weights W1 < W2 < .... < Wn is satisfied where W1, W2, ..., Wn are weights of nodes N1, N2, ..., Nn thus dictating the order in which the syntactic category or the syntactic constituent nodes should occur.

Syntactic ambiguity:

The tackling of syntactic ambiguity has been performed by conventional parsers in many different ways, however most of the parsing methods required a backing mechanism and an additional amount of time for processing. In the parser used in this system ambiguous structures are constructed simultaneously during the first course of parsing. Two types of syntactic ambiguity are handled by the syntactic parser, one the local word syntactic category ambiguity where a word may belong to more than one syntactic category. In this case more than one syntactic category node will be fired by a single word node but the syntactic constituent formation constraint dictated by the grammar rules allows only one of the fixed category nodes to influence the firing of a syntactic constituent node.

The second type of ambiguity is more global in nature and arises due to the fact that some sentence may be assigned more than one syntactic structure, since the grammar rules do not specify an unique structure for the particular sentence. In this system this leads to more than the firing of more than one sentence node that is more than one structure is associated with the sentence. Eventually the constraints imposed by the semantic processing can help to disambiguate syntactic ambiguities [9].

Selective retrieval of syntactic structures:

The aim of the syntactic analyzer phase is not only to recognize the syntactic legality of the sentence but also to form a detailed syntactic list structure representing the parse tree of the sentence [10]. Selective retrieval of the surface parse tree of the sentence is used by the ensuing semantic processing stage where semantic representation is syntactically controlled and the representation may itself require semantic projection of particular constituents. Distributive representations enables the large number of primitive elements to selectively form constituent structures [11]. In our system the virtual processors that are fired form a complex constituent structure that represent all syntactic inter-dependencies between the words of the sentence. Since the structure is formed in a layered fashion it is possible to selectively retrieve any portion of the parse tree, by choosing the appropriate parent node.

Semantic Representation

The next level of processing constructs a comprehensive semantic representation using appropriate patterns from the syntactic structure constructed to select the appropriate semantic information necessary to completely represent the meaning of the sentence by making suitable connections. In essence it applies syntactic information to assign thematic roles to the sentence, where the verb is considered as the central concept, with the other syntactic constituents acting to enhance the meaning of the verb concept. Semantic details of words are represented by semantic markers which indicate the primitive semantic concepts and the categorized variable which are syntactic place markers [12]. These indicate where in the syntactic structure to look for constituents to fill the role and is expressed by specifying the path of the parse tree. Semantic details also associative selectional restrictions on the words. These are restrictions which decide the essential semantic concepts that have to be possessed by the other words in order to combine with the word. By suitable marker propagation dictated by the nature of the categorized variable (example [sent,nnp,np]) it is possible to extract the semantic details and check whether the selectional restrictions are satisfied. The semantic details of the role is then associated with the semantic details of the word by having a semantic link connection between them.

Appropriate projection rules are applied simultaneously to all the word nodes to fire the appropriate parent node. Projection rules can be applied only if the child nodes (or sub-constituents) already have semantic details associated with them that is they are fired. The application of appropriate projection rules continues up the tree till the sentence has been fired and the semantic representation has been obtained for the complete sentence.

Here the syntactic structure of the sentence, marker propagation initiated by the projection rules and the selective removal of syntactic constituents specified by categorized variables are used to build the semantic structure. The connectionist approach only allows the simultaneous initiation of projection rules whenever their condition of applications are satisfied to enable the system to efficiently build its semantic representation.

Conclusion

The system uses the syntactic information to guide the construction of the semantic representation and thus avoids the disadvantage of relying heavily an expectation raised by verbs about the realization of their arguments as in CD [13] and in the preference semantics systems [14]. Here the syntactic structure formed is used to control the semantic representation phase since separate projection rules are associated with each syntactic constituent. The connectionist approach allows the details available at the word nodes to pattern into distinct syntactic and semantic representations after linguistic constraints have been satisfied.

The system handles the syntactic analysis rather well since there is a specialized phase that handles parsing rather than use explicit cues to the syntax in the form of surface location of the constituents in the input as in the Sentence Gestalt Model [2]. The marker propagation allows a syntactically controlled semantic structure to be produced which can be independently stored and used for inferencing purposes.

The system could be extended to take into account the semantic constraints conveyed by contextual information by exposing the system to a large number of sentences defining similar circumstances. It could also incorporate world knowledge by mapping part - whole hierarchies into the connectionist system [15] and settle on a plausible model to deal with everyday common - sense reasoning [16]. Thus a full - fledged NLP system could be modeled with connectionist system possessing to produce distributed internal representations depicting syntactic, semantic contextual common - sense and world knowledge.

References

[1] M. W. Firebaugh, Artificial Intelligence = A Knowledge Based Approach. Burton: PWS Kent Publishing Company, 1989.

[2] M.F. St.John and J. L. McClelland, "Learning and applying contextual constraints in sentence comprehension," Artificial Intelligence, Vol. 46, No. 1-2, pp. 217-257, November 1990.

[3] G. D. Garson, "Interpreting neural-network connection weights," AI Expert, pp. 47-51, April 1991.

[4] S. E. Fahlman and G. E. Hinton, "Connectionist architectures for artificial intelligence," IEEE Computer, pp. 100-109, January 1987.

[5] L. G. Valiant, "A theory of the learnable," Comm. of the ACM 27, pp. 1134- 1142, 1984.

[6] G. E. Hinton, "Preface to special issue on connectionist symbol processing," Artificial Intelligence, Vol. 46, No. 1-2, pp. 217-257, November 1990.

[7] D. L. Waltz, "Applications of the connection machine," IEEE Computer, pp. 100-109, January 1987.

[8] I. Aleksander and H. Morton, An Introduction to Neural Computing, Chapman and Hall, 1990.

[9] T. V. Geetha and R. K. Subramanian, "Semantic predictions and disambiguation in natural language undestanding," Journal of Computer Science and Informatics, Vol. 17, No. 2, pp.5-13, 1988.

[10] T. V. Geetha and R. K. Subramanian, "Repesenting natural language with Prolog," IEEE Software, pp.85-91, March 1990.

[11] J. B. Pollack, "Recursive distributed representation," pp. 77-105, Artificial Intelligence, Vol. 46, No. 1-2, pp.77-105, November 1990.

[12] T. V. Geetha and R. K. Subramanian, "Natural language understanding - an effort to interpret pragmatics," IETE Journal, Vol.34, No.3, 1988.

[13] R. C. Schank, "Language and memory," Cognitive Science 2(3), 1980.

[14] Y. Wilks, "An intelligent analyzer and understander of English", Comm. of the ACM 18(5), 1975.

[15] G. E. Hinton, "Mapping part-whole hierarchies into connectionist networks," Artificial Intelligence, Vol. 46, No. 1-2, pp. 47-75, November 1990.

[16] M. Derthick, "Mundane reasoning by settling on a plausible model," Artificial Intelligence, Vol. 46, No. 1-2, pp. 107-157, November 1990.

****

XP 20460
11th Int. Conf. Computational linguistics (Bonn, 25-29 Aug. 1986)
EXPK
"(COLING '86)"
pp 244-246
G06F 17/276
G06F 15/38

Context Analysis System for Japanese Text

Hitoshi Isahara and Shun Ishizaki

Electrotechnical Laboratory
1-1-4, Umezono, Sakura-mura, Niihari-gun,
Ibaraki, Japan 305

ABSTRACT

A natural language understanding system is described which extracts contextual information from Japanese texts. It integrates syntactic, semantic and contextual processing serially. The syntactic analyzer obtains rough syntactic structures from the text. The semantic analyzer treats modifying relations inside noun phrases and case relations among verbs and noun phrases. Then, the contextual analyzer obtains contextual information from the semantic structure extracted by the semantic analyzer. Our system understands the context using precoded contextual knowledge on terrorism and plugs the event information in input sentences into the contextual structure.

1: Introduction

Despite the advanced state of syntactic analysis research for natural language processing and the many useful results it has produced, there have been few studies involving contextual information, and many problems remain unsolved.

The natural language understanding system described here employs a syntactic analyzer, a semantic analyzer treating modifying relations inside noun phrases and the relations among verbs and phrases, that is, word-level semantics, and a contextual analyzer (Fig. 1). These analyzers operate in a serially integrated fashion. Though humans seem to understand natural language texts using these three analyzers simultaneously, we have made their methodology essentially different from their human counterparts for more efficient computing. Our system uses a context-free grammar parser named Extended-Lingol as a syntactic analyzer to analyze the Japanese sentences and produce parsing trees. From an analysis of these, in turn, it obtains word-level semantic structures expressed in frame-like representations. Finally, it extracts contextual information, using our representation from the semantic structures. We remain far from certain at this stage whether this system represents the best realization of an engineering-based natural language understanding system. Future plans include combining these three processes into one process and bringing the system closer to the human process.

Because our system uses bottom-up analysis first (including syntactic analysis and word-level semantic analysis), it can obtain not only the outline of the input sentences but also their details, as necessary. This method is the best one in situations where the detailed information of texts are quite important, such as Machine-Translation systems and precise question-answering systems. Of course, in this way, we must build up a sizable dictionary of precise word definitions.

In our system, predictive-style processing is not used in syntactic analysis and word-level semantic analysis. But, in the contextual analysis part, predictions from the tree structure of the contextual information are used for instantiation of the contextual structure.

We are now developing a system which can understand newspaper articles through contextual structure (see Fig. 2a). After applying the procedures outlined above, the system obtains

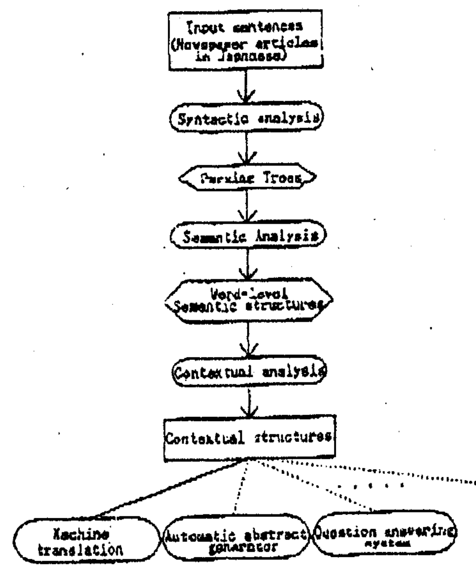
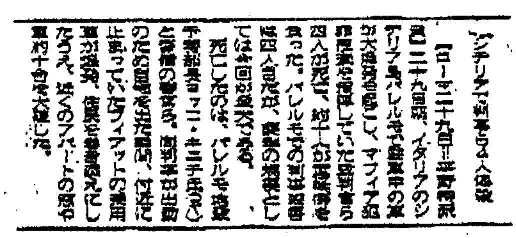


Fig.1 System flow chart of this paper and its applications.



シチリア 判事ら四人爆殺

【ローマ二十九日＝平野特派員】二十九日朝、イタリアのシシリア島パレルモで駐車中の車が大爆発を起こし、マフィア犯罪捜査を指揮していた判事ら四人が死亡、約十人が重軽傷を負った。パレルモでの判事殺害は四人目だが、規模の規模としては今回が最大である。

死亡したのは、パレルモ地裁予審部長ロッコ・キンニチ氏(五八)と護衛の警察官ら。同判事が出勤のため自宅を出た瞬間、付近に止まっていたフィアットの乗用車が爆発、護衛を巻き添えにしたうえ、近くのアパートの窓や車約十台を大破した。

a: Original input (Morning edition of the Asahi Shimbun--July 30, 1983).

THE BOMB KILLS FOUR PEOPLE INCLUDING A JUDGE.
[Rome 29th = correspondent Hirano]

In the morning of the 29th, at Palermo, Sicily in Italy, a parked car exploded, which killed 4 people including a judge who had directed an investigation into Mafia crimes, and injured about 10 people seriously or slightly. This is the fourth murder case on judges at Palermo and is of the largest scale.

Judge Rocco Chinnici, 58, the director of the Palermo preliminary court, police bodyguards and others were murdered. At the moment when the judge left home, the bomb exploded which had been set in the car of Fiatt parked near there. The explosion involved the residents, windows of the apartment and about 10 cars near there.

b: The translation of the example article (a) from Japanese into English.

Fig. 2. An example of newspaper articles

contextual representations expressed as shown in Fig. 3. Some details of the input text are abbreviated in the figure.

2: Syntactic and semantic analysis[2]

Let us proceed to an explanation of the methodologies adopted by our system, using the newspaper article in Fig. 2a as an example. First, the system analyzed each sentence syntactically, obtaining parsing trees. Next, the system constructs a semantic structure for each phrase. Word meanings in our word dictionary are described in SRL (Semantic Representation Language) which uses frame-like expression as shown in Fig. 4. Each word meaning shares a suitable position in the hierarchy of concepts. SRL enables deep semantic analysis in a flexible way. The formal definition of its syntax and semantics is not stated here. In our system, a word meaning written in the lexical entry using SRL plays an important role in semantic analysis. The interaction between the word meanings is the central issue of the semantic analysis. The modifying relations inside noun phrases and the case relations among verbs and noun phrases are determined in the word-level semantic structure. In Fig. 4, three scenes (explosion, death and injury) are obtained by analyzing the first sentence of the article in Fig. 2a. "Human" is a dummy node that means human beings. Here, the people who died include a judge and some policemen.

There are several types of ambiguity in input text. In syntactic analysis, ambiguity means the existance of several parsing trees. Word-level semantics often specify which should be selected. Here, we should use a kind of prediction. For example, people who are in authority could be a target of terrorism (See Fig. 2a). These constraints are very helpful in eliminating ambiguity, as well as surface syntactic information. Some of this processing is done in an interactive way in our system. Our system asks the user how to specify the relations between events in some decision points. Even after the elimination of ambiguity by the word semantics, there may be unsolved ambiguities. These will be eliminated by contextual analysis with the contextual structure.

3: Features of contextual representation

Our contextual structure fits into a tree structure with one root node and a number of leaf nodes. Relations between events in a story are defined in the structure as "scenes", and the relations among our structure are defined by a tree structure. Our structure can share scenes with others.

Leaf nodes with a shared root node have either an "and" or an "or" relationship with each other. The hierarchy shown in Fig. 5 is an example. The node "terrorism involving bomb" has, as in Fig. 5, three leaf nodes (scenes) - "explosion," "damage" and "rescue". Since these seem to occur serially, the relationship among them is an "and" relationship. On the other hand, the root node "terrorist action" in Fig. 5 has several leaf nodes - "terrorism involving bomb", "shooting" and so on. As only one of these usually corresponds to the main topic in newspaper stories, they share an "or" relationship with each other.

Input events are matched not only directly with scenes in the structure, but also with higher concepts in accordance with a predefined tree structure of a concept hierarchy like that in Fig. 6. In other words, the system has a concept thesaurus. So, matching between the scene of the structure and the input events becomes flexible.

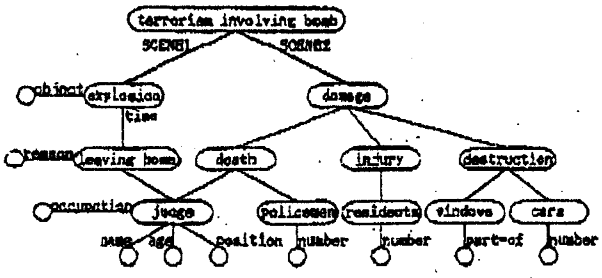


Fig. 3. An example of the contextual structure.

```
爆発する(explode)
    ACTOR  自動車(car)
               STATE  駐車中(parked)
                          LOCATION  パレルモ(Palermo)
                                     LOCATION  シシリー(Sicily)
                                                LOCATION  イタリア(Italy)
    TIME  20日朝(on the morning of the 20th)
    SCALE  大きい(large)
死亡する(die)
    ACTOR  人間(human)
               NUMBER  4
               INCLUSION  判事(judge)
                           JOB  指揮する(direct)
                                  OBJECT  捜査(inspection)
                                           OBJECT  犯罪(crime)
                                                    ACTOR  マフィア(Mafia)
重軽傷を負う(injured seriously or slightly)
    ACTOR  人間(human)
               NUMBER  約10人(above 10)
```

Fig. 4. The word-level semantic structure extracted from the first sentence in Fig. 2a.

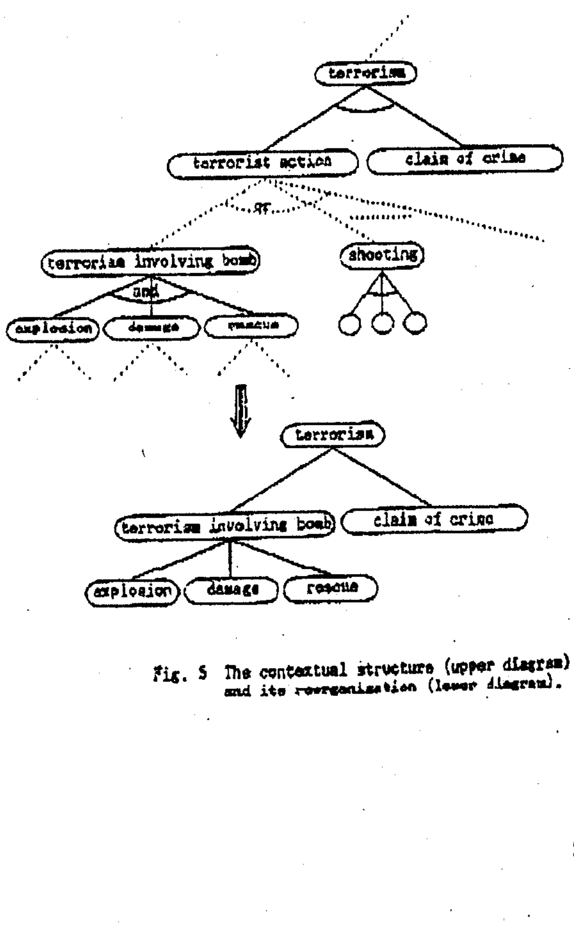


Fig. 5 The contextual structure (upper diagram) and its reorganisation (lower diagram).

XP.002040077

189C ientific American
251(1984) Sept., No. 3, New York, USA

G06F 17/276

E

Computer Software for Working with Language

p. 91 – 101

Programs can manipulate linguistic symbols with great facility, as in word-processing software, but attempts to have computers deal with meaning are vexed by ambiguity in human languages

by Terry Winograd

In the popular mythology the computer is a mathematics machine: it is designed to do numerical calculations. Yet it is really a language machine: its fundamental power lies in its ability to manipulate linguistic tokens—symbols to which meaning has been assigned. Indeed, "natural language" (the language people speak and write, as distinguished from the "artificial" languages in which computer programs are written) is central to computer science. Much of the earliest work in the field was aimed at breaking military codes, and in the 1950's efforts to have computers translate text from one natural language into another led to crucial advances, even though the goal itself was not achieved. Work continues on the still more ambitious project of making natural language a medium in which to communicate with computers.

Today investigators are developing unified theories of computation that embrace both natural and artificial languages. Here I shall concentrate on the former, that is, on the language of everyday human communication. Within that realm there is a vast range of software to be considered. Some of it is mundane and successful. A multitude of microcomputers have invaded homes, offices and schools, and most of them are used at least in part for "word processing." Other applications are speculative and far from realization. Science fiction is populated by robots that converse as if they were human, with barely a mechanical tinge to their voice. Real attempts to get computers to converse have run up against great difficulties, and the best of the laboratory prototypes are still a pale reflection of the linguistic competence of the average child.

The range of computer software for processing language precludes a comprehensive survey; instead I shall look at four types of program. The programs deal with machine translation, with word processing, with question answering and with the adjuncts to electronic mail known as coordination systems. In each case the key to what is possible lies in analyzing the nature of linguistic competence and how that competence is related to the formal rule structures that are the theoretical basis of all computer software.

The prospect that text might be translated by a computer arose well before commercial computers were first manufactured. In 1949, when the few working computers were all in military laboratories, the mathematician Warren Weaver, one of the pioneers of communication theory, pointed out that the techniques developed for code breaking might be applicable to machine translation.

At first the task appears to be straightforward. Given a sentence in a source language, two basic operations yield the corresponding sentence in a target language. First the individual words are replaced by their translations; then the translated words are reordered and adjusted in detail. Take the translation of "Did you see a white cow?" into the Spanish "*¿Viste una vaca blanca?*" First one needs to know the word correspondences: "*vaca*" for "cow" and so on. Then one needs to know the structural details of Spanish. The words "did" and "you" are not translated directly but are expressed through the form of the verb "*viste*." The adjective "*blanca*" follows the noun instead of preceding it as it does in English. Finally, "*una*" and "*blanca*" are in the feminine form corresponding to "*vaca*." Much of the early study of machine translation dwelt on the technical problem of putting a large dictionary into computer storage and empowering the computer to search efficiently in it. Meanwhile the software for dealing with grammar was based on the then current theories of the structure of language, augmented by rough-and-ready rules.

The programs yielded translations so bad that they were incomprehensible. The problem is that natural language does not embody meaning in the same way that a cryptographic code embodies a message. The meaning of a sentence in a natural language is dependent not only on the form of the sentence but also on the context. One can see this most clearly through examples of ambiguity.

In the simplest form of ambiguity, known as lexical ambiguity, a single word has more than one possible meaning. Thus "Stay away from the bank" might be advice to an investor or to a child too close to a river. In translating it into Spanish one would need to choose between "*orilla*" and "*banco*," and nothing in the sentence itself reveals which is intended. Attempts to deal with lexical ambiguity in translation software have included the insertion of all the possibilities into the translated text and the statistical analysis of the source text in an effort to decide which translation is appropriate. For example, "*orilla*" is likely to be the correct choice if words related to rivers and water are nearby in the source text. The first strategy leads to complex, unreadable text; the second yields the correct choice in many cases but the wrong one in many others.

In structural ambiguity the problem goes beyond a single word. Consider the sentence "He saw that gasoline can explode." It has two interpretations based on quite different uses of "that" and "can." Hence the sentence has two possible grammatical structures, and the translator must choose between them [*see bottom illustration on page 93*].

An ambiguity of "deep structure" is subtler still: two readings of a sentence can have the same apparent grammatical structure but nonetheless differ in meaning. "The chickens are ready to eat" implies that something is about to eat something, but which are the chickens? One of the advances in linguistic

F4G.

189C Scientific American
251(1984) Sep ,No.3,New York,USA

theory since the 1950's has been the development of a formalism in which the deep structure of language can be represented, but the formalism is of little help in deducing the intended deep structure of a particular sentence.

A fourth kind of ambiguity—semantic ambiguity—results when a phrase can play different roles in the overall meaning of a sentence. The sentence "David wants to marry a Norwegian" is an example. In one meaning of the sentence the phrase "a Norwegian" is referential. David intends to marry a particular person, and the speaker of the sentence has chosen an attribute of the person—her being from Norway—in order to describe her. In another meaning of the sentence the phrase is attributive. Neither David nor the speaker has a particular person in mind; the sentence simply means that David hopes to marry someone of Norwegian nationality.

A fifth kind of ambiguity might be called pragmatic ambiguity. It arises from the use of pronouns and special nouns such as "one" and "another." Take the sentence "When a bright moon ends a dark day, a brighter one will follow." A brighter day or a brighter moon? At times it is possible for translation software to simply translate the ambiguous pronoun or noun, thereby preserving the ambiguity in the translation. In many cases, however, this strategy is not available. In a Spanish translation of "She dropped the plate on the table and broke it," one must choose either the masculine "*lo*" or the feminine "*la*" to render "it." The choice forces the translator to decide whether the masculine "*plato*" (plate) or the feminine "*mesa*" (table) was broken.

In many ambiguous sentences the meaning is obvious to a human reader, but only because the reader brings to the task an understanding of context. Thus "The porridge is ready to eat" is unambiguous because one knows porridge is inanimate. "There's a man in the room with a green hat on" is unambiguous because one knows rooms do not wear hats. Without such knowledge virtually any sentence is ambiguous.

Although fully automatic, high-quality machine translation is not feasible, software is available to facilitate translation. One example is the computerization of translation aids such as dictionaries and phrase books. These vary from elaborate systems meant for technical translators, in which the function of "looking a word up" is made a part of a multilingual word-processing program, to hand-held computerized libraries of phrases for use by tourists. Another strategy is to process text by hand to make it suitable for machine translation. A person working as a "pre-editor" takes a text in the source language and creates a second text, still in the source language, that is simplified in ways facilitating machine translation. Words with multiple meanings can be eliminated, along with grammatical constructions that complicate syntactic analysis. Conjunctions that cause ambiguity can be suppressed, or the ambiguity can be resolved by inserting special punctuation, as in "the [old men] and [women]." After the machine translation a "post-editor" can check for blunders and smooth the translated text.

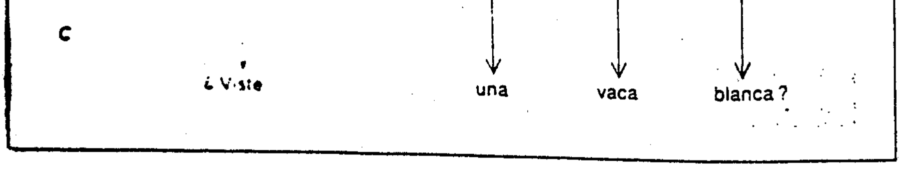
The effort is sometimes cost-effective. In the first place, the pre-editor and post-editor need not be bilingual, as a translator would have to be. Then too, if a single text (say an instruction manual) is to be translated into several languages, a large investment in pre-editing may be justified because it will serve for all the translations. If the author of the text can be taught the less ambiguous form of the source language, no pre-editor is needed. Finally, software can help in checking the pre-edited text to make certain it meets the specifications for input to the translation system (although this is no guarantee that the translation will be acceptable).

A machine-translation system employing pre- and post-editing has been in use since 1980 at the Pan-American Health Organization, where it has translated more than a million words of text from Spanish into English. A new system is being developed for the European Economic Community, with the goal of translating documents among the official languages of the community: Danish, Dutch, English, French, German, Greek and Italian. Meanwhile the theoretical work on syntax and meaning has continued, but there have been no breakthroughs in machine translation. The ambiguity pervading natural language continues to limit the possibilities, for reasons I shall examine more fully below.

I turn next to word processing, that is, to software that aids in the preparation, formatting and printing of text. Word processors deal only with the manipulation and display of strings of characters and hence only with superficial aspects of the structure of language. Even so, they pose technical problems quite central to the design of computer software. In some cases the end product of a word-processing program is no more than a sequence of lines of text. In others it is a complex layout of typographic elements, sometimes with drawings intercalated. In still others it is a structured document, with chapter headings, section numbers and so on, and with a table of contents and an index compiled by the program.

The key problems in designing word-processing software center on issues of representation and interaction. Representation is the task of devising data structures that can be manipulated conveniently by the software but still make provision for the things that concern the user of the system, say the layout of the printed page. Interaction takes up the issue of how the user expresses instructions and how the system responds.

Consider the fundamental problem of employing the data-storage devices of a computer to hold an encoded sequence of natural-language characters. The first devices that encoded text were card-punch and teletype machines, and so the earliest text-encoding schemes were tailored to those devices. The teletype machine is essentially a typewriter that converts key presses into numerical codes that can be transmitted electronically;



MACHINE TRANSLATION of text from one language into another was thought to be quite feasible in the 1950's, when the effort was undertaken. In the first step of the process (*a*) the computer would search a bilingual dictionary to find translations of the individual words in a source sentence (in this case Spanish equivalents of the words in the sentence "Did you see a white cow?"). Next the translated words would be rearranged according to the grammar of the target language (*b*). The changes at this stage could include excision or addition of words. Finally, the morphology of the translation (for example the endings of words) would be adjusted (*c*).

189C .ientific American
251(1984) Sept.,No.3,New York,USA

thus there are teletype codes for most of the keys on a typewriter. The codes include the alphabetic characters *A* through *Z*, the digits 0 through 9 and common punctuation marks such as the period and the comma. Standards are harder to establish, however, for symbols such as ÷, @, ¢ and }. And what about keys that print nothing, such as the tab key, the carriage-return key and the backspace key?

The difficulties that arise in choosing standards can be illustrated by one peculiarity of text encoding. The teletype code distinguishes between a carriage return (which returns the type carriage to the beginning of the line without advancing the paper) and a line feed (which advances the paper without repositioning the carriage). Hence the end of a line is marked by a sequence of two characters: a carriage return and a line feed. One code would suffice, and so some programs eliminate either the carriage return or the line feed, or they replace both characters with another code entirely. The problem is that various programs employ different conventions, so that lines encoded by one program may not be readable by another.

The problems become worse when a full range of characters—punctuation marks, mathematical symbols, diacritical marks such as the umlaut—is considered. Moreover, word processing is now being extended to languages such as Chinese and Japanese, which require thousands of ideographic characters, and to languages such as Arabic and Hebrew, which are written from right to left. Coding schemes adequate for English are useless for alphabets with thousands of characters. It should be said that the schemes continue to vary because political and economic forces play a role in the design of computer systems. A given manufacturer wants to promulgate a standard that suits its own equipment; thus some present-day standards exist because they were offered by a vendor that dominates a market. On the other hand, technical matters such as the efficiency of certain software running on certain hardware perpetuate differences in detail. It will be quite a while before universal standards emerge and users gain the ability to transport text from one word-processing system to any other.

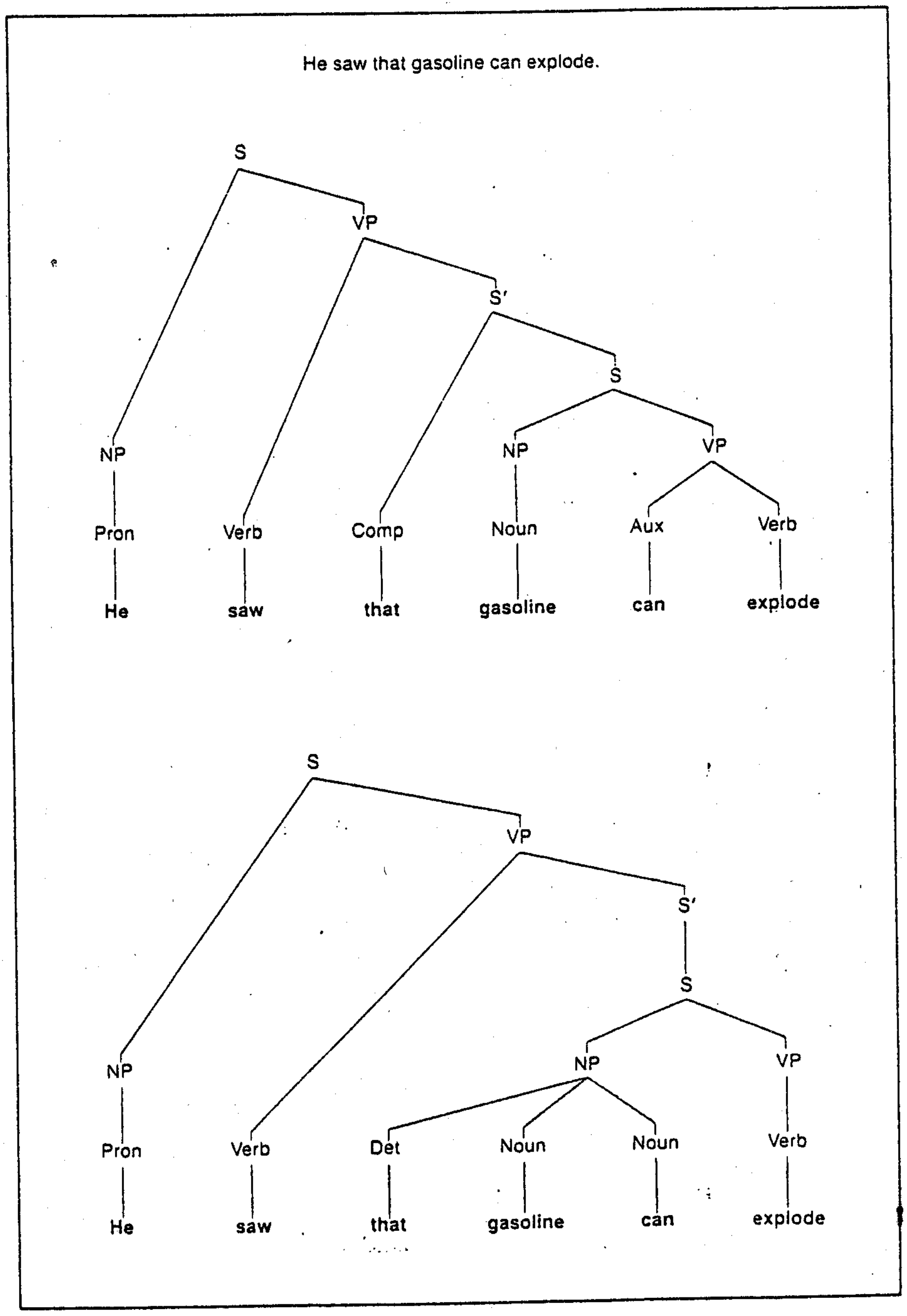
Encoding schemes aside, there is the form of the letters themselves. On a typewriter keyboard an A is simply an A. Typographically, however, an A is an A or an *A* or an **A**. In the new field of digital typography the computer is a tool for the design and presentation of forms of type. Some of the efforts in the field are applied to the forms themselves: in particular the representation of characters as composites of dots and spaces. Additional efforts go into the devising of code for the computer stor-

Stay away from the bank.

bank *n* 1. the rising terrain that borders a river or lake.

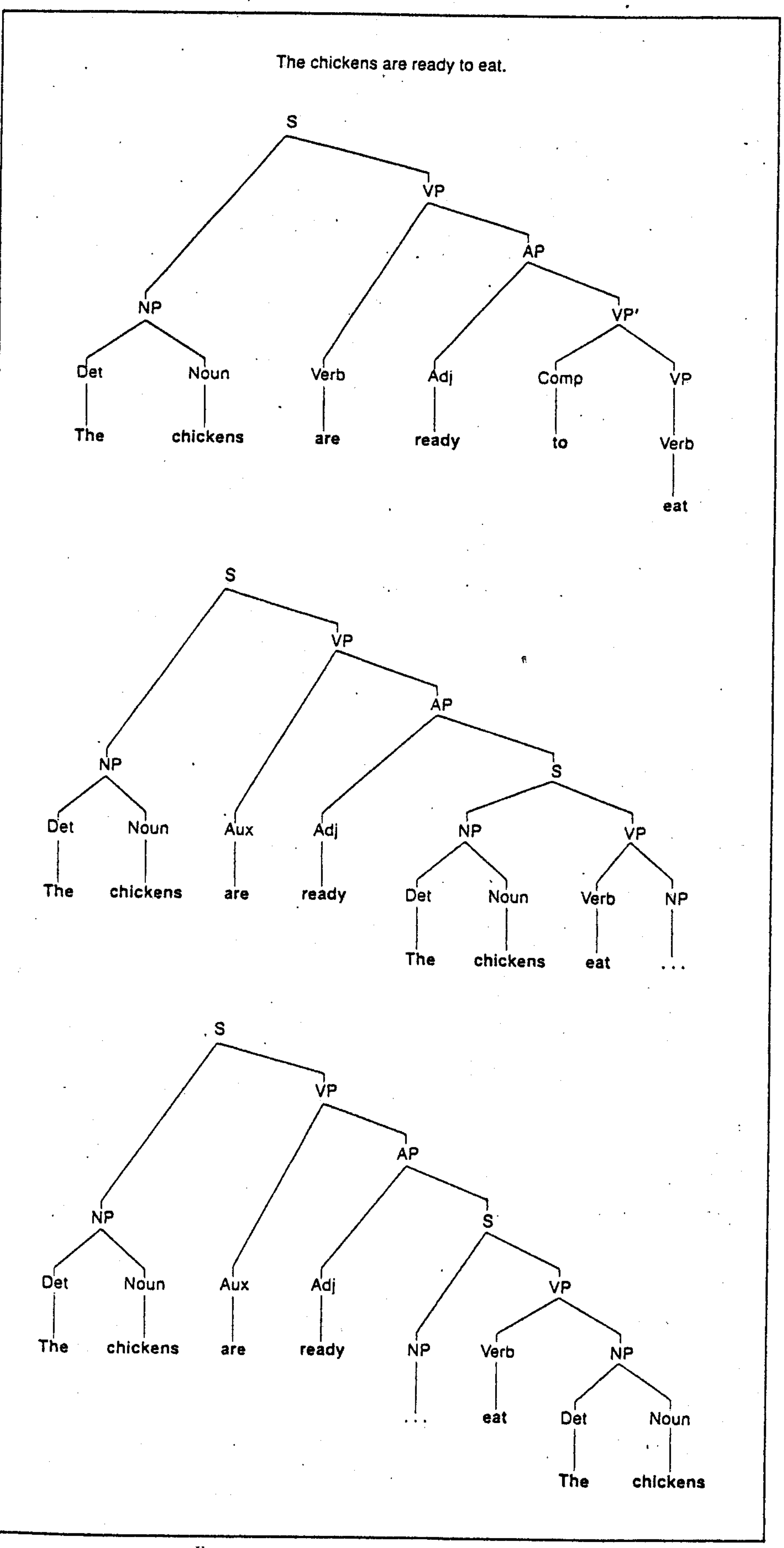
bank *n* 2. an establishment for the deposit, loan, issuance and transmission of money.

AMBIGUOUS MEANINGS permeate natural languages (that is, languages that people speak and write) and thus subvert the attempt to have computers translate text from one language into another. Here lexical ambiguity, the simplest type of ambiguity, is diagrammed. In lexical ambiguity a word in a sentence has more than one possible meaning. In this case the word is "bank" (*color*), which might equally well refer to either a river or a financial institution. A translator must choose. The following four illustrations show more complex types of ambiguity.



STRUCTURAL AMBIGUITY arises when a sentence can be described by more than one grammatical structure. Here the conflicting possibilities for the sentence "He saw that gasoline can explode" are displayed in the form of grammatical "trees." In one of the trees the sentence has a subordinate clause whose subject is "gasoline" (*color*); the sentence refers to the recognition of a property of that substance. In the other tree "gasoline can" is part of a noun phrase (*NP*) meaning a container of gasoline; the sentence refers to the sight of a specific explosio

189C Scienti : American
251(1984)Sep..,No.3,New York,USA



DEEP-STRUCTURAL AMBIGUITY arises when a sentence has a single apparent structure but nonetheless has more than one possible meaning. In this example the sentence is "The chickens are ready to eat." Its grammatical structure (*top*) leaves the role of the chickens ambiguous: in one interpretation they will eat; in the other they will be eaten. Deep-structure trees make the chickens' role explicit: they are the subject of the sentence (*middle*), in which case their food is undetermined, or they are the object (*bottom*), and their eaters are undetermined.

age of text that combines different fonts (such as Times Roman and Helvetica) and different faces (such as *italic* and **boldface**).

So far I have dealt only with stored sequences of characters. Yet one of the major tasks of a word-processing program is to deal with margins and spacing—with the "geography" of the printed page. In the typesetting language called TEX commands that specify nonstandard characters, change the style of type, set the margins and so on are embedded in the text [*see top illustration on page 96*]. A command to TEX is distinguished from ordinary text by the backslash character (\). The stored text is "compiled" by the TEX program, which interprets the embedded commands in order to create a printed document in the specified format.

The compiling is quite complex, and a good deal of computation is often needed to get from code created by means of a word-processing program to code that readily drives a printer or a typesetting machine. An algorithm that justifies text (fills the full width of each line of type) must determine how many words will fit in a line, how much space should be inserted between the words and whether a line would be improved by dividing and hyphenating a word. The algorithm may also take actions to avoid visual defects such as a line with wide interword spacing followed by a line that is very compact. Positioning each line on the page is further complicated by the placement of headings, footnotes, illustrations, tables and so on. Mathematical formulas have their own typographic rules.

TEX and similar programs are primitive with respect to another aspect of word processing: the user interface. The high-resolution display screens becoming available are now making it possible for the computer to display to the user a fair approximation of the pages it will print, including the placement of each item and the typeface to be employed. This suggests that the user should not have to type special command sequences but might instead manipulate page geography directly on the screen by means of the computer keyboard and a pointing device such as a "mouse." The resulting interface between the computer and the user would then fall into the class of interfaces known as WYSIWYG, which stands for "What you see is what you get."

It is worth noting that programs for manipulating text are called different things by different professions. Programmers call them text editors, but in business and publishing they are referred to as word processors; in the latter fields an editor is a person who works to improve the quality of text. Computer software is emerging to aid in this

189C . .ientific American
251(1984) Sept., No.3, New York, USA

more substantive aspect of editing. It deals with neither the visual format of language nor the conceptual content but with spelling, grammar and style. It includes two kinds of programs: mechanized reference works and mechanized correctness aids.

An example of a mechanized reference work is a thesaurus program designed so that when the writer designates a word, a list of synonyms appears on the display screen. In advanced systems the thesaurus is fully integrated into the word-processing program. The writer positions a marker to indicate the word to be replaced. The thesaurus is then invoked; it displays the alternatives in a "window" on the screen. The writer positions the marker on one of the alternatives, which automatically replaces the rejected word.

The design of such a program involves both linguistic and computational issues. A linguistic issue is that the mechanism for looking up a word should be flexible enough to accept variant forms. For example, the store of information pertaining to "endow" should be accessible to queries about "endowed," "endowing," "endows" and even "unendowed" or "endowment." Recognizing the common root in such words calls for a morphological analysis, which can be done by techniques developed in the course of work on machine translation. Computational issues include devising methods for storing and searching through a thesaurus or a dictionary, which must be fairly large to be useful.

A correctness aid deals with spelling, grammar and even elements of style. The simplest such programs attempt to match each word in a text with an entry in a stored dictionary. Words that have no match are flagged as possible misspellings. Other programs look for common grammatical errors or stylistic infelicities. For example, the Writer's Workbench software developed at AT&T Bell Laboratories includes programs that search for repeated words, such as "the the" (a common typing mistake), for incorrect punctuation such as "?." and for wordy phrases such as "at this point in time." A different correctness aid calls attention to "pompous phrases" such as "exhibit a tendency" and "arrive at a decision" and suggests simpler replacements such as "tend" and "decide." Still another correctness aid searches for gender-specific terms such as "mailman" and "chairman" and suggests replacements such as "mail carrier" and "chairperson."

In addition to searching a text for particular strings of characters, some correctness-aid programs do statistical analyses. By calculating the average length of sentences, the length of words and similar quantities, they compute a "readability index." Passages that score poorly can be brought to the writer's attention. No program is yet able to make a comprehensive grammatical analysis of a text, but an experimental system called Epistle, developed at the International Business Machines Corporation, makes some grammatical judgments. It employs a grammar of 400 rules and a dictionary of 130,000 words. As with all software that tries to parse text without dealing with what the text means, there are many sentences that cannot be analyzed correctly.

David wants to marry a Norwegian.

$\exists x$ Norwegian(x) $\wedge$ Want(David,[Marry(David,x)])

Want(David,[$\exists x$ Norwegian(x) $\wedge$ Marry(David,x)])

SEMANTIC AMBIGUITY arises when a phrase can play different roles in the meaning of a sentence. Here the roles of the phrase "a Norwegian" become explicit when the sentence "David wants to marry a Norwegian" is "translated" into a logical form based on the notation called predicate calculus. According to one interpretation, the speaker of the sentence has a particular person in mind and has chosen nationality as a way to specify who. Hence the sentence means: There exists ($\exists$) an x such that x is Norwegian and ($\wedge$) x is the person David wants to marry. According to another interpretation, neither David nor the speaker has any particular person in mind. David might be going to Norway hoping to meet someone marriageable.

She dropped the plate on the table and broke it.

She dropped the plate on the table and broke [the plate].

She dropped the plate on the table and broke [the table].

PRAGMATIC AMBIGUITY arises when a sentence is given more than one possible meaning by a word such as the pronoun "it." Suppose a computer is given the sentence shown in the illustration. If the computer has access to stored knowledge of the grammar of English sentences but lacks access to commonsense knowledge of the properties of tables and plates, the computer could infer with equal validity that the table was broken or that the plate was broken.

Is there software that really deals with meaning—software that exhibits the kind of reasoning a person would use in carrying out linguistic tasks such as translating, summarizing or answering a question? Such software has been the goal of research projects in artificial intelligence since the mid-1960's, when the necessary computer hardware and programming techniques began to appear even as the impracticability of machine translation was becoming apparent. There are many applications in which the software would be useful. They include programs that accept natural-language commands, programs for information retrieval, programs that summarize text and programs that acquire language-based knowledge for expert systems.

No existing software deals with meaning over a significant subset of English; each experimental program is based on finding a simplified version of language and meaning and testing what can be done within its confines. Some investigators see no fundamental barrier to writing programs with a full understanding of natural language. Others argue that computerized understanding of language is impossible. In order to follow the arguments it is important to examine the basics of how a language-understanding program has to work.

A language-understanding program needs several components, corresponding to the various levels at which language is analyzed [*see illustrations on pages 96–100*]. Most programs deal with written language; hence the analysis of sound waves is bypassed and the first level of analysis is morphological. The program applies rules that decompose a word into its root, or basic form, and inflections such as the endings *-s* and *-ing*. The rules correspond in large part to the spelling rules children are taught in elementary school. Children learn, for example, that the root of "baking" is "bake," whereas the root of "barking" is "bark." An exception list handles words to which the rules do not apply, such as forms of the verb "be." Other rules associate inflections with "features" of words. For example, "am going" is a progressive verb: it signals an act in progress.

For each root that emerges from the morphological analysis a dictionary yields the set of lexical categories to which the root belongs. This is the second level of analysis carried out by the computer. Some roots (such as "the") have only one lexical category; others have several. "Dark" can be a noun or *a*

\inset
This is a sample of a {\italic justified} piece of text, which contains {\eightpoint small letters {\bold and }} {bigFont big ones}.
It includes foreign words such as \lquote pe\~na\rquote—which is Spanish—and foreign letters like \alpha\ and \aleph,
which can be baffling, and includes one \hskip 1.3in wide space.

b . . .

01110100	01100101	01110010	01110011	00000000	00100111	00101101	11010011	00001000	01100001	01101110	01100100	00000000
t	e	r	s	NEW ENTITY	FONT CODE	X-POSITION	Y-POSITION	X-INCREMENT	a	n	d	NEW ENTITY

00110100	00110001	10110110	00101101	01100010	01101001	01100111	00100000	01101111	01101110	01100101	01101011	00101110
FONT CODE	X-POSITION	Y-POSITION	X-INCREMENT	b	i	g	SPACE	o	n	e	s	•

00000000	00000001	10101111	10110110	00101100	01001001	01110100	00100000	01101001	. . .
NEW ENTITY	FONT CODE	X-POSITION	Y-POSITION	X-INCREMENT	I	t	SPACE	i	

c

This is a sample of a *justified* piece of text. which contains small letters **and**
big ones. It includes foreign words such as "peña"—which is Spanish—and foreign letters like α and $\aleph$. which can be baffling. and includes one wide space.

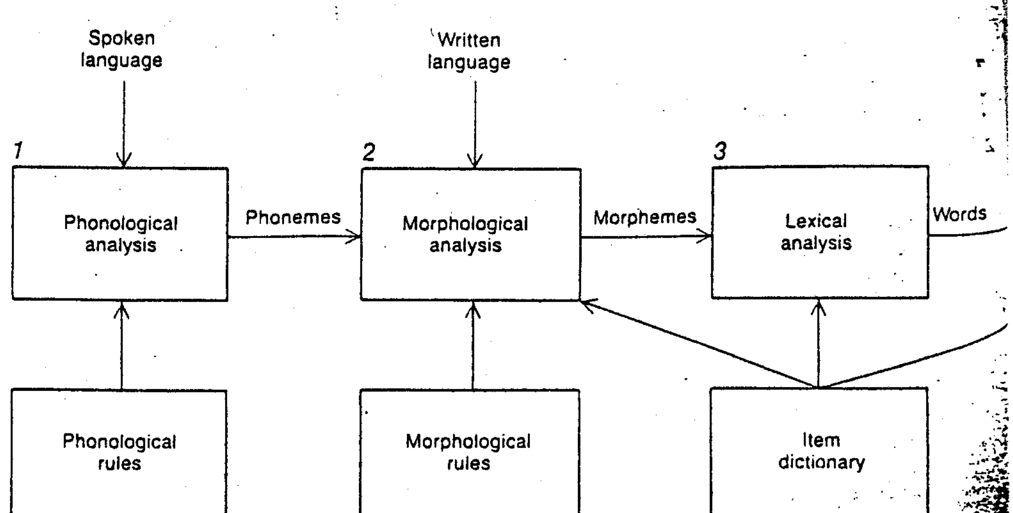
WORD PROCESSING, that is, the computer-aided preparation and editing of text, requires several representations of the text, because the format best for interactions between the software and its user is not efficient for sending instructions to a printing machine, nor can it efficiently give a preview of the result of the printing. In the typesetting language TEX the user's typed input (*a*) includes commands that specify nonstandard characters, change the style of type, set margins and so on. Such commands are distinguished by a backslash (\). The TEX software "compiles" the input, producing computer code that will drive a printing machine (*b*). To that end the code is divided into "entities," each of which specifies the typeface and the starting position for a sequence of words. Coded "*X* increments" space out the words to fill the distance between margins on the printed page; thus they "justify" lines of type. The printed page (*c*) shows the result.

an adjective; "bloom" can be a noun or a verb. In some instances the morphological analysis limits the possibilities. (In its common usages "bloom" can be a noun or a verb, but "blooming" is only a verb.) The output of the morphological and lexical analysis is thus a sequence of the words in a sentence, with each word carrying a quantity of dictionary and feature information. This output serves in turn as the input to the third component of the program, the parser, or syntactic-analysis component, which applies rules of grammar to determine the structure of the sentence.

Two distinct problems arise in designing an adequate parser. The first problem is the specification of a precise set of rules—a grammar—that determines the set of possible sentence structures in a language. Over the past 30 years much work in theoretical linguistics has been directed toward devising formal linguistic systems: constructions in which the syntactic rules of a language are stated so precisely that a computer could employ them to analyze the language. The generative transformational grammars invented by Noam Chomsky of the Massachusetts Institute of Technology were the first comprehensive attempt; they specify the syntax of a language by means of a set of rules whose mechanical application generates all allowable structures.

The second problem is that of the parsing itself. It is not always possible to tell, when a part of a sentence is encountered, just what role it plays in the sentence or whether the words in it go together. Take the sentence "Roses will be blooming in the dark gardens we abandoned long ago." The words "in the dark" might be interpreted as a complete phrase; after all, they are grammatically well formed and they make sense. But the phrase cannot form a coherent unit in a complete analysis of the sentence because it forces "Roses will be blooming in the dark" to be interpreted as a sentence and therefore leaves "gardens we abandoned long ago" without a role to play.

Parsers adopt various strategies for exploring the multiple ways phrases can be put together. Some work from the top down, trying from the outset to find possible sentences; others work from the bottom up, trying local word combinations. Some backtrack to explore alternatives in depth if a given possibility fails; others use parallel processing



COMPUTERIZED UNDERSTANDING OF LANGUAGE requires the computer to draw on several types of stored data (*white boxes*) and perform several levels of analysis (*colored boxes*). If the language is spoken, the first analysis is phonological (*1*): the computer analyzes sound waves. If the language is written, the first analysis is morphological (*2*): the computer decomposes each word into its root, or basic form, and inflections (for example *-ing*). Next is lexi-

189C :entific American
251(1984) Sept.,No.3,New York,USA

to keep track of a number of alternatives simultaneously. Some make use of formalisms (such as transformational grammar) that were developed by linguists. Others make use of newer formalisms designed with computers in mind. The latter formalisms are better suited to the implementation of parsing procedures. For example, "augmented-transition networks" express the structure of sentences and phrases as an explicit sequence of "transitions" to be followed by a machine. "Lexical-function grammars" create a "functional structure" in which grammatical functions such as head, subject and object are explicitly tied to the words and phrases that serve those functions.

Although no formal grammar successfully deals with all the grammatical problems of any natural language, existing grammars and parsers can handle well over 90 percent of all sentences. This is not entirely to the good. A given sentence may have hundreds or even thousands of possible syntactic analyses. Most of them have no plausible meaning. People are not aware of considering and rejecting such possibilities, but parsing programs are swamped by meaningless alternatives.

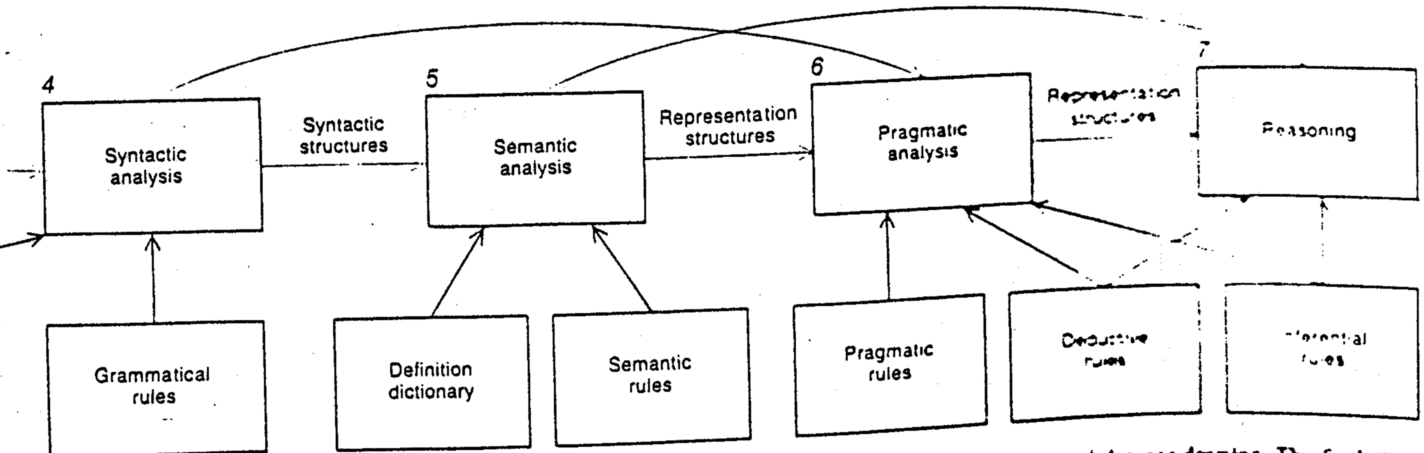
The output of a parsing program becomes the input to the fourth component of a language-understanding program: a semantic analyzer, which translates the syntactic form of a sentence into a "logical" form. The point is to put the linguistic expressions into a form that makes it possible for the computer to apply reasoning procedures and draw inferences. Here again there are competing theories about what representation is most appropriate. As with parsing, the key issues are effectiveness and efficiency.

Effectiveness depends on finding the appropriate formal structures to encode the meaning of linguistic expressions. One possibility is predicate calculus, which employs the quantifiers $\forall$ to mean "all" and $\exists$ to mean "there exists." In predicate calculus "Roses will be blooming..." is equivalent to the assertion "There exists something that is a rose and that is blooming...." This entails a difficulty. Is one rose adequate to represent the meaning of "roses will be blooming," or would it be better to specify two or more? How can the computer decide? The dilemma is worsened if a sentence includes a mass noun such as "water" in "Water will be flowing...." One cannot itemize water at all. In designing a formal structure for the meaning of linguistic expressions many similar problems arise from the inherent vagueness of language.

Efficiency must also be considered, because the computer will employ the logical form of a sentence to draw inferences that in turn serve both the analysis of the meaning of the sentence and the formulation of a response to it. Some formalisms, such as predicate calculus, are not directly amenable to efficient computation, but other, more "procedural" representations have also been devised. Consider the effort to answer the question "Are there flowers in the gardens we abandoned long ago?" The computer needs to know that roses are flowers. This knowledge could be represented by a formula in predicate calculus amounting to the assertion "Everything that is a rose is a flower." The computer could then apply techniques developed for mechanical theorem-proving to make the needed deduction. A different approach would be to give certain inferences a privileged computational status. For example, basic classificational deductions could be represented directly in data structures [*see bottom illustration on page 100*]. Such deductions are required constantly for reasoning about the ordinary properties of objects. Other types of fact (for example that flowers need water in order to grow) could then be represented in a form closer to predicate calculus. The computer could draw on both to make inferences (for example that if roses do not get water, they will not grow).

A good deal of research has gone into the design of "representation languages" that provide for the effective and efficient encoding of meaning. The greatest difficulty lies in the nature of human commonsense reasoning. Most of what a person knows cannot be formulated in all-or-nothing logical rules; it lies instead in "normal expectations." If one asks, "Is there dirt in the garden?" the answer is almost certainly yes. The yes, however, cannot be a logical inference; some gardens are hydroponic, and the plants there grow in water. A person tends to rely on normal expectations without thinking of exceptions unless they are relevant. But little progress has been made toward formalizing the concept of "relevance" and the way it shapes the background of expectations brought to bear in the understanding of linguistic expressions.

The final stage of analysis in a language-understanding program is pragmatic analysis: the analysis of context. Every sentence is embedded in a setting: it comes from a particular speaker at a particular time and it refers, at least implicitly, to a particular body of understanding. Some of the embedding is straightforward: the pronoun "I" refers to the speaker; the adverb "now" refers to the moment at which the sen-



cal analysis (3), in which the computer assigns words to their lexical category (noun, for instance) and identifies "features" such as plurals. Then comes syntactic analysis, or parsing (4): the application of rules of grammar to yield the structure of the sentence. After that comes semantic analysis (5). Here the sentence is converted into a form that makes it amenable to inference-drawing. The final stage is pragmatic (6): it makes explicit the context of the sentence, such as the relation between the time at which it is spoken and the time to which it refers. The computer is now in a position to draw inferences (7), perhaps in preparation for responding to the sentence.

189C S[illegible]entific American
251(1[illegible])Sept.,No.3,New York,USA

tence is uttered. Yet even these can be problematic: consider the use of "now" in a letter I write today expecting you to read it three or four days hence. Still, fairly uncomplicated programs can draw the right conclusion most of the time. Other embedding is more complex. The pronoun "we" is an example. "We" might refer to the speaker and the hearer or to the speaker and some third party. Which of these it is (and who the third party might be) is not explicit and in fact is a common source of misunderstanding when people converse.

Still other types of embedding are not signaled by a troublesome word such as "we." The sentence "Roses will be blooming..." presupposes the identification of some future moment when the roses will indeed be in bloom. Thus the sentence might have followed the sentence "What will it be like when we get home?" or "Summer is fast upon us." Similarly, the noun phrase "the dark gardens we abandoned long ago" has a context-dependent meaning. There may be only one instance of gardens in which we have been together; there may be more than one. The sentence presupposes a body of knowledge from which the gardens are identifiable. The point is that a phrase beginning with "the" rarely specifies fully the object to which it refers.

One appr[illegible]ch to such phrases has been to enc[illegible] knowledge of the world in a form the program can use to make inferences. For example, in the sentence "I went to a restaurant and the waiter was rude" one can infer that "the waiter" refers to the person who served the speaker's meal if one's knowledge includes a script, so to speak, of the typical events attending a meal in a restaurant. (A particular waiter or waitress serves any given customer.) In more complex cases an analysis of the speaker's goals and strategies can help. If one hears "My math exam is tomorrow, where's the book?" one can assume that the speaker intends to study and that "the book" means the mathematics text employed in a course the speaker is taking. The approach is hampered by the same difficulty that besets the representation of meaning: the difficulty of formalizing the commonsense background that determines which scripts, goals and strategies are relevant and how they interact. The programs written so far work only in highly artificial and limited realms, and it is not clear how far such programs can be extended.

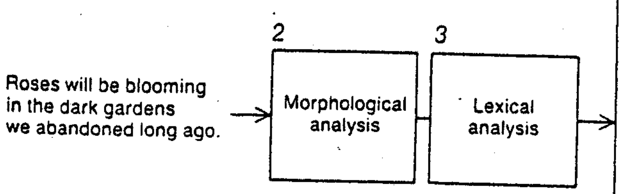
Even more problematic are the effects of context on the meaning of words. Suppose that in coming to grips with "the dark gardens we abandoned long ago" one tries to apply a particular meaning to "dark." Which should it be? The "dark" of "those dark days of tribulation" or that of "How dark it is with the lights off!" or that of "dark colors"? Although a kernel of similarity unites the uses of a word, its full meaning is determined by how it is used and by the prior understanding the speaker expects of the hearer. "The dark gardens" may have a quite specific meaning for the person addressed; for the rest of us it is slightly mysterious.

At first it might seem possible to distinguish "literal" uses of language from those that are more metaphorical or poetical. Computer programs faced with exclusively literal language could then be freed from contextual dilemmas. The problem is that metaphor and "poetic meaning" are not limited to the pages of literature. Everyday language is pervaded by unconscious metaphor, as when one says, "I lost two hours trying to get my idea across." Virtually every word has an open-ended field of meanings that shade gradually from those that seem utterly literal to those that are clearly metaphorical.

The limitations on the formalization of contextual meaning make it impossible at present—and conceivably forever—to design computer programs that come close to full mimicry of human language understanding. The only programs in practical use today that attempt even limited understanding are natural-language "front ends" that enable the user of a program to request information by asking questions in English. The program responds with English sentences or with a display of data.

A program called SHRDLU is an early example. Developed in the late 1960's, it enables a person to communicate with a computer in English about a simulated world of blocks on a tabletop. The program analyzes requests, commands and statements made by the user and responds with appropriate words or with actions performed in the simulated scene. SHRDLU succeeded in part because its world of conversation is limited to a simple and specialized domain: the blocks and a few actions that can be taken with them.

Some more recent front-end interfaces have been designed with practical applications in mind. A person wanting access to information stored in the computer types natural-language sentences



Word	Root	Lexical categories	Features
Roses	rose	Noun	[plural]
will		Verb (auxiliary)	[modal]
be		Verb (auxiliary)	[infinitive]
		Verb (copular)	[infinitive]
blooming	bloom	Verb (intransitive)	[progressive]
in		Preposition	
the		Determiner	[definite]
dark		Adjective	
		Noun	[mass]
gardens	garden	Noun	[plural]
		Verb	[third-person, singular, present]
we		Pronoun	[first-person, plural, nominative]
abandoned	abandon	Verb (transitive)	[past]
		Verb (transitive)	[participle]
long		Adjective	
ago		Adverb	

SUCCESSION OF ANALYSES done by a hypothetical computer program suggests how software that understands language works. In this illustration the program has been given the sentence "Roses will be blooming in the dark gardens we abandoned long ago." The first analyses (morphological and lexical) yield a list of the words in the sentence, with their roots, their lexical categories and their features. "Blooming," for instance, is a progressive verb: it signifies an act in progress. The data serve as input for the syntactic level of analysis: the parsing of the sentence. Here the surface, or grammatical, structure of "Roses will be blooming..." is put in the form of a tree. Pre-

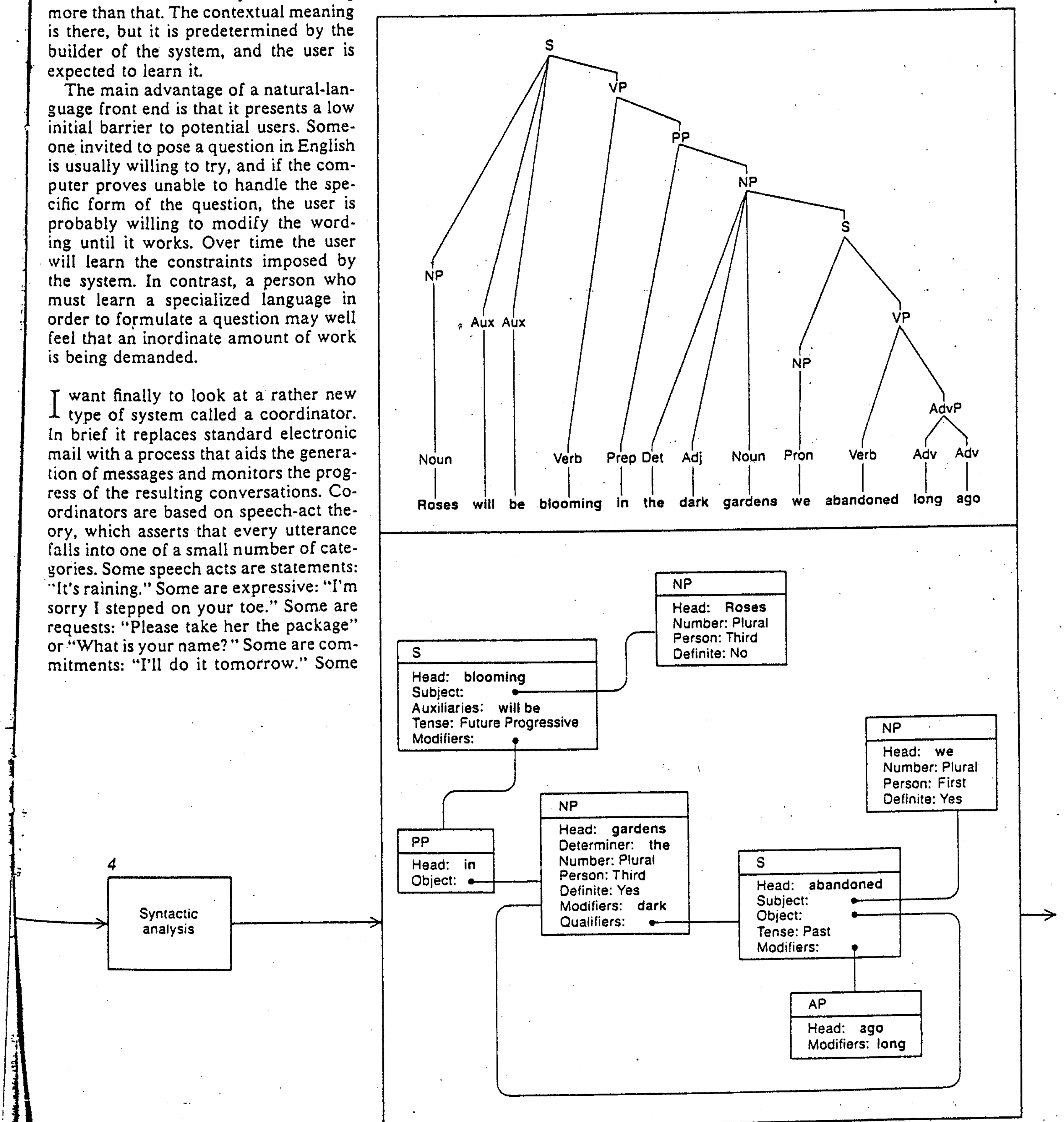
189C .entific American
251(1984) Sept.,No.3,New York,USA

that the computer interprets as queries. The range of the questioning is circumscribed by the range of the data from which answers are formulated; in this way words can be given precise meaning. In a data base on automobiles, for example, "dark" can be defined as the colors "black" and "navy" and nothing more than that. The contextual meaning is there, but it is predetermined by the builder of the system, and the user is expected to learn it.

The main advantage of a natural-language front end is that it presents a low initial barrier to potential users. Someone invited to pose a question in English is usually willing to try, and if the computer proves unable to handle the specific form of the question, the user is probably willing to modify the wording until it works. Over time the user will learn the constraints imposed by the system. In contrast, a person who must learn a specialized language in order to formulate a question may well feel that an inordinate amount of work is being demanded.

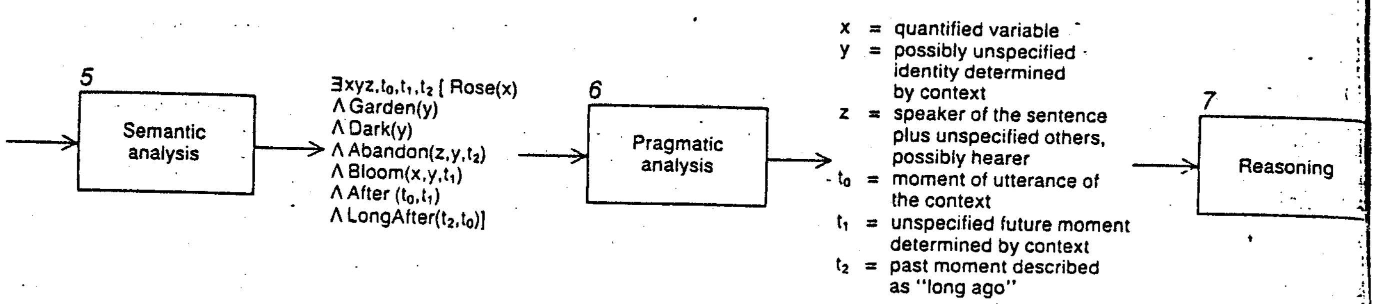
I want finally to look at a rather new type of system called a coordinator. In brief it replaces standard electronic mail with a process that aids the generation of messages and monitors the progress of the resulting conversations. Coordinators are based on speech-act theory, which asserts that every utterance falls into one of a small number of categories. Some speech acts are statements: "It's raining." Some are expressive: "I'm sorry I stepped on your toe." Some are requests: "Please take her the package" or "What is your name?" Some are commitments: "I'll do it tomorrow." Some are declarative: "You're fired." (Declaratives differ from statements in that they take effect by virtue of having been said.)

The classification of speech acts is useful because acts in the various categories do not occur at random. Each speech act has "felicity conditions" under which it is an appropriate thing to say and "conditions of satisfaction" under which it is fulfilled. For example, a request or a commitment carries with it, either implicitly or explicitly, a time by which it should be satisfied. Moreover,



ably the computer discards numerous incorrect trees. For example, it discards a tree in which "Roses will be blooming in the dark" is construed as a sentence. The deep structure of "Roses will be blooming..." is put in the form of a functional-structure diagram. There the relations between the parts of a sentence become explicit; they are shown by strings between boxes. Some relations were explicit in the surface structure (for example that "roses" is the subject of "blooming"). Others were not (for example that "gardens" is the object of "abandoned"). The syntactic analysis is supplied to the final stages of the program, which appear in the top illustration on the next page.

189C Scientific American
251(J) Sept.,No.3,New York,USA



ANALYSES CONCLUDE with the conversion of the syntactic structure of "Roses will be blooming..." into a form from which the computer can draw inferences. In this example the conversion is based on predicate calculus; thus the semantic-analysis module of the hypothetical software represents the logical content of "Roses will be blooming..." by symbols that can be translated as "x is a rose and y is a garden and y is dark...." Finally, the pragmatic-analysis module specifies what is known about the variables x, y, z, t_0, t_1 and t_2. The variable x, for example, is "quantified": it declares the existence of something instead of identifying a particular object. In other words, the computer takes "roses" as referring to roses in general, not to particular roses. Hence roses is not a "definite" noun. (That decision was made in the course of semantic analysis.) On the other hand, z remains ambiguous because it stands for the ambiguous pronoun "we."

each speech act is part of a conversation that follows a regular pattern. The regularity is crucial for successful communication.

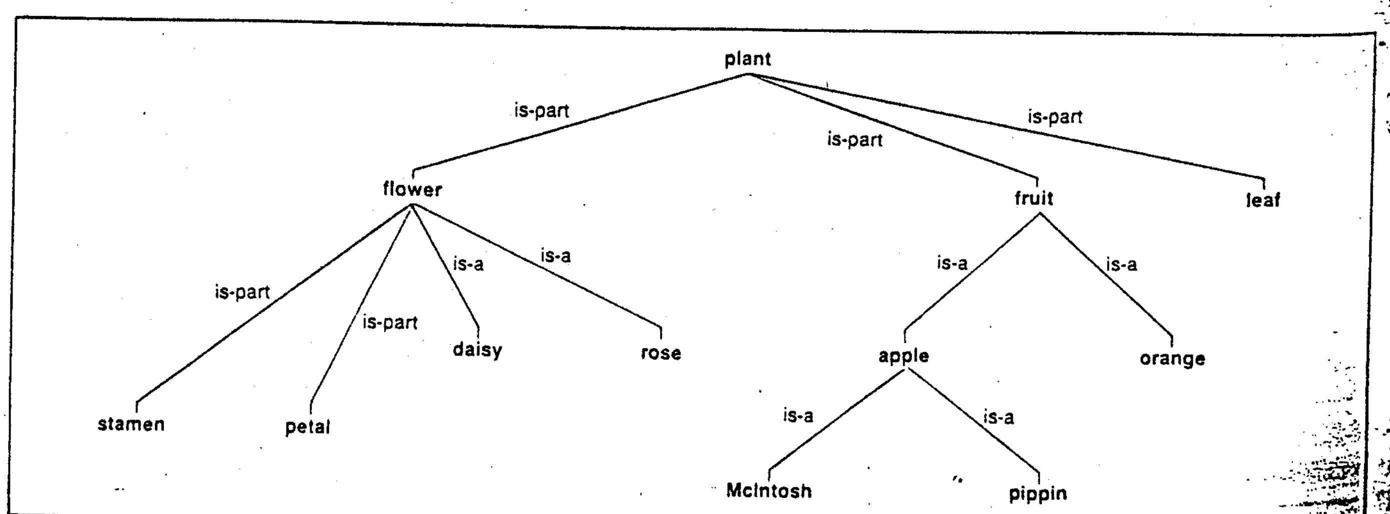
As with every aspect of language, the full understanding of any given speech act is always enmeshed in the unarticulated background expectations of the speaker and the hearer. The speech act "I'll be here tomorrow" might be a prediction or a promise, and "Do you play tennis?" might be a question or an invitation. In spoken conversation intonation and stress play a prominent part in establishing such meaning.

Coordinator systems deal with the speech acts embodied in messages by specifying what needs to be done and when. The system does not itself attempt to analyze the linguistic content of messages. Instead the word-processing software at the sender's end asks the sender to make explicit the speech-act content of each message. A person may write "I'll be happy to get you that report" in the message itself but must add (with a few special keystrokes) that the message is an ACCEPT of a particular REQUEST. The computer system can then keep track of messages and their interconnections. In particular the system can monitor the completion of conversations, calling the users' attention to cases in which something immediate is pending or in which an agreed-on time for satisfaction has not been met.

From a broad perspective, coordinators are just one member of a large family of software that gives users a structured medium in which language is augmented by explicit indications of how things fit together. Another type of software in this family provides tools for outlining and cross-indexing documents. Still another type is a computerized bulletin board that enables users to store and receive messages not addressed to a specific receiver. The messages are "posted" with additional structure that indicates their content and helps interested readers to find them.

The most obvious prediction about the future of computer software dealing with language is that the decreasing cost of hardware will make applications that are possible but impractical today available quite widely in the future. Yet software that mimics the full human understanding of language is simply not in prospect. Some specific trends can be noted.

The first is that spoken language will get more emphasis. To be sure, the computerized understanding of spoken language presents all the difficulties of written language and more. Merely separating an utterance into its component words can vex a computer; thus hopes for a "voice typewriter" that types text from dictation are just as dim as hopes for high-quality machine translation and language-understanding. On the other hand, many useful devices do not require the analysis of connected speech. Existing systems that can identify a spoken word or phrase from a fixed vocabulary of a few hundred items will improve the interface between users and machines; the recent emergence of inexpensive integrated-circuit chips that



SEMANTIC NETWORK is a specialized form of stored data that represents logical relations so that certain types of inference can be drawn efficiently by a computer. Here a simple tracing of links in the network (*color*) has yielded the inference that a pippin is a fruit and that a rose has petals. Facts not readily represented by a network can be represented in other ways, for example by predicate calculus.

189C Scientific American
251(1984) Sept.,No.3,New York,USA

process acoustic signals will facilitate the trend. Speech synthesizers that generate understandable utterances (although not in a natural-sounding voice) will also play an increasing role. Improved speech "compression" and encoding techniques will make acoustic messages and acoustic annotation of computer files commonplace.

A second trend in software dealing with language is that constraints on linguistic domain will be handled with increasing care and theoretical analysis. At several points in this article I have noted instances in which computers deal with meaning in an acceptable way because they operate in a limited domain of possible meanings. People using such software quickly recognize that the computer does not understand the full range of language, but the subset available is nonetheless a good basis for communication. Much of the commercial success of future software that deals with language will depend on the discovery of domains in which constraints on what sentences can mean still leave the user a broad range of language.

A third trend lies in the development of systems that combine the natural and the formal. Often it is taken for granted that natural language is the best way for people to communicate with computers. Plans for a "fifth generation" of intelligent computers are based on this proposition. It is not at all evident, however, that the proposition is valid. In some cases even the fullest understanding of natural language is not as expressive as a picture. And in many cases a partial understanding of natural language proves to be less usable than a well-designed formal interface. Consider the work with natural-language front ends. Here natural language promotes the initial acceptance of the system, but after that the users often move toward stylized forms of language they find they can employ with confidence, that is, without worrying about whether or not the machine will interpret their statements correctly.

The most successful current systems facilitate this transition. Some systems (including coordinators) mix the natural and the formal: the user is taught to recognize formal properties of utterances and include them explicitly in messages. Thus the computer handles formal structures, while people handle tasks in which context is important and precise rules cannot be applied. Other systems incorporate a highly structured query system, so that as the user gains experience the artificial forms are seen to save time and trouble. In each case the computer is not assigned the difficult and open-ended tasks of linguistic analysis; it serves instead as a structured linguistic medium. That is perhaps the most useful way the computer will deal with natural language.

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR		ATTORNEY DOCKET NO.
08/674,610	06/28/96	HEIDORN	G	661005.447

LM61/0327

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

EXAMINER	
ZINTEL,H	
ART UNIT	**PAPER NUMBER**
2741	7

DATE MAILED: 03/27/98

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev. 2/95)
*U.S. GPO: 1996-404-496/40510

1- File Copy

Office Action Summary	Application No. 08/674,610	Applicant(s) Heldorn
	Examiner Harold Zintel	Group Art Unit 2741

☒ Responsive to communication(s) filed on *Jun 28, 1996*

☐ This action is **FINAL.**

☐ Since this application is in condition for allowance except for formal matters, **prosecution as to the merits is closed** in accordance with the practice under *Ex parte Quayle* 1935 C.D. 11; 453 O.G. 213.

A shortened statutory period for response to this action is set to expire *3* month(s), or thirty days, whichever is longer, from the mailing date of this communication. Failure to respond within the period for response will cause the application to become abandoned. (35 U.S.C. § 133). Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

Disposition of Claim

☒ Claim(s) *1-36* is/are pending in the applicat

Of the above, claim(s) ______ is/are withdrawn from consideration

☐ Claim(s) ______ is/are allowed.

☒ Claim(s) *1-36* is/are rejected.

☐ Claim(s) ______ is/are objected to.

☐ Claims ______ are subject to restriction or election requirement.

Application Papers

☒ See the attached Notice of Draftsperson's Patent Drawing Review, PTO-948.

☒ The drawing(s) filed on *Jun 28, 1996* is/are objected to by the Examiner.

☐ The proposed drawing correction, filed on ______ is ☐ approved ☐ disapproved.

☐ The specification is objected to by the Examiner.

☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

☐ Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).

☐ All ☐ Some* ☒ None of the CERTIFIED copies of the priority documents have been

☐ received.

☐ received in Application No. (Series Code/Serial Number) ______.

☐ received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

*Certified copies not received: ______

☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

☐ Notice of References Cited, PTO-892

☒ Information Disclosure Statement(s), PTO-1449, Paper No(s). *6*

☐ Interview Summary, PTO-413

☒ Notice of Draftsperson's Patent Drawing Review, PTO-948

☐ Notice of Informal Patent Application, PTO-152

--- SEE OFFICE ACTION ON THE FOLLOWING PAGES ---

DETAILED ACTION

Drawings

1. Figure 1-23 should be designated by a legend such as --Prior Art-- because only that which is old is illustrated. See MPEP § 608.02(g).

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claim 12 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. The phrase "with and infinitive clause" is not clear. It is assumed that this should read "with an infinitive clause"

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless --
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

5. Claims 1-36 are rejected under 35 U.S.C. 102(b) as being disclosed by Jensen et al. "Natural Language Processing: The PLNLP Approach" included in the information disclosure statement. Jensen teaches the following features:

a. As per claims 1, 21, 22, 24, 25, 27:

i. adding syntactic rules and adjusting the syntax parse tree, on page 3 in the *Syntax* and *Corrected Syntax* component sections, which anticipates "including implied" and "added syntactic roles" to create a complete syntactic analysis implied by the phrase "the broad-coverage English sentence analysis grammar"

ii. generating a separate skeletal logical form, on page 4 in *Derivation of logical form* section

iii. adding and adjusting logical form graph, on page 4 in *Sense disambiguation* section, also as shown in figure 5 on page 209

b. As per claims 2 and 3: adding syntactic constructs for omitted verb after a predefined word, figure 6 on page 209 shows syntactic construct related to the predefined word "to"

c. As per claim 5: a syntactic construct for a pronoun, figure 4 page 209.

d. As per claims 7 and 8: syntactic construct for coordinate structures, in figure 5 on page 209, in this case "and"

e. As per claim 10: resolving long-distance attachment phenomena, on page 207 the paragraph above figure 3.

f. As per claim 13: logical form graph based on syntax parse tree, page 204 first paragraph

g. As per claims 14-20: deep parts of speech, on pages 205 and 206

h. As per claim 23 and 26: semantic labels, in figure 2 on page 205

i. As per claim 28, 29 and 31: syntax parse tree and logical form graph, in figures 1 and 2 on page 205, both of which a speaker of the natural language can understand

j. As per claim 30: a first and second sub-components, on page 4 in *Derivation of logical form* section, taking step "a" as the first sub-component and "b" and "c" together as the second sub-component

k. As per claim 32 and 33: add syntactic roles, on page 204 second paragraph

l. As per claim 34: a skeletal logical form, on page 4, as a graph which is the basis for further semantic processing

m. As per claim 35: add semantic labels to skeletal graph, in figure 2 on page 205

n. As per claim 36: constructs a complete logical form graph, on page 204 in section 16.1

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 4 rejected under 35 U.S.C. 103(a) as being unpatentable over Jensen as applied to claim 1.

a. As pre claim 4, Jensen teaches the step of adding a syntactic construct for a omitted verb after the word "to", as discussed in claim 3 but not after the word "not". It would have been obvious at the time the invention was made to a person having ordinarily skill in the art to use the same type of construct for a omitted verb after the word "not" since both are defined as "omitted" and are thus needed to form the complete logical graph of the sentence.

b. As per claim 5, Jensen teaches the step of adding syntactic constructs for the word "and" as discussed in claim 8, but not the word "or". It would have been obvious at the time the invention was made to a person having ordinarily skill in the art to use the same type of construct for the word "or" since both are coordinate structures and are needed to form the complete logical graph of the.

c. As per claim 11, Jensen teaches transforming a syntax parse tree into a logical graph but, does not teach transforming verbal phrases into verbs with prepositional phrase objects. It is well known in the art these form equivalent parse trees. Therefore, it would have been obvious at the time the invention was made to a person having ordinarily skill in the art to first do this transformation in order to form a logical graph.

d. As per claim 12, Jensen teaches transforming a syntax parse tree into a logical graph but does not teach replacing "it" with an infinitive clause. It is well known in the art these form equivalent parse trees. Therefore, it would have been obvious at the time the invention was made to a person having ordinarily skill in the art to first do this transformation in order to form a logical graph.

8. *Conclusion*

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 308-9051, (for formal communications intended for entry)

Or:

(703) 305-9508 (for informal or draft communications, please label PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA., Sixth Floor (Receptionist).

Any inquiry concerning this communication from the examiner should be directed to Harold A. Zintel whose telephone number is (703) 305-2381. The examiner can normally be reached on Monday-Friday from 8:30 a.m.-5:00 p.m.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, David Hudspeth, can be reached at (703) 305-4825.

The facsimile phone number for the Art Unit is (703) 305-9508. Alternately, facsimile messages may be sent directly to (703) 305-9644 where they will be stored in the examiner's voice mailbox (telling the examiner that a fax was received) and be automatically printed (i.e. - no delay by examiner).

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-3900.

Harold A. Zintel

Assistant Patent Examiner

March 26, 1998

DAVID R. HUDSPETH
SUPERVISORY PATENT EXAMINER
GROUP 2700

Form PTO 948 (Rev. 10-94) U.S. DEPARTMENT OF COMMERCE - Patent and Trademark Office Application No. 674 610

NOTICE OF DRAFTSPERSON'S PATENT DRAWING REVIEW

PTO Draftpersons review all originally filed drawings regardless of whether they are designated as formal or informal. Additionally, patent Examiners will review the drawings for compliance with the regulations. Direct telephone inquiries concerning this review to the Drawing Review Branch, 703-305-8404.

The drawings filed (insert date) 6/28/96, are
A.___ not objected to by the Draftsperson under 37 CFR 1.84 or 1.152.
B. X objected to by the Draftsperson under 37 CFR 1.84 or 1.152 as indicated below. The Examiner will require submission of new, corrected drawings when necessary. Corrected drawings must be submitted according to the instructions on the back of this Notice.

1. DRAWINGS. 37 CFR 1.84(a): Acceptable categories of drawings: Black ink. Color.
   - ___ Not black solid lines. Fig(s)___
   - ___ Color drawings are not acceptable until petition is granted. Fig(s) ___
2. PHOTOGRAPHS. 37 CFR 1.84(b)
   - ___ Photographs are not acceptable until petition is granted. Fig(s) ___
   - ___ Photographs not properly mounted (must use brystol board or photographic double-weight paper). Fig(s)___
   - ___ Poor quality (half-tone). Fig(s)___
3. GRAPHIC FORMS. 37 CFR 1.84 (d)
   - ___ Chemical or mathematical formula not labeled as separate figure. Fig(s)___
   - ___ Group of waveforms not presented as a single figure, using common vertical axis with time extending along horizontal axis. Fig(s)___
   - ___ Individuals waveform not identified with a separate letter designation adjacent to the vertical axis. Fig(s)___
4. TYPE OF PAPER. 37 CFR 1.84(c)
   - ___ Paper not flexible, strong, white, smooth, nonshiny, and durable. Sheet(s)___
   - ___ Erasures, alterations, overwritings, interlineations, cracks, creases, and folds copy machine marks not accepted. Fig(s) ___
   - ___ Mylar, velum paper is not acceptable (too thin). Fig(s) ___
5. SIZE OF PAPER. 37 CFR 1.84(f): Acceptable sizes:
   21.6 cm. by 35.6 cm. (8 1/2 by 14 inches)
   21.6 cm. by 33.1 cm. (8 1/2 by 13 inches)
   21.6 cm. by 27.9 cm. (8 1/2 by 11 inches)
   21.0 cm. by 29.7 cm. (DIN size A4)
   - ___ All drawing sheets not the same size. Sheet(s)___
   - ___ Drawing sheet not an acceptable size. Sheet(s) ___
6. MARGINS. 37 CFR 1.84(g): Acceptable margins:

Paper size

	21.6 cm. X 35.6 cm. (8 1/2 X 14 inches)	21.6 cm X 33.1 cm. (8 1/2 X 13 inches)	21.6 cm. X 27.9 cm. (8 1/2 X 11 inches)	21.0 cm. X 29.7 cm. (DIN Size A4)
T	5.1 cm. (2")	2.5 cm. (1")	2.5 cm. (1")	2.5cm.
L	.64 cm. (1/4")	.64 cm. (1/4")	.64 cm. (1/4")	2.5 cm.
R	.64 cm. (1/4")	.64 cm. (1/4")	.64 cm. (1/4")	1.5 cm.
B	.64 cm. (1/4")	.64 cm. (1/4")	.64 cm. (1/4")	1.0 cm.

Margins do not conform to chart above.
Sheet(s)___
X Top (T) X Left (L) ___ Right (R) ___ Bottom (B)

7. VIEWS. 37 CFR 1.84(h)
   REMINDER: Specification may require revision to correspond to drawing changes.
   - ___ All views not grouped together. Fig(s)___
   - ___ Views connected by projection lines or lead lines. Fig(s)___

   Partial views. 37 CFR 1.84(h) 2
   - ___ View and enlarged view not labled separatly or properly. Fig(s)___

   Sectional views. 37 CFR 1.84 (h) 3
   - ___ Hatching not indicated for sectional portions of an object. Fig(s)___
   - ___ Cross section not drawn same as view with parts in cross section with regularly spaced parallel oblique strokes. Fig(s)___
8. ARRANGEMENT OF VIEWS. 37 CFR 1.84(i)
   - ___ Words do not appear on a horizontal, left-to-right fashion when page is either upright or turned so that the top becomes the right side, except for graphs. Fig(s)___
9. SCALE. 37 CFR 1.84(k)
   - ___ Scale not large enough to show mechanism with crowding when drawing is reduced in size to two-thirds in reproduction. Fig(s)___
   - ___ Indication such as "actual size" or scale 1/2" not permitted. Fig(s)___
10. CHARACTER OF LINES, NUMBERS, & LETTERS. 37 CFR 1.84(l)
    - X Lines, numbers & letters not uniformly thick and well defined, clean, durable, and black (except for color drawings). Fig(s) 1-67
11. SHADING. 37 CFR 1.84(m)
    - ___ Solid black shading areas not permitted. Fig(s)___
    - ___ Shade lines, pale, rough and blurred. Fig(s)___
12. NUMBERS, LETTERS, & REFERENCE CHARACTERS. 37 CFR 1.84(p)
    - ___ Numbers and reference characters not plain and legible. 37 CFR 1.84(p)(l) Fig(s)___
    - ___ Numbers and reference characters not oriented in same direction as the view. 37 CFR 1.84(p)(l) Fig(s)___
    - ___ English alphabet not used. 37 CFR 1.84(p)(2) Fig(s)___
    - X Numbers, letters, and reference characters do not measure at least .32 cm. (1/8 inch) in height. 37 CFR(p)(3) Fig(s)___
13. LEAD LINES. 37 CFR 1.84(q)
    - ___ Lead lines cross each other. Fig(s)___
    - ___ Lead lines missing. Fig(s)___
14. NUMBERING OF SHEETS OF DRAWINGS. 37 CFR 1.84(t)
    - ___ Sheets not numbered consecutively, and in Arabic numerals, beginning with number 1. Sheet(s)___
15. NUMBER OF VIEWS. 37 CFR 1.84(u)
    - ___ Views not numbered consecutively, and in Arabic numerals, beginning with number 1. Fig(s)___
    - ___ View numbers not preceded by the abbreviation Fig. Fig(s)___
16. CORRECTIONS. 37 CFR 1.84(w)
    - ___ Corrections not made from prior PTO-948. Fig(s)___
17. DESIGN DRAWING. 37 CFR 1.152
    - ___ Surface shading shown not appropriate. Fig(s)___
    - ___ Solid black shading not used for color contrast. Fig(s)___

COMMENTS:

ATTACHMENT TO PAPER NO. 7 REVIEWER

REMINDER

Drawing changes may also require changes in the specification, e.g., if Fig. 1 is changed to Fig. 1A, Fig. 1B, Fig. 1C, etc., the specification, at the Brief Description of the Drawings, must likewise be changed. Please make such changes by 37 CFR 1.312 Amendment at the time of submitting drawing changes.

INFORMATION ON HOW TO EFFECT DRAWING CHANGES

1. Correction of Informalities--37 CFR 1.85

File new drawings with the changes incorporated therein. The application number or the title of the invention, inventor's name, docket number (if any), and the name and telephone number of a person to call if the Office is unable to match the drawings to the proper application, should be placed on the back of each sheet of drawings in accordance with 37 CFR 1.84(c). Applicant may delay filing of the new drawings until receipt of the Notice of Allowability (PTOL-37). Extensions of time may be obtained under the provisions of 37 CFR1.136. The drawing should be filed as a separate paper with a transmittal letter addressed to the Drawing Review Branch.

2. Timing of Corrections

Applicant is required to submit **acceptable** corrected drawings within the three-month shortened statutory period set in the Notice of Allowability (PTOL-37). If a correction is determined to be unacceptable by the Office, applicant must arrange to have acceptable correction resubmitted within the original three-month period to avoid the necessity of obtaining as extension of time and paying the extension fee. Therefore, applicant should file corrected drawings as soon as possible.

Failure to take corrective action within set (or extended) period will result in **ABANDONMENT** of the Application.

3. Corrections other than Informalities Noted by the Drawing Review Branch on the Form PTO 948

All changes to the drawings, other than informalities noted by the Drawing Review Branch, **MUST** be approved by the examiner before the application will be allowed. No changes will be permitted to be made, other than correction of informalities, unless the examiner has approved the proposed changes.

Notice of References Cited	Application No. 08/674,610	Applicant(s) Heidorn et al.	
	Examiner Harold Zintel	Group Art Unit 2741	Page 1 of 1

U.S. PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS
	A					
	B			NONE		
	C					
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUBCLASS
	N						
	O						
	P						
	Q						
	R						
	S						
	T						

NON-PATENT DOCUMENTS

*		DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
	U		
	V		
	W		
	X		

* A copy of this reference is not being funished with this Office action.
(See Manual of Patent Examining Procedure, Section 707.05(a).)

U.S. Patent and Trademark Office
PTO-892 (Rev. 9-96)

Part of Paper No. 7

*U.S. GPO: 1997-417-381/62700

JUL 3 0 1998 JC49 PATENT & TRADEMARK OFFICE

PATENT #8 MJJ 8-6-98

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

July 27, 1998
Date

Maurice J. Pirio

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Examiner	:	H. Zintel
Art Unit	:	2741
Docket No.	:	661005.447
Date	:	July 27, 1998

RECEIVED 98 AUG -5 AM 8:47

Assistant Commissioner for Patents
Washington, DC 20231

PETITION FOR AN EXTENSION OF TIME
UNDER 37 C.F.R. § 1.136(a)

Sir:

Applicants herewith petition the Assistant Commissioner of Patents under 37 C.F.R. § 1.136(a) for a 1-month extension of time for filing the response to the Examiner's Action dated March 27, 1998, from June 27, 1998 to July 27, 1998. Submitted herewith is a check in the amount of $1,054 (including $110 to cover the cost of the extension.)

Any deficiency or overpayment should be charged or credited to Deposit Account No. 19-1090. This petition is being submitted in triplicate.

08/03/1998 AIBRAHIM 00000015 08674610

03 FC:115 110.00 OP

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio
Registration No. 33,273

Enclosures:
Postcard
Check
Two copies of this Petition

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900 Fax: (206) 682-6031

FORM PTO-1083

OIPE JC49 JUL 3 0 1998 PATENT & TRADEMARK OFFICE

SEED AND BERRY LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
Phone (206) 622-4900
Fax (206) 682-6031

6P2741 /$

Docket No.: **661005.447**
Date: **July 27, 1998**

RECEIVED AUG -5 1998 GROUP 2700

In re application of **George Heidorn and Karen Jensen**
Application No.: **08/674,610**
Filed: **June 28, 1996**
For: **METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES**

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON DC 20231

Sir:

Transmitted herewith is **an amendment** in the above-identified application.

[ ] Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
[ ] A verified statement to establish small entity status under CFR 1.9 and 1.27 is enclosed.
[**X**] A Petition for an Extension of Time for one month is enclosed.
[**X**] A General Authorization Under 37 C.F.R. § 1.136(a)(3) is enclosed.
[ ] No additional claim fee is required.
[**X**] The fee has been calculated as shown.

	(Col. 1)		(Col. 2)	(Col. 3)	SMALL ENTITY			OTHER THAN A SMALL ENTITY	
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST PREV. PAID FOR	PRESENT EXTRA	RATE	ADDITIONAL FEE	OR	RATE	ADDITIONAL FEE
TOTAL	* **64**	MINUS	** **36**	**28**	**x 11**	$		**x 22**	**$ 616**
INDEP.	* **11**	MINUS	*** **7**	**4**	**x 41**	$		**x 82**	**$ 328**
[ ] FIRST PRESENTATION OF MULTIPLE CLAIMS					**+ 135**	$	OR	**+ 270**	$
EXTENSION OF TIME FEE						$			**$ 110**
TOTAL ADDITIONAL FEE						$	TOTAL		**$ 1,054**

* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space.
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

[ ] Please charge my Deposit Account No. 19-1090 in the amount of $_. A duplicate copy of this sheet is enclosed.
[**X**] A check in the amount of $ **1,054** is attached.
[**X**] The Assistant Commissioner is hereby authorized to charge payment of the following additional fees associated with this communication or credit any overpayment to Deposit Account No. 19-1090. A duplicate copy of this sheet is enclosed.
[**X**] Any filing fees under 37 CFR 1.16 for the presentation of extra claims.
[**X**] Any patent application processing fees under 37 CFR 1.17.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED AND BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

OIPE JC49 JUL 30 1998 PATENT & TRADEMARK OFFICE

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Examiner	:	H. Zintel
Art Unit	:	2741
Docket No.	:	661005.447
Date	:	July 27, 1998

RECEIVED 98 AUG -5 AM 8:47 GROUP 2700

Assistant Commissioner for Patents
Washington, DC 20231

GENERAL AUTHORIZATION UNDER 37 C.F.R. § 1.136(a)(3)

Sir:

With respect to the above-identified application, the Assistant Commissioner is authorized to treat any concurrent or future reply requiring a petition for an extension of time under 37 C.F.R. § 1.136(a)(3) for its timely submission as incorporating a petition therefor for the appropriate length of time. The Assistant Commissioner is also authorized to charge any fees which may be required under 37 C.F.R. § 1.136(a)(3), or credit any overpayment, to Deposit Account No. 19-1090.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

SEED and BERRY LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900 FAX: (206) 682-6031

9/A
MDJ
PATENT 8-6-98

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

July 27, 1998
Date

Maurice J Pirio
Maurice J. Pirio

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Examiner	:	H. Zintel
Art Unit	:	2741
Docket No.	:	661005.447
Date	:	July 27, 1998

Assistant Commissioner for Patents
Washington, DC 20231

AMENDMENT

Sir:

In response to the Office Action dated March 27, 1998, please extend the period of time for response one month, to expire on July 27, 1998. Enclosed are a General Authorization, a Petition for an Extension of Time, and the requisite fee. Please amend the application as follows:

In the Claims:

Please amend claims 1, 12, and 27-29 as follows.

A1

1. (Amended) A method in a computer system for generating a logical form graph for a sentence in a natural language, the sentence having words that are ordered, the sentence being represented by a syntax parse tree having nodes representing syntactic constructs of the sentence, the syntax parse tree being represented in a data structure and having leaf nodes representing each word of the sentence, the leaf nodes being reordered in the same order as the words of the sentence, the method comprising:

adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence;

adjusting the syntax parse tree with the added syntactic roles, wherein the leaf nodes are reordered to represent a more complete syntactic analysis of the sentence;

generating a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is separate from the data structure of the syntax parse tree;

adding semantic labels to the generated skeletal logical form graph; and

adjusting the logical form graph with semantic labels to add semantic constructs to complete the logical form graph.

A2

12. (Amended) The method of claim 1 wherein the step of adjusting the syntax parse tree includes replacing the word "it" with [and] an infinitive clause.

A3

27. (Amended) A computer system for generating a logical form graph for a sentence in a natural language, the sentence having words that are ordered, the sentence being represented by a syntax parse tree having nodes representing syntactic constructs of the sentence, the syntax parse tree being represented in a data structure and having leaf nodes representing each word of the sentence, the leaf nodes being ordered in the same order as the words of the sentence, the method comprising:

a phase one component for adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence and for adjusting the syntax parse tree with the added syntactic roles wherein the leaf nodes are reordered to represent a complete syntactic analysis of the sentence;

a phase two component for generating a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is separate from the data structure of the syntax parse tree, the logical form graph having nodes and links, the nodes corresponding to semantic constructs and the links corresponding to relationships between semantic constructs; and

a phase three component for adding semantic labels to the generated skeletal logical form graph and for adjusting the logical form graph with semantic labels to add semantic constructs to complete the logical form graph.

28. (Amended) A method in a computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the method comprising the steps of:

generating a first data structure for a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

generating a [separate] second data structure for a logical form graph to represent a semantic analysis of the input text, the second data structure being generated from the syntax parse tree but being a separate data structure from the first data structure.

29. (Amended) A computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the system comprising:

a component that generates a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

a component that generates a [separate] logical form graph to represent a semantic analysis of the input text, [wherein the logical form graph comprises nodes and directional links] the logical form graph being stored in a data structure that is separate from a data structure in which the generated syntax parse tree is stored, the logical form graph being generated based in part on the generated syntax parse tree.

Please add the following new claims.

-- 37. A method in a computer system for generating a syntax parse tree for a sentence, the sentence having words that are ordered, the method comprising:

generating a syntax parse tree with leaf nodes representing the words of the sentence and intermediary nodes representing syntactic constructs, the leaf nodes being ordered in the same order as the words of the sentence; and

altering the order of the leaf nodes of the generated syntax parse tree to reflect a more complete understanding of the syntax of ~~the sentences.~~

38. The method of claim 37 wherein the altering includes adding leaf nodes for syntactic roles that are implicit in the sent~~ence.~~

39. The method of claim 37 wherein the altering includes resolving long-distance attachment phenomena.

40. The method of claim 37 wherein the altering includes transforming verbal phrases into verbs with prepositional phrase objects.

41. The method of claim 37 wherein the altering includes replacing the word "it" with an infinitive clause.

42. A method in a computer system for generating a syntax parse tree for a sentence, the sentence having words, the method comprising:

generating a syntax parse tree with leaf nodes representing the words of the sentence and intermediary nodes representing syntactic constructs; and

adding nodes to the syntax parse tree to represent syntactic roles that are implicit in the sentence.

43. The method of claim 42 wherein when the sentence omits a verb after a predefined word, adding a node for the omitted verb.

44. The method of claim 43 wherein the predefined word is the word "to."

45. The method of claim 43 wherein the predefined word is the word "not."

46. The method of claim 42 wherein when the sentence is missing a pronoun, adding a node for the missing pronoun.

47. The method of claim 46 wherein the missing pronoun is the word "you" and the sentence is an imperative sentence.

48. The method of claim 42 wherein when the sentence includes coordinate structures, adding nodes to expand the coordinate structure.

49. The method of claim 48 wherein the coordinate structures include the word "and."

50. The method of claim 48 wherein the coordinate structures include the word "or."

51. A computer-readable medium containing instructions for causing a computer system to modify a syntax parse tree for a sentence, the sentence having words that are ordered, the syntax parse tree having leaf nodes representing the words of the sentence and intermediary nodes representing syntactic constructs, the leaf nodes being ordered in the same order as the words of the sentence, the modifying including altering the order of the leaf nodes of the syntax parse tree to reflect a more ~~complete~~ understanding of the syntax of the sentences.

52. The computer-readable medium of claim 51 wherein the altering includes adding leaf nodes for syntactic roles that are implicit in the sentence.

53. The computer-readable medium of claim 51 wherein the altering includes resolving long-distance attachment phenomena.

54. The computer-readable medium of claim 51 wherein the altering includes transforming verbal phrases into verbs with prepositional phrase objects.

55. The computer-readable medium of claim 51 wherein the altering includes replacing the word "it" with an infinitive clause.

56. A computer-readable medium containing instructions for causing a computer system to modify a syntax parse tree for a sentence, the sentence having words, the syntax parse tree having leaf nodes representing the words of the sentence and intermediary nodes representing syntactic constructs by adding leaf nodes to the syntax parse tree to represent syntactic roles that are implicit in the sentence.

57. The computer-readable medium of claim 56 wherein when the sentence omits a verb after a predefined word, adding a leaf node for the omitted verb.

58. The computer-readable medium of claim 57 wherein the predefined word is the word "to."

59. The computer-readable medium of claim 57 wherein the predefined word is the word "not."

60. The computer-readable medium of claim 56 wherein when the sentence is missing a pronoun, adding a leaf node for the missing pronoun.

61. The computer-readable medium of claim 60 wherein the missing pronoun is the word "you" and the sentence is an imperative sentence.

62. The computer-readable medium of claim 56 wherein when the sentence includes coordinate structures, adding leaf nodes to expand the coordinate structure.

63. The computer-readable medium of claim 62 wherein the coordinate structures include the word "and."

64. The computer-readable medium of claim 62 wherein the coordinate structures include the word "or." --

REMARKS

Claims 1-20, 27-30, and 37-64 are now pending in the application. Applicants have amended claims 1, 12, and 27-29, canceled claims 21-26 and 31-36, and added claims 37-64 to clarify the subject matter of that applicants regard as their invention.

Applicants' invention is directed to several aspects of techniques for generating a logical form graph. One technique alters the ordering of the leaf nodes of a syntax parse tree. Traditionally, a syntax parse tree contains one leaf node for each word of the sentence, and the leaf nodes are ordered in the same order as the words of the sentence. Applicants' technique reflects a more complete understanding of the sentence by reordering the leaf nodes of the syntax parse tree. This reordering of the leaf nodes of the syntax sparse tree facilitates the generation of the logical form graph. In another aspect of applicants' technique, leaf nodes are added to the syntax parse tree to represent syntactic roles that are implicit in the sentence, that is, the sentence contains no word explicit to that role. These added nodes may represent a missing pronoun such as the word "you" or a missing verb after the words "to" or "not." In another aspect, applicants' technique generates the logical form graph as a data structure that is separate from the syntax parse tree. Prior techniques, in contrast, imposed the logical form graph data structure on the nodes of the syntax parse tree. Such imposition of one data structure upon another resulted and added complexity when generating a logical form graph.

The Examiner has rejected claims 1-36 under 35 U.S.C. § 102(b) as being anticipated by the Jensen reference. Although applicants disagree, applicants have amended the claims to clarify the subject matter that applicants regard as their invention. The Jensen reference on page 3 describes a component which generates "corrected syntax." However, the

Jensen reference neither teaches nor suggests that the "corrected syntax" includes any reordering of the leaf nodes of the syntax parse tree or any adding of leaf nodes to the syntax parse tree. Moreover, the Jensen reference describes techniques that transform a syntax parse tree into a logical form graph so that a common data structure is used to store both the syntax parse tree and the logical form graph. As such, the Jensen reference neither teaches nor suggests that a separate data structure is used for the syntax parse tree and the logical form graph.

Claims 1-20 and 27 now recite "wherein the leaf nodes are reordered to represent a more complete syntactic analysis of the sentence." As discussed above, the Jensen reference does not describe reordering of leaf nodes. Claims 28-30 now make it particularly clear that the logical form graph and the syntax parse tree are stored in separate data structures. The Jensen reference only describes the use of a single data structure which represents both the syntax parse tree and the logical form graph. Newly added claims 37-64 either recite altering the ordering of the nodes or adding of leaf nodes to the syntax parse tree. The Jensen reference neither teaches nor suggests such altering or adding.

The Examiner has rejected claim 12 under 35 U.S.C. § 112, second paragraph, as being indefinite. Applicants have amended claim 12 to correct a minor typographical error.

Based upon the above remarks and amendments, applicants respectfully request reconsideration of this application and its early allowance.

Respectfully submitted,

George Heidorn and Karen Jensen

SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

Enclosures:
- Postcard
- Check
- Form PTO-1083 (+ copy)
- Petition for an Extension of Time (+ 2 copies)
- General Authorization

6300 Columbia Center, 701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900 Fax: (206) 682-6031

WPN/MS/661005/447-AM/V1

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, DC 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
08/674,610	06/28/96	HEIDORN	B 661005.447

LM02/0915

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

EXAMINER	
ZINTEL, H	
ART UNIT	PAPER NUMBER
2741	16

DATE MAILED: 09/15/98

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (REV. 2/95)

1-File Copy

☆U.S.GOVERNMENT PRINTING OFFICE 1995-319-826

Office Action Summary	Application No. 08/674,610	Applicant(s) Heidorn et al
	Examiner Harold Zintel	Group Art Unit 2741

☒ Responsive to communication(s) filed on *Jul 30, 1998*.

☒ This action is **FINAL.**

☐ Since this application is in condition for allowance except for formal matters, **prosecution as to the merits is closed** in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11; 453 O.G. 213.

A shortened statutory period for response to this action is set to expire 3 month(s), or thirty days, whichever is longer, from the mailing date of this communication. Failure to respond within the period for response will cause the application to become abandoned. (35 U.S.C. § 133). Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

Disposition of Claims

☒ Claim(s) *1-20, 27-30, and 37-64* is/are pending in the application.

Of the above, claim(s) ______ is/are withdrawn from consideration.

☐ Claim(s) ______ is/are allowed.

☒ Claim(s) *1-20, 27-30, and 37-64* is/are rejected.

☐ Claim(s) ______ is/are objected to.

☐ Claims ______ are subject to restriction or election requirement.

Application Papers

☐ See the attached Notice of Draftsperson's Patent Drawing Review, PTO-948.

☒ The drawing(s) filed on *Jun 28, 1996* is/are objected to by the Examiner.

☐ The proposed drawing correction, filed on ______ is ☐approved ☐disapproved.

☒ The specification is objected to by the Examiner.

☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

☐ Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).

☐ All ☐ Some* ☐ None of the CERTIFIED copies of the priority documents have been

☐ received.

☐ received in Application No. (Series Code/Serial Number) ______.

☐ received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

*Certified copies not received: ______.

☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

☐ Notice of References Cited, PTO-892

☐ Information Disclosure Statement(s), PTO-1449, Paper No(s). ______

☐ Interview Summary, PTO-413

☐ Notice of Draftsperson's Patent Drawing Review, PTO-948

☐ Notice of Informal Patent Application, PTO-152

--- *SEE OFFICE ACTION ON THE FOLLOWING PAGES* ---

DETAILED ACTION

Drawings

1. Figure 1-23 should be designated by a legend such as --Prior Art-- because only that which is old is illustrated. See MPEP § 608.02(g).

Claim Rejections - 35 USC § 112

2. Claims 1, 27, 37, 51 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. In lines 9 and 10 (in claim 1, same wording in other claims) the step of reordering is done "to represent a more complete syntactic analysis". How can reordering per se make an analysis more complete?

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless --
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1-64 are rejected under 35 U.S.C. 102(b) as being disclosed by Jensen et al. "Natural Language Processing: The PLNLP Approach" included in the information disclosure statement. Jensen teaches the following features:

a. As per claims 1, 27, 37, 51 and 56:

i. parse tree with the same order as the original sentence, in figure 1 on page 205

ii. adding syntactic rules, on page 205, as deep argument attributes are added to the analysis record structure

iii. adjusting or reordering the syntax parse tree, on page 4, since relationships are to be normalized, sentences that mean the same thing are represented in the same form, adjusting or reordering is inherent. As in figures 1 and 2 on page 205 the leaves are reordered as the graph is created.

iv. generating a separate skeletal logical form, on page 4 in *Derivation of logical form* section

b. As per claims 2, 3, 44, 52, 57 and 58: adding syntactic constructs for omitted verb after a predefined word, on page 212, as "fill in all missing arguments"

c. As per claim 5 and 46: a syntactic construct for a pronoun, figure 4 page 209.

d. As per claims 7 and 8, 48, 49, 62 and 63: syntactic construct for coordinate structures, in figure 5 on page 209, in this case "and"

e. As per claim 10, 39 and 53: resolving long-distance attachment phenomena, on page 207 the paragraph above figure 3.

f. As per claim 13: logical form graph based on syntax parse tree, page 204 first paragraph

g. As per claims 14-20: deep parts of speech, on pages 205 and 206

h. As per claim 28 and 29: syntax parse tree and logical form graph, in figures 1 and 2 on page 205, both of which a speaker of the natural language can understand

i. As per claim 30: a first and second sub-components, on page 4 in *Derivation of logical form* section, taking step "a" as the first sub-component and "b" and "c" together as the second sub-component

j. As per claim 38: add syntactic roles, on page 204 second paragraph and page 211 as "filled structure

k. As per claim 42: Adding nodes that are implicit, in figure 5, as missing information

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 4, 6, 9, 11, 12, 41, 40, 45, 47, 50, 54, 55, 59, 60, 61, 64 rejected under 35 U.S.C. 103(a) as being unpatentable over Jensen as applied to claim 1.

a. As per claims 4, 45 and 59: Jensen teaches the step of adding a syntactic construct for a omitted verb after the word "to", as discussed in claim 3 but not after the word "not". It would have been obvious at the time the invention was made to a person having ordinarily skill in the art to use the same type of construct for a omitted verb after the word "not" since both are defined as "omitted" and are thus needed to form the complete logical graph of the sentence.

b. As per claims 9, 50 and 64: Jensen teaches the step of adding syntactic constructs for the word "and" as discussed in claim 8, but not the word "or". It would have been obvious at the time the invention was made to a person having ordinarily skill in the art to use the same type of construct for the word "or" since both are coordinate structures and are needed to form the complete logical graph of the.

c. As per claims 11, 40 and 54: Jensen teaches transforming a syntax parse tree into a logical graph but, does not teach transforming verbal phrases into verbs with prepositional phrase objects. It would have been obvious at the time the invention was made to a person having ordinarily skill in the art to first do this transformation in order to form a logical graph because it is well known in the art these form equivalent parse trees.

d. As per claims 12, 41 and 55: Jensen teaches transforming a syntax parse tree into a logical graph but does not teach replacing "it" with an infinitive clause. It is well known in the art these form equivalent parse trees. As per claim 32 and 33: add syntactic roles, on page 204 second paragraph

e. As per claim 6, 47, 60 and 61: It is well known in the art to supply you in a imperative sentence.

Response to Amendment

7. Applicant's arguments filed 7/30/98 have been fully considered but they are not all persuasive.

a. 112 2 nd paragraph rejection of claim 12 is withdrawn in view of amendment of 7/30/98.

b. Attorney argues that figures 1-23 are not prior art although the examiner sees no distinguishing features between these drawing and the figures in Jensen.

c. Attorney argues that the nodes are reordered in the traditional manner, thus this feature is not novel.

d. Attorney argues that adding nodes for an implicit pronoun is novel. But, since speakers of the language would understand to add the implied pronouns when deciphering a spoken phrase it would be obvious to add a node for that implied pronoun. Further Jensen on page 212 teaches the step of "fill(ing) in missing arguments"

e. Attorney argues that the prior art does not show the logical form graph and the parse tree in separate data structures. But Jensen on page 204 teaches a parse tree as input and a logical form as output thus separating the two.

Conclusion

8. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

9. ***Conclusion***

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 308-9051, (for formal communications intended for entry)

Or:

(703) 305-9508 (for informal or draft communications, please label

PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA., Sixth Floor (Receptionist).

Any inquiry concerning this communication from the examiner should be directed to Harold A. Zintel whose telephone number is (703) 305-2381. The examiner can normally be reached on Monday-Friday from 8:30 a.m.-5:00 p.m.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, David Hudspeth, can be reached at (703) 305-4825.

The facsimile phone number for the Art Unit is (703) 305-9508. Alternately, facsimile messages may be sent directly to (703) 305-9644 where they will be stored in the examiner's voice mailbox (telling the examiner that a fax was received) and be automatically printed (i.e. - no delay by examiner).

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-3900.

Harold A. Zintel

Assistant Patent Examiner

September 11, 1998

DAVID R. HUDSPETH
SUPERVISORY PATENT EXAMINER
GROUP 2700

1

11/B

PATENT

I hereby certify that this paper is being facsimile transmitted to the U.S. Patent and Trademark Office on the date shown below.

MDJ
12-16-98

December 14, 1998
Date

Maurice J. Pirio

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Examiner	:	Harold A. Zintel
Art Unit	:	2741
Docket No.	:	661005.447
Date	:	December 14, 1998

Assistant Commissioner for Patents
Washington, DC 20231

AMENDMENT

Sir:

In response to the Office Action dated September 15, 1998, please amend the application as follows:

In the Claims:

Please cancel claims 1-20, 27, and 31-64.

Please amend claims 21, 24, 28, and 29 as follows:

B1

21. (Amended) A method in a computer system for generating a logical form graph for a phrase of words specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the computer system having a memory the method comprising:

generating in the memory an initial syntax parse tree of the phrase based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of the words of the phrase;

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the phrase;

generating in the memory a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph.

B2

24. (Amended) A computer-readable medium containing instructions for causing a computer system to generate a logical form graph for a sentence specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the computer system having an initial syntax parse tree of the sentence that represents a parse of the sentence based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of words of the sentence, the initial syntax parse tree being stored in memory of the computer system by:

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the sentence;

generating in memory of the computer system a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph for the sentence.

28. (Twice Amended) A method in a computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the method comprising the steps of:

generating in memory of the computer system a first data structure for a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

generating in memory of the computer system a second data structure for a logical form graph to represent a semantic analysis of the input text, the second data structure being generated from the syntax parse tree but being a separate data structure from the first data structure.

29. (Twice Amended) A computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the system comprising:

a component that generates in memory of the computer system a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

a component that generates in memory of the computer system a logical form graph to represent a semantic analysis of the input text, the logical form graph being stored in a data structure that is separate from a data structure in which the generated syntax parse tree is stored, the logical form graph being generated based in part on the generated syntax parse tree.

REMARKS

Claims 21-26 and 28-30 are now pending. Applicants have canceled claims 1-20, 27, and 31-64 and amended claims 21, 24, and 28-29 to clarify the subject matter of their invention.

Applicants would like to thank the Examiner for his consideration during the telephone interview of December 11, 1998. During that interview, applicants' representative and the Examiner discussed the concept of generating a logical form graph that is a separate data structure from the syntax parse tree from which the logical form graph is derived. The pending claims are directed to this concept. In addition, applicants' representative and the Examiner discussed whether claim 28, before being amended, encompassed the display of a syntax parse tree and a logical form graph as shown on page 205 of the Jensen reference.

The Examiner rejected claims 21-26 and 28-30 under 35 U.S.C. § 102(b) as being anticipated by the Jensen reference. It is the Examiner's position that "Jensen on page 204 teaches a parse tree as input and a logical form as output thus separating the two." (Examiner's Action, September 15, 1998, page 7). Although applicants believe that the unamended claims did not encompass the display of a syntax parse tree and a logical form graph, applicants nevertheless have amended the claims to make it particularly clear that the generated syntax parse tree and the generated logical form graphs are stored in the memory of computer system. For example, claim 21 now recites "generating <u>in the memory</u> an initial syntax parse tree" and "generating <u>in the memory</u> a skeletal logical form graph." Thus, the amended claims clearly do not encompass the display of a syntax parse tree and a logical form graph as shown on page 205 of the Jensen reference.

Each of the pending claims also recites that the logical form graph is stored in a data structure that is "separate" from or "independent" of the syntax parse tree. For example, claim 21 recites "the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree," and claim 28 recites "the second data structure being generated from the syntax parse tree being a separate data structure from the first data structure." The Jensen reference, in contrast, describes that the data structure for the logical form graph uses the same underlying records that are used to represent the syntax parse tree. In particular, the Jensen reference at page 204 states:

> A graph is produced by displaying only those attributes and values that are defined to be semantic. However, <u>the underlying record structure contains all</u> the attributes resulting from the parse.

Thus, it is clear that Jensen's underlying record structure contains information for both the logical form graph and the syntax parse tree.

Applicants further emphasized this combined data structure aspect of the prior art in the background section of the application. For example, the background states that "the logical form graph is constructed from the nodes of the syntax parse tree, adding to them attributes and new bi-directional links." (Specification, p. 10.) Applicants also pointed out in the background section of the application the disadvantage of representing the logical form graph using the same data structure that is used to represent the syntax parse tree. In particular, the background states that:

> because nodes of the syntax parse tree are extended and reused as nodes of the logical form graph, prior art semantic subsystems produce large, cumbersome, and complicated data structures. The size and complexity of a logical form graph overlayed [sic] onto a syntax parse tree makes further use of the combined data structure error-prone and inefficient.

(Specification, pages 11-12.) Thus, applicants' separation of the syntax parse tree data structure from the logical form data structure avoids these disadvantages of the prior art.

As requested by the Examiner, applicants are adding the legend "prior art" to Figures 1-23. The drawings are being filed under a separate cover.

Based upon the above amendments and remarks, applicants respectfully request reconsideration of this application and its early allowance.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

Enclosures:
Form PTO-1083 (+ copy)

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

WPN/MS/661005/FAXED AMENDMENT (12-14-98)

SEED AND BERRY LLP
6300 Columbia Center
Seattle, Washington 98104-7092, U.S.A.
FAX: (206) 682-6031 / PHONE: (206) 622-4900

Official FAX RECEIVED
DEC 15 1998
Group 2700

FACSIMILE TRANSMITTAL

DATE: December 14, 1998 OUR REF. NO: 661005.447

TO: Examiner H. Zintel, Art Unit 2741

FROM: Maurice J. Pirio

YOUR REF. NO.: Serial No. 08/674,610

YOUR FAX NO.: (703) 308-9051

RE:

We are transmitting 8 pages (including this sheet). If transmission is incomplete, please call Victoria Sellers at (206) 622-4900 or fax our office at the number above.

COMMENTS

TO BE ENTERED

Please hand-deliver to Examiner Zintel
as soon as possible

Confirmation copy sent ____

DATE FAXED:

TIME FAXED:

BY:

FORM PTO-1083

SEED AND BERRY LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
Phone (206) 622-4900
Fax (206) 682-6031

FAX RECEIVED
Official DEC 1 5 1998
Group 2700

Docket No.: **661005.447**
Date: **December 14, 1998**

In re application of **George Heidorn and Karen Jensen**
Application No.: **08/674,610**
Filed: **June 28, 1996**
For: **METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES**

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON DC 20231

Sir:

Transmitted herewith is an **amendment** in the above-identified application.

[ ] Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
[ ] A verified statement to establish small entity status under CFR 1.9 and 1.27 is enclosed.
[ ] A Petition for an Extension of Time for month is enclosed.
[ ] A General Authorization Under 37 C.F.R. § 1.136(a)(3) is enclosed.
[**X**] No additional claim fee is required.
[ ] The fee has been calculated as shown.

	(Col. 1)		(Col. 2)	(Col. 3)	SMALL ENTITY			OTHER THAN A SMALL ENTITY	
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST PREV. PAID FOR	PRESENT EXTRA	RATE	ADDITIONAL FEE	OR	RATE	ADDITIONAL FEE
TOTAL	* 9	MINUS	** 64	0	x 9	$		x 18	$ 0
INDEP.	* 4	MINUS	*** 11	0	x 39	$		x 78	$ 0
[ ] FIRST PRESENTATION OF MULTIPLE CLAIMS					+ 130	$	OR	+ 260	$
EXTENSION OF TIME FEE						$			$
TOTAL ADDITIONAL FEE						$	TOTAL		$ 0

* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space.
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

[ ] Please charge my Deposit Account No. 19-1090 in the amount of $_. A duplicate copy of this sheet is enclosed.
[ ] A check in the amount of $_ is attached.
[X] The Assistant Commissioner is hereby authorized to charge payment of the following additional fees associated with this communication or credit any overpayment to Deposit Account No. 19-1090. A duplicate copy of this sheet is enclosed.
[X] Any filing fees under 37 CFR 1.16 for the presentation of extra claims.
[X] Any patent application processing fees under 37 CFR 1.17.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED AND BERRY LLP

Maurice J. Pirio
Registration No. 33,273

FORM PTO-1083

SEED AND BERRY LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
Phone (206) 622-4900
Fax (206) 682-6031

Official

Docket No.: 661005.447
Date: December 14, 1998

FAX RECEIVED
DEC 15 1998
Group 2700

In re application of: George Heidorn and Karen Jensen
Application No.: 08/674,610
Filed: June 28, 1996
For: METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON DC 20231

Sir:

Transmitted herewith is an amendment in the above-identified application.

[ ] Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
[ ] A verified statement to establish small entity status under CFR 1.9 and 1.27 is enclosed.
[ ] A Petition for an Extension of Time for month is enclosed.
[ ] A General Authorization Under 37 C.F.R. § 1.136(a)(3) is enclosed.
[X] No additional claim fee is required.
[ ] The fee has been calculated as shown.

	(Col. 1)		(Col. 2)	(Col. 3)	SMALL ENTITY			OTHER THAN A SMALL ENTITY	
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST PREV. PAID FOR	PRESENT EXTRA	RATE	ADDITIONAL FEE	OR	RATE	ADDITIONAL FEE
TOTAL	* 9	MINUS	** 64	0	x 9	$		x 18	$ 0
INDEP.	* 4	MINUS	*** 11	0	x 39	$		x 78	$ 0
[ ] FIRST PRESENTATION OF MULTIPLE CLAIMS					+ 130	$	OR	+ 260	$
EXTENSION OF TIME FEE						$			$
TOTAL ADDITIONAL FEE						$	TOTAL		$ 0

* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space.
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

[ ] Please charge my Deposit Account No. 19-1090 in the amount of $_. A duplicate copy of this sheet is enclosed.
[ ] A check in the amount of $_ is attached.
[X] The Assistant Commissioner is hereby authorized to charge payment of the following additional fees associated with this communication or credit any overpayment to Deposit Account No. 19-1090. A duplicate copy of this sheet is enclosed.
[X] Any filing fees under 37 CFR 1.16 for the presentation of extra claims.
[X] Any patent application processing fees under 37 CFR 1.17.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED AND BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

FAX RECEIVED
DEC 15 1998
Group 2700

SEED AND BERRY LLP
6300 Columbia Center
Seattle, Washington 98104-7092, U.S.A.
FAX: (206) 682-6031 / PHONE: (206) 622-4900

FACSIMILE TRANSMITTAL

Official

DATE: December 14, 1998 OUR REF. NO: 661005.447

TO: Examiner H. Zintel, Art Unit 2741

FROM: Maurice J. Pirio

YOUR REF. NO.: Serial No. 08/674,610

YOUR FAX NO.: (703) 308-9051

RE:

We are transmitting 8 pages (including this sheet). If transmission is incomplete, please call Victoria Sellers at (206) 622-4900 or fax our office at the number above.

COMMENTS

TO BE ENTERED

Please hand-deliver to Examiner Zintel
as soon as possible

Confirmation copy sent

DATE FAXED:

TIME FAXED:

BY:

FORM PTO-1083

SEED AND BERRY LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
Phone (206) 622-4900
Fax (206) 682-6031

Official

Docket No.: **661005.447**
Date: **December 14, 1998**

In re application of **George Heidorn and Karen Jensen**
Application No.: **08/674,610**
Filed: **June 28, 1996**
For: **METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES**

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON DC 20231

Sir:

Transmitted herewith is an amendment in the above-identified application.

[ ] Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
[ ] A verified statement to establish small entity status under CFR 1.9 and 1.27 is enclosed.
[ ] A Petition for an Extension of Time for month is enclosed.
[ ] A General Authorization Under 37 C.F.R. § 1.136(a)(3) is enclosed.
[X] No additional claim fee is required.
[ ] The fee has been calculated as shown.

	(Col. 1)		(Col. 2)	(Col. 3)	SMALL ENTITY			OTHER THAN A SMALL ENTITY	
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST PREV. PAID FOR	PRESENT EXTRA	RATE	ADDITIONAL FEE	OR	RATE	ADDITIONAL FEE
TOTAL	* 9	MINUS	** 64	0	x 9	$		x 18	$ 0
INDEP.	* 4	MINUS	*** 11	0	x 39	$		x 78	$ 0
[ ] FIRST PRESENTATION OF MULTIPLE CLAIMS					+ 130	$	OR	+ 260	$
EXTENSION OF TIME FEE						$			$
TOTAL ADDITIONAL FEE						$	TOTAL		$ 0

* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space.
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

[ ] Please charge my Deposit Account No. 19-1090 in the amount of $_. A duplicate copy of this sheet is enclosed.
[ ] A check in the amount of $_ is attached.
[X] The Assistant Commissioner is hereby authorized to charge payment of the following additional fees associated with this communication or credit any overpayment to Deposit Account No. 19-1090. A duplicate copy of this sheet is enclosed.
[X] Any filing fees under 37 CFR 1.16 for the presentation of extra claims.
[X] Any patent application processing fees under 37 CFR 1.17.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED AND BERRY LLP

Maurice J. Pirio
Registration No. 33,273

PATENT

I hereby certify that this paper is being facsimile transmitted to the U.S. Patent and Trademark Office on the date shown below.

December 14, 1998
Date

Maurice J. Pirio

FAX RECEIVED
DEC 15 1998
Group 2700

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Examiner	:	Harold A. Zintel
Art Unit	:	2741
Docket No.	:	661005.447
Date	:	December 14, 1998

Assistant Commissioner for Patents
Washington, DC 20231

AMENDMENT

Sir:

In response to the Office Action dated September 15, 1998, please amend the application as follows:

In the Claims:

Please cancel claims 1-20, 27, and 31-64.

Please amend claims 21, 24, 28, and 29 as follows:

28. (Twice Amended) A method in a computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the method comprising the steps of:

generating in memory of the computer system a first data structure for a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

generating in memory of the computer system a second data structure for a logical form graph to represent a semantic analysis of the input text, the second data structure being generated from the syntax parse tree but being a separate data structure from the first data structure.

29. (Twice Amended) A computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the system comprising:

a component that generates in memory of the computer system a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

a component that generates in memory of the computer system a logical form graph to represent a semantic analysis of the input text, the logical form graph being stored in a data structure that is separate from a data structure in which the generated syntax parse tree is stored, the logical form graph being generated based in part on the generated syntax parse tree.

REMARKS

Claims 21-26 and 28-30 are now pending. Applicants have canceled claims 1-20, 27, and 31-64 and amended claims 21, 24, and 28-29 to clarify the subject matter of their invention.

Applicants would like to thank the Examiner for his consideration during the telephone interview of December 11, 1998. During that interview, applicants' representative and the Examiner discussed the concept of generating a logical form graph that is a separate data structure from the syntax parse tree from which the logical form graph is derived. The pending claims are directed to this concept. In addition, applicants' representative and the Examiner discussed whether claim 28, before being amended, encompassed the display of a syntax parse tree and a logical form graph as shown on page 205 of the Jensen reference.

The Examiner rejected claims 21-26 and 28-30 under 35 U.S.C. § 102(b) as being anticipated by the Jensen reference. It is the Examiner's position that "Jensen on page 204 teaches a parse tree as input and a logical form as output thus separating the two." (Examiner's Action, September 15, 1998, page 7). Although applicants believe that the unamended claims did not encompass the display of a syntax parse tree and a logical form graph, applicants nevertheless have amended the claims to make it particularly clear that the generated syntax parse tree and the generated logical form graphs are stored in the memory of computer system. For example, claim 21 now recites "generating <u>in the memory</u> an initial syntax parse tree" and "generating <u>in the memory</u> a skeletal logical form graph." Thus, the amended claims clearly do not encompass the display of a syntax parse tree and a logical form graph as shown on page 205 of the Jensen reference.

Each of the pending claims also recites that the logical form graph is stored in a data structure that is "separate" from or "independent" of the syntax parse tree. For example, claim 21 recites "the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree," and claim 28 recites "the second data structure being generated from the syntax parse tree being a separate data structure from the first data structure." The Jensen reference, in contrast, describes that the data structure for the logical form graph uses the same underlying records that are used to represent the syntax parse tree. In particular, the Jensen reference at page 204 states:

> A graph is produced by displaying only those attributes and values that are defined to be semantic. However, <u>the underlying record structure contains all</u> the attributes resulting from the parse.

Thus, it is clear that Jensen's underlying record structure contains information for both the logical form graph and the syntax parse tree.

Applicants further emphasized this combined data structure aspect of the prior art in the background section of the application. For example, the background states that "the logical form graph is constructed from the nodes of the syntax parse tree, adding to them attributes and new bi-directional links." (Specification, p. 10.) Applicants also pointed out in the background section of the application the disadvantage of representing the logical form graph using the same data structure that is used to represent the syntax parse tree. In particular, the background states that:

> because nodes of the syntax parse tree are extended and reused as nodes of the logical form graph, prior art semantic subsystems produce large, cumbersome, and complicated data structures. The size and complexity of a logical form graph overlayed [sic] onto a syntax parse tree makes further use of the combined data structure error-prone and inefficient.

(Specification, pages 11-12.) Thus, applicants' separation of the syntax parse tree data structure from the logical form data structure avoids these disadvantages of the prior art.

As requested by the Examiner, applicants are adding the legend "prior art" to Figures 1-23. The drawings are being filed under a separate cover.

Based upon the above amendments and remarks, applicants respectfully request reconsideration of this application and its early allowance.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

Enclosures:
Form PTO-1083 (+ copy)

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

WPN/MS/661005/FAXED AMENDMENT (12-14-98)

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
08/674,610	06/28/96	HEIDORN G	661005.447

LM61/1228

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

EXAMINER	
ZINTEL,H	
ART UNIT	PAPER NUMBER
2741	12

DATE MAILED: 12/28/98

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev. 2/95)

1- File Copy

Advisory Action

Application No.	Applicant(s)	
08/674,610	Heidorn et al	
Examiner		**Group Art Unit**
Harold Zintel		2741

THE PERIOD FOR RESPONSE: [check only a) or b)]

a) ☒ expires 3 months from the mailing date of the final rejection.

b) ☐ expires either three months from the mailing date of the final rejection, or on the mailing date of this Advisory Action, whichever is later. In no event, however, will the statutory period for the response expire later than six months from the date of the final rejection.

Any extension of time must be obtained by filing a petition under 37 CFR 1.136(a), the proposed response and the appropriate fee. The date on which the response, the petition, and the fee have been filed is the date of the response and also the date for the purposes of determining the period of extension and the corresponding amount of the fee. Any extension fee pursuant to 37 CFR 1.17 will be calculated from the date of the originally set shortened statutory period for response or as set forth in b) above.

☐ Appellant's Brief is due two months from the date of the Notice of Appeal filed on ______________ (or within any period for response set forth above, whichever is later). See 37 CFR 1.191(d) and 37 CFR 1.192(a).

Applicant's response to the final rejection, filed on *Dec 14, 1998* **has been considered with the following effect, but is NOT deemed to place the application in condition for allowance:**

☒ The proposed amendment(s):

- ☐ will be entered upon filing of a Notice of Appeal and an Appeal Brief.
- ☒ will not be entered because:
  - ☐ they raise new issues that would require further consideration and/or search. (See note below).
  - ☐ they raise the issue of new matter. (See note below).
  - ☒ they are not deemed to place the application in better form for appeal by materially reducing or simplifying the issues for appeal.
  - ☐ they present additional claims without cancelling a corresponding number of finally rejected claims.

  NOTE: ______________

☐ Applicant's response has overcome the following rejection(s): ______________

☐ Newly proposed or amended claims ______________ would be allowable if submitted in a separate, timely filed amendment cancelling the non-allowable claims.

☐ The affidavit, exhibit or request for reconsideration has been considered but does NOT place the application in condition for allowance because: ______________

☐ The affidavit or exhibit will NOT be considered because it is not directed SOLELY to issues which were newly raised by the Examiner in the final rejection.

☒ For purposes of Appeal, the status of the claims is as follows (see attached written explanation, if any):

Claims allowed: ______________

Claims objected to: ______________

Claims rejected: *1-20, 27-30, and 37-64*

☐ The proposed drawing correction filed on ______________ ☐ has ☐ has not been approved by the Examiner.

☐ Note the attached Information Disclosure Statement(s), PTO-1449, Paper No(s). ______________.

☐ Other

DAVID R. HUDSPETH
SUPERVISORY PATENT EXAMINER
GROUP 2700

U. S. Patent and Trademark Office
PTO-303 (Rev. 8-95)

Advisory Action

Part of Paper No. 12

Interview Summary	Application No. 08/674,610	Applicant(s) Heidorn et al	
	Examiner Harold Zintel	Group Art Unit 2741	

All participants (applicant, applicant's representative, PTO personnel):

(1) *Harold Zintel* (3) ______

(2) *Maurice Pirio* (4) ______

Date of Interview *Dec 10, 1998*

Type: ☒ Telephonic ☐ Personal (copy is given to ☐ applicant ☐ applicant's representative).

Exhibit shown or demonstration conducted: ☐ Yes ☒ No. If yes, brief description:

Agreement ☐ was reached. ☒ was not reached.

Claim(s) discussed: *1, 27, and 28*

Identification of prior art discussed:
Natural Language Language Processing page205 figures 1 and 2

Description of the general nature of what was agreed to if an agreement was reached, or any other comments:
clarification of inventive features

(A fuller description, if necessary, and a copy of the amendments, if available, which the examiner agreed would render the claims allowable must be attached. Also, where no copy of the amendents which would render the claims allowable is available, a summary thereof must be attached.)

1. ☒ It is not necessary for applicant to provide a separate record of the substance of the interview.

Unless the paragraph above has been checked to indicate to the contrary, A FORMAL WRITTEN RESPONSE TO THE LAST OFFICE ACTION IS NOT WAIVED AND MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW. (See MPEP Section 713.04). If a response to the last Office action has already been filed, APPLICANT IS GIVEN ONE MONTH FROM THIS INTERVIEW DATE TO FILE A STATEMENT OF THE SUBSTANCE OF THE INTERVIEW.

2. ☐ Since the Examiner's interview summary above (including any attachments) reflects a complete response to each of the objections, rejections and requirements that may be present in the last Office action, and since the claims are now allowable, this completed form is considered to fulfill the response requirements of the last Office action. Applicant is not relieved from providing a separate record of the interview unless box 1 above is also checked.

Examiner Note: You must sign and stamp this form unless it is an attachment to a signed Office action.

G01/2741 #13 MD 3-24-99

Please type a plus sign (+) inside this box [+]

PTO/SB/29 (2/98)

FEB 18 1999 PATENT & TRADEMARK OFFICE

CONTINUED PROSECUTION APPLICATION (CPA) REQUEST TRANSMITTAL

RECEIVED FEB 23 1999 Group 2700

Submit an original, and a duplicate for fee processing

(only for Continuation or Divisional applications under 37 CFR § 1.53(d))

Address to:	Attorney Docket No.	661005.447
Box CPA	*First Named Inventor*	George Heidorn
Assistant Commissioner for Patents	*Examiner Name*	H. Zintel
Washington, DC 20231	*Group / Art Unit*	2741
	Express Mail Label No	EM150269464US

This is a request for a [ ] continuation or [ ] divisional application under 37 CFR § 1.53(d), (continued prosecution application (CPA)) of prior application number 08/674,610 filed on June 28, 1996, entitled METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

NOTES

FILING QUALIFICATIONS: *The prior application identified above must be a nonprovisional application that is either: (1) complete as defined by 37 CFR § 1.51(b, or (2) the national stage of an international application in compliance with 35 U.S.C. § 371. A Notice will be placed on a patent issuing from a CPA, except for reissues and designs, to the effect that the patent issued on a CPA and is subject to the twenty-year patent term provisions of 35 U.S.C. § 154(a)(2). Therefore, the prior application of a CPA may have been filed before, on or after June 8, 1995.*

C-I-P NOT PERMITTED: *A continuation-in-part application cannot be filed as a CPA under 37 CFR § 1.53(d), but must be filed under 37 CFR § 1.53(b).*

EXPRESS ABANDONMENT OF PRIOR APPLICATION: *The filing of this CPA is a request to expressly abandon the prior application as of the filing date of the request for a CPA. 37 C.F.R. § 1.53(b) must be used to file a continuation, divisional, or continuation-in-part of an application that is not to be abandoned.*

ACCESS TO PRIOR APPLICATION: *The filing of this CPA will be construed to include a waiver of confidentiality by the applicant under 35 U.S.C. § 122 to the extent that any member of the public who is entitled under the provisions of 37 CFR § 1.14 to access to, copies of, or information concerning, the prior application may be given similar access to, copies of, or similar information concerning, the other application or applications in the file jacket.*

35 U.S.C. § 120 STATEMENT: *In a CPA, no reference to the prior application is needed in the first sentence of the specification and none should be submitted. If a sentence referencing the prior application is submitted, it will not be entered. A request for a CPA is the specific reference required by 35 U.S.C. § 120 and to every application assigned the application number identified in such request, 37 CFR § 1.78(a)*

1. [+] Enter the unentered amendment previously filed on December 14, 1998 under 37 CFR § 1.116 in the prior nonprovisional application.
2. [ ] A preliminary amendment is enclosed.
3. This application is being filed by fewer than all the inventors named in the prior application, 37 CFR § 1.53(d)(4).
   a. [ ] ***DELETE*** the following inventor(s) named in the prior non-provisional application: ____
   b. [ ] The inventor(s) to be deleted are set forth on a separate sheet attached hereto.
4. [ ] A new power of attorney or authorization of agent (PTO/SB/81) is enclosed.
5. Information Disclosure Statement (IDS) is enclosed:
   a. [ ] PTO-1449
   b. [ ] Copies of IDS Citations

02/22/1999 DWONG 00000014 08674610 01 FC:101 760.00 02 FC:102 78.00

[Page 1 of 2]

(1) For	Claims (2) Number filed	(3) Number extra		(4) Rate		(5) Calculations
Basic Fee						$ 760
Total Claims	9 - 20* =	0	X	$	=	$ 0
Independent Claims	4 - 3** =	1	X	$ 78	=	$ 78
Extension of Time Fee (two months)					+	$ 380
TOTAL FEE						$ 1,218
*Reissue claims in excess of 20 and over original patent. **Reissue independent claims over original patent.						

6. Small Entity Status:
   a. ☐ A small entity statement is enclosed, if (b) and (c) do not apply.
   b. ☐ A small entity statement was filed in the prior nonprovisional application and such status is still proper and desired.
   c. ☐ Is no longer claimed.
7. The Assistant Commissioner is hereby authorized to credit overpayments or charge the following fees or insufficiencies in the following fees to Deposit Account No. 19-1090.
   a. [+] Fees Required Under 37 CFR § 1.16.
   b. [+] Fees Required Under 37 CFR § 1.17.
   c. ☐ Fees Required Under 37 CFR § 1.18.
8. [+] A check in the amount of $1,218 is enclosed.
9. [+] Other: Certificate of Express Mail

NOTE: **The prior application's correspondence address will carry over to this CPA UNLESS a new correspondence address is provided below.**

10. CORRESPONDENCE ADDRESS

Maurice J. Pirio
Seed and Berry LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900 *phone*
(206) 682-6031 *fax*

Respectfully submitted,

SIGNATURE Maurice J Pirio Date February 16, 1999

TYPED or PRINTED NAME Maurice J. Pirio REGISTRATION NO. 33,273

OIPE
FEB 16 1999
JC21
PATENT & TRADEMARK OFFICE

EXPRESS MAIL NO. EM150269464US

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
FEB 23 1999
Group 2700

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Examiner	:	Harold A. Zintel
Art Unit	:	2741
Docket No.	:	661005.447
Date	:	February 16, 1999

Box CPA
Assistant Commissioner for Patents
Washington, DC 20231

PETITION FOR AN EXTENSION OF TIME
UNDER 37 C.F.R. § 1.136(a)

Sir:

Applicants herewith petition the Assistant Commissioner of Patents under 37 C.F.R. § 1.136(a) for a two-month extension of time for filing the response to the Examiner's Action dated September 15, 1998, from December 15, 1998 to February 15, 1999 (Patent and Trademark Office official holiday.) Submitted herewith is a check in the amount of $1,218, including $380 to cover the cost of the extension.

Any deficiency or overpayment should be charged or credited to Deposit Account No. 19-1090. This petition is being submitted in triplicate.

02/22/1999 DVUONG 00000014 08674610
03 FC:116 380.00 OP

Respectfully submitted,
George Heidorn and Karen Jensen
SEED and BERRY LLP

Maurice J. Pirio

Maurice J. Pirio
Registration No. 33,273

Enclosures:
- Postcard
- Check
- Two copies of this Petition

6300 Columbia Center, 701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900 Fax: (206) 682-6031

OIPE
FEB 16 1999
PATENT & TRADEMARK OFFICE

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
FEB 23 1999
Group 2700

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Docket No.	:	661005.447
Date	:	February 16, 1999

Box CPA
Assistant Commissioner for Patents
Washington, DC 20231

CERTIFICATE OF MAILING BY "EXPRESS MAIL"

Sir:

I hereby certify that the enclosures listed below are being deposited with the United States Postal Service "EXPRESS MAIL Post Office to Addressee" service under 37 C.F.R. § 1.10, Mailing Label Certificate No. EM150269464US, on February 16, 1999, addressed to Box CPA, Assistant Commissioner for Patents, Washington, DC 20231.

Respectfully submitted,

SEED and BERRY LLP

Susan Johnson

Jeanette West / Brunetta Ishman / Kennethia Ishman
Susan JOHNSON

Enclosures:
- Postcard
- Check
- Petition for an Extension of Time (+ 2 copies)
- CPA Request Transmittal (+ copy)

1

FAX RECEIVED
MAR 0 4 1999
Group 2700

Official

PATENT

#14/C
8h

I hereby certify that this paper is being facsimile transmitted to the U.S. Patent and Trademark Office on the date shown below.

March 3, 1999
Date

Maurice J. Pirio

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

Art Unit	:	2741
Docket No.	:	661005.447
Date	:	March 3, 1999

Assistant Commissioner for Patents
Washington, DC 20231

PRELIMINARY AMENDMENT

Sir:

Please amend the above-identified application as follows:

In the Claims:

Please cancel claims 1-64.

Please add the following new claims 65-73.

C1 cont.

~~65.~~ 1 A method in a computer system for generating a logical form graph for a phrase of words specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the computer system having a memory the method comprising:

generating in the memory an initial syntax parse tree of the phrase based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of the words of the phrase;

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the phrase;

generating in the memory a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph.

~~66.~~ 2 The method of claim ~~65~~ 1 wherein the step of adjusting the initial syntax parse tree includes adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the phrase.

~~67.~~ 3 The method of claim ~~65~~ 1 wherein the step of adjusting the skeletal logical form graph includes adding semantic labels to the generated skeletal logical form graph.

~~68.~~ 4 A computer-readable medium containing instructions for causing a computer system to generate a logical form graph for a sentence specified in a natural

34

language, the natural language having a grammar specifying syntax of the natural language, the computer system having an initial syntax parse tree of the sentence that represents a parse of the sentence based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of words of the sentence, the initial syntax parse tree being stored in memory of the computer system by:

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the sentence;

generating in memory of the computer system a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

C1 Cont

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph for the sentence.

~~69~~ 5 The computer-readable medium of claim ~~68~~ 4 wherein the adjusting of the initial syntax parse tree includes adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence.

~~70~~ 6 The computer-readable medium of claim ~~68~~ 4 wherein adjusting of the skeletal logical form graph includes adding semantic labels to the generated skeletal logical form graph.

~~71~~ 7 A method in a computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the method comprising the steps of:

generating in memory of the computer system a first data structure for a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

35

generating in memory of the computer system a second data structure for a logical form graph to represent a semantic analysis of the input text, the second data structure being generated from the syntax parse tree but being a separate data structure from the first data structure.

8 ~~72~~. A computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the system comprising:

C1 Cont'd

a component that generates in memory of the computer system a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

a component that generates in memory of the computer system a logical form graph to represent a semantic analysis of the input text, the logical form graph being stored in a data structure that is separate from a data structure in which the generated syntax parse tree is stored, the logical form graph being generated based in part on the generated syntax parse tree.

9 ~~73~~. The system of claim 8 ~~72~~ wherein the component that generates a separate logical form graph comprises the following sub-components:

a first sub-component that generates an initial skeletal logical form graph; and

a second sub-component that identifies semantic roles for the nodes of the skeletal logical form graph and labels the directed links of the skeletal logical form graph to produce a final, complete logical form graph.

36

REMARKS

Claims 65-73 are now pending. Applicants have canceled all the claims that were previously pending.

Based on the above amendments, applicants respectfully request reconsideration of the application and its early allowance.

Respectfully submitted,

George Heidorn and Karen Jensen

SEED and BERRY LLP

Maurice J. P.

Maurice J. Pirio
Registration No. 33,273

Enclosure:
Form PTO-1083 (+ copy)

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

9200

Official FAX RECEIVED MAR 0 4 1999 GROUP 2700

SEED AND BERRY LLP

6300 Columbia Center
Seattle, Washington 98104-7092, U.S.A.
FAX: (206) 682-6031 / PHONE: (206) 622-4900

FACSIMILE TRANSMITTAL

DATE: March 3, 1999 OUR REF. NO: 661005.447

TO: Examiner Harold Zintel, Group Art Unit 2741

FROM: Maurice Pirio

YOUR REF. NO.: Serial No. 08/674,610

YOUR FAX NO.: (703) 308-9051

RE: ______

We are transmitting 8 pages (including this sheet). If transmission is incomplete, please call Victoria Sellers at (206) 622-4900 or fax our office at the number above.

COMMENTS

PLEASE ENTER AND HAND-DELIVER TO

EXAMINER ZINTEL AS SOON AS POSSIBLE

Confirmation copy sent ____

DATE FAXED: ______

TIME FAXED: ______

BY: ______

FORM PTO-1083

SEED AND BERRY LLP
6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
Phone (206) 622-4900
Fax (206) 682-6031

FAX RECEIVED
MAR 0 4 1999
Group 2700

Official

Docket No.: **661005.447**
Date: **March 3, 1999**

In re application of **George Heidorn and Karen Jensen**
Application No.: **08/674,610**
Filed: **June 28, 1996**
For: **METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES**

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON DC 20231

Sir:

Transmitted herewith is a **preliminary amendment** in the above-identified application.

[ ] Small entity status of this application under 37 CFR 1.9 and 1.27 has been established by a verified statement previously submitted.
[ ] A verified statement to establish small entity status under CFR 1.9 and 1.27 is enclosed.
[ ] A Petition for an Extension of Time for month is enclosed.
[ ] A General Authorization Under 37 C.F.R. § 1.136(a)(3) is enclosed.
[X] No additional claim fee is required.
[ ] The fee has been calculated as shown.

	(Col. 1)		(Col. 2)	(Col. 3)	SMALL ENTITY			OTHER THAN A SMALL ENTITY	
	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST PREV. PAID FOR	PRESENT EXTRA	RATE	ADDITIONAL FEE	OR	RATE	ADDITIONAL FEE
TOTAL	* 9	MINUS	** 64	0	x 9	$		x 18	$ 0
INDEP.	* 4	MINUS	*** 11	0	x 39	$		x 78	$ 0
[ ] FIRST PRESENTATION OF MULTIPLE CLAIMS					+ 130	$	OR	+ 260	$
EXTENSION OF TIME FEE						$			$
TOTAL ADDITIONAL FEE						$	TOTAL		$ 0

* If the entry in Col. 1 is less than the entry in Col. 2, write "0" in Col. 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, write "20" in this space.
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, write "3" in this space.
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found from the equivalent box in Col. 1 of a prior amendment or the number of claims originally filed.

[ ] Please charge my Deposit Account No. 19-1090 in the amount of $_. A duplicate copy of this sheet is enclosed.
[ ] A check in the amount of $_ is attached.
[X] The Assistant Commissioner is hereby authorized to charge payment of the following additional fees associated with this communication or credit any overpayment to Deposit Account No. 19-1090. A duplicate copy of this sheet is enclosed.

[X] Any filing fees under 37 CFR 1.16 for the presentation of extra claims.
[X] Any patent application processing fees under 37 CFR 1.17.

Respectfully submitted,
George Heidorn and Karen Jensen
SEED AND BERRY LLP

Maurice J Pirio

Maurice J. Pirio
Registration No. 33,273

UNITED STAT.. DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR		ATTORNEY DOCKET NO.
08/674,610	06/28/96	HEIDORN	G	661005.447

LM41/0427

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

EXAMINER	
THOMAS, J	
ART UNIT	PAPER NUMBER
2747	15

DATE MAILED: 04/27/99

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev. 2/95)
*U.S. GPO: 1996-404-498/40510

1- File Copy

Notice of Allowability	Application No. 08/674,610	Applicant(s) HEIDORN, et al.
	Examiner JOSEPH THOMAS	Group Art Unit 2747

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance and Issue Fee Due or other appropriate communication will be mailed in due course.

☒ This communication is responsive to *CPA filed 2/16/99 and preliminanry amendment filed 3/3/99* .

☒ The allowed claim(s) is/are *65-73, now renumbered 1-9* .

☐ The drawings filed on ______ are acceptable.

☐ Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).

☐ All ☐ Some* ☐ None of the CERTIFIED copies of the priority documents have been

☐ received.

☐ received in Application No. (Series Code/Serial Number) ______ .

☐ received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

*Certified copies not received: ______ .

☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

A SHORTENED STATUTORY PERIOD FOR RESPONSE to comply with the requirements noted below is set to EXPIRE **THREE MONTHS** FROM THE "DATE MAILED" of this Office action. Failure to timely comply will result in ABANDONMENT of this application. Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

☐ Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL APPLICATION, PTO-152, which discloses that the oath or declaration is deficient. A SUBSTITUTE OATH OR DECLARATION IS REQUIRED.

☒ Applicant MUST submit NEW FORMAL DRAWINGS

☐ because the originally filed drawings were declared by applicant to be informal.

☒ including changes required by the Notice of Draftsperson's Patent Drawing Review, PTO-948, attached ~~hereto or~~ to Paper No. 7 .

☐ including changes required by the proposed drawing correction filed on ______ , which has been approved by the examiner.

☒ Including changes required by the attached Examiner's ~~Amendment~~/Comment.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the reverse side of the drawings. The drawings should be filed as a separate paper with a transmittal lettter addressed to the Official Draftsperson.

☐ Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Any response to this letter should include, in the upper right hand corner, the APPLICATION NUMBER (SERIES CODE/SERIAL NUMBER). If applicant has received a Notice of Allowance and Issue Fee Due, the ISSUE BATCH NUMBER and DATE of the NOTICE OF ALLOWANCE should also be included.

Attachment(s)

☒ Notice of References Cited, PTO-892

☐ Information Disclosure Statement(s), PTO-1449, Paper No(s). ______

☐ Notice of Draftsperson's Patent Drawing Review, PTO-948

☐ Notice of Informal Patent Application, PTO-152

☐ Interview Summary, PTO-413

☒ Examiner's ~~Amendment~~/Comment

☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material

☒ Examiner's Statement of Reasons for Allowance

EXAMINER'S COMMENT & REASONS FOR ALLOWANCE

(Attachment to Paper # 15)

Continued Prosecution Application

1. The request filed on 2/16/99 for a Continued Prosecution Application (CPA) under 37 CFR 1.53(d) based on parent Application No. 08/674,610 is acceptable and a CPA has been established. The preliminary amendment filed 3/3/99 has been entered. An action on the CPA follows.

Reasons for Allowance

2. The following is an Examiner's Statement of Reasons for Allowance:

The prior art of record fails to teach or fairly suggest, either singly or in combination, a computer-implemented method, computer readable medium, or computer system for generating a logical form graph for a word phrase specified in a natural language, by generating and adjusting in memory a syntax parse tree to identify syntactic constructs/analysis and a skeletal logical form graph to identity semantic constructs/analysis, wherein the syntax parse tree and the skeletal logical form graph are represented as independent and separate data structures within the memory, in the specific manner and combinations recited in independent claims 65, 68, 71, and 72 (now renumbered claims 1, 4, 7, and 8, respectively). The Examiner interprets the claimed "logical form graph" to be a labeled, directed graph representing semantic information and not having hierarchical ordering, as specifically defined at page 10, line 3 to page 11, line 12 of the specification and as depicted in, for example, figure 23 of the drawings.

Claims 66-67, 69-70, and 73 (now renumbered as claims 2-3, 5-6, and 9, respectively), incorporate the above features through dependency, and likewise distinguish over the prior art of record.

Any comments considered necessary by Applicant must be submitted no later than the payment of the Issue Fee and, to avoid processing delays, should preferably **accompany** the Issue Fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Citation of References

3. Attached to this Examiner's Reasons for Allowance is the citation of several reference, namely U.S. Patents to Nunberg, et al. (5,111,398); Hedin, et al. (5,386,556); and Nagao, et al. (5,424,947). These patents generally teach various systems and

methods for natural language processing and analysis having syntactic and/or semantic analyzers and parse trees. The prior art made of record and not relied upon is considered pertinent to Applicant's disclosure as background material and is not of particular significance. These prior art patents fail to teach or fairly suggest the novel and non-obvious features of the instant claims, as described above in section 1. In particular, the newly cited prior art of record fails to disclose generating and adjusting in memory a syntax parse tree to identify syntactic constructs/analysis and a skeletal logical form graph to identity semantic constructs/analysis, wherein the syntax parse tree and the skeletal logical form graph are represented as independent and separate data structures within the memory.

Comments on Drawing Corrections

4. In the response filed 12/14/98 (Paper No. 11), Applicant agreed to add the legend "Prior Art" to Figures 1-23 and to file such drawing corrections under a separate cover. As of the present Office Action, no drawing corrections have been received. As the application is now allowed by the Examiner, formal correction of the noted defect can no longer be deferred. Applicant is required to submit NEW formal drawings including the aforementioned drawing correction.

5. **Any response to this action should be mailed to:**

Commissioner of Patents and Trademarks
Washington, D.C. 20231

or faxed to:

(703) 305-9051, (for formal communications intended for entry)

Or:

(703) 305-5356 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2021 Crystal Drive, Arlington. VA., Sixth Floor (Receptionist).

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Joseph Thomas, whose telephone number is (703) 305-9588. The examiner can normally be reached on Monday through Thursday from 8:30 AM to 5:00 PM. The examiner can also be reached on alternate Fridays.

If attempts to reach the examiner are unsuccessful, the examiners' supervisor, Forester W. Isen, can be reached at (703) 305-4386.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-3900.

jt
April 19, 1999

Joseph Thomas
Primary Examiner
Art Unit 2747

Notice of References Cited

Application No.	Applicant(s)
08/674,610	HEIDORN, et al.
Examiner	**Group Art Unit**
JOSEPH THOMAS	2747

Page 1 of 1

U.S. PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS
x	A	5,111,398	5/5/92	NUNBERG, et al.	704	9
x	B	5,386,556	1/31/95	HEDIN, et al.	707	4
x	C	5,424,947	6/13/95	NAGAO, et al.	704	9
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUBCLASS
	N						
	O						
	P						
	Q						
	R						
	S						
	T						

NON-PATENT DOCUMENTS

*		DOCUMENT (Including Author, Title, Source, and Pertinent Pages)	DATE
	U		
	V		
	W		
	X		

* A copy of this reference is not being furnished with this Office action.
(See Manual of Patent Examining Procedure, Section 707.05(a).)

NOTICE OF ALLOWANCE AND ISSUE FEE DUE

LM41/0427

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

APPLICATION NO.	FILING DATE	TOTAL CLAIMS	EXAMINER AND GROUP ART UNIT		DATE MAILED
08/674,610	06/28/96	009	THOMAS, J	2747	04/27/99

First Named Applicant: HEIDORN, 35 USC 154(b) term ext. = 0 Days.

TITLE OF INVENTION: METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

ATTY'S DOCKET NO.	CLASS-SUBCLASS	BATCH NO.	APPLN. TYPE	SMALL ENTITY	FEE DUE	DATE DUE
2 661005.447	704-009.000	S42	UTILITY	NO	$1210.00	07/27/99

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. <u>PROSECUTION ON THE MERITS IS CLOSED.</u>

THE ISSUE FEE MUST BE PAID WITHIN <u>THREE MONTHS</u> FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. <u>THIS STATUTORY PERIOD CANNOT BE EXTENDED.</u>

HOW TO RESPOND TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.
If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is changed, pay twice the amount of the FEE DUE shown above and notify the Patent and Trademark Office of the change in status, or
B. If the status is the same, pay the FEE DUE shown above.

If the SMALL ENTITY is shown as NO:

A. Pay FEE DUE shown above, or

B. File verified statement of Small Entity Status before, or with, payment of 1/2 the FEE DUE shown above.

II. Part B-Issue Fee Transmittal should be completed and returned to the Patent and Trademark Office (PTO) with your ISSUE FEE. Even if the ISSUE FEE has already been paid by charge to deposit account, Part B Issue Fee Transmittal should be completed and returned. If you are charging the ISSUE FEE to your deposit account, section "4b" of Part B-Issue Fee Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give application number and batch number.
Please direct all communications prior to issuance to Box ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: ***Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.***

PATENT AND TRADEMARK OFFICE COPY

PTOL-85 (REV. 10-96) Approved for use through 06/30/99. (0651-0033)

#162B

73/0/7530
5794 7/17/99

B

PATENT

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Drawing, Review Branch, Assistant Commissioner for Patents, Washington, D.C. 20231.

July 13, 1999
Date

Steven D. Lawrenz

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

OIPE JUL 19 1999 PATENT & TRADEMARK OFFICE

Examiner	:	J. Thomas
Art Unit	:	2747
Docket No.	:	661005.447
Date	:	July 13, 1999

Drawing Review Branch
Assistant Commissioner for Patents
Washington, DC 20231

REQUEST FOR DRAWING CHANGE

Sir:

Drawing changes, as indicated in red on the attached drawings, are hereby submitted for approval by the Examiner.

RECEIVED
JUL 23 1999
Publishing Division

Respectfully submitted,

George Heidorn et al.

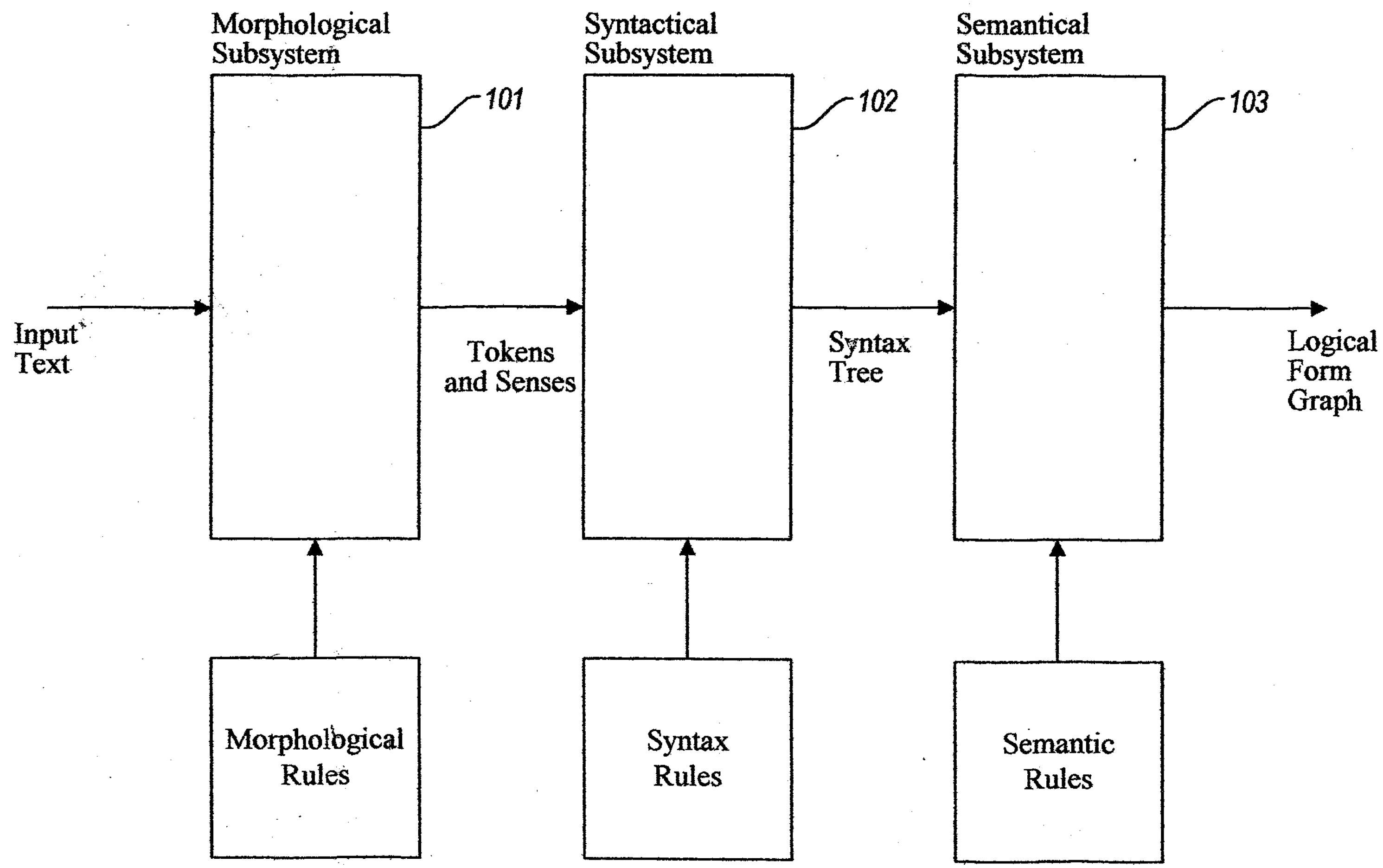
SEED and BERRY LLP

Steven D. Lawrenz
Registration No. 37,376

SDL:brg

Enclosures:
Postcard
Figures 1-23

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031
wpn/MS/661005/447/Forms/Request Drawing Change



(PRIOR ART)
Fig. 1

```
                                                              201
the   203
{
  Adj   204        205
        {Lemma         "the"
         Bits          Sing Plur Wa6 Det Art
                       B0 Def}   206

  Adv   212
        {Lemma         "the"
         Bits          Wa5}

  Senses   207
        {Bits   208    Sing Plur Wa6 Closed
                       Det Art Def
         Lemma         "the"
         Cat           Adj
         Infl          Adj-nil
         Defin         "used when it is clearly understood who or what is meant"
         Exs           "We have a cat and a dog.  The cat (= our cat) is black and the dog
                       (= our dog) white."
                       "the history of China (= Chinese history)"
                       "The Danes that I know work very hard."
                       "Take these letters to the post office (it is understood that you know
                       which post office and where it is)"}   209
210
        {Lemma         "the"
         Cat           Adv
         Defin         "To that extent; by that much"
         Exs           "the sooner the better."}   211

        (more sense records)   213
}
                                                              202
person
{
 Noun
        {Lemma         "person"
         Bits          Pers3 Sing Humn Mass
                       Anim Count Conc C9
                       Humn_sr
         Infl          Noun-default}
 Senses
        {Lemma         "person"
         Cat           Noun
         Defin         "A living human being."
         Exs           "chairperson"
                       "spokesperson"
                       "salesperson."}

          (more sense records)
}
```

(PRIOR ART) Fig. 2

```
whom
{
 Pron
          {Lemma     "who"
           Bits      Pers3 Sing Plur Rel Wh
                     Humn Obj Anim}
 Senses
          {Lemma     "who"
           Bits      Pers3 Sing Plur Rel Wh
                     Closed Humn Obj Anim
           Cat       Pron
           Defin     "(the object form of who, used esp. in writing and careful speech)"
           Exs       "With whom?"
                     "The man with whom he talked."
                     "You saw whom?"
                     "Whom did they see?"
                     "the man (whom) they saw arriving"
                     "a man (whom) you may know of"}
         (more sense records)
 }
```

```
i
{
 Noun
          {Lemma     "i"
           Bits      Pers3 Sing TakesAn
           Infl      Noun-irreg}
 Pron
          {Lemma     "I"
           Bits      Sing Nom TakesAn Persl
                     Humn Anim LexCap}
 Senses
          {Lemma     "i"
           Cat       Noun
           Infl      Noun-irreg
           Defin     "The ninth letter of the modern English alphabet."}

          {Lemma     "I"
           Cat       Pron
           Defin     "Used to refer to oneself as speaker or writer."}
         (more sense records)
}
```

```
met
{
  Verb
          {Lemma     "meet"
           Bits      Sing Plur Past
                     Pastpart
           Infl      Verb-meet}
  Senses
          {Lemma     "meet"
           Bits      Past Pastpart
           Cat       Verb}
}
```

(PRIOR ART) Fig. 3

```
was
{
  Verb
           {Lemma      "be"
            Bits       Pers3 Sing Past Pers1
            Infl       Verb-be } }
  Senses
           {Lemma      "be"
            Bits       Past Pastpart
            Cat        Verb}

           (more sense records)
}
```

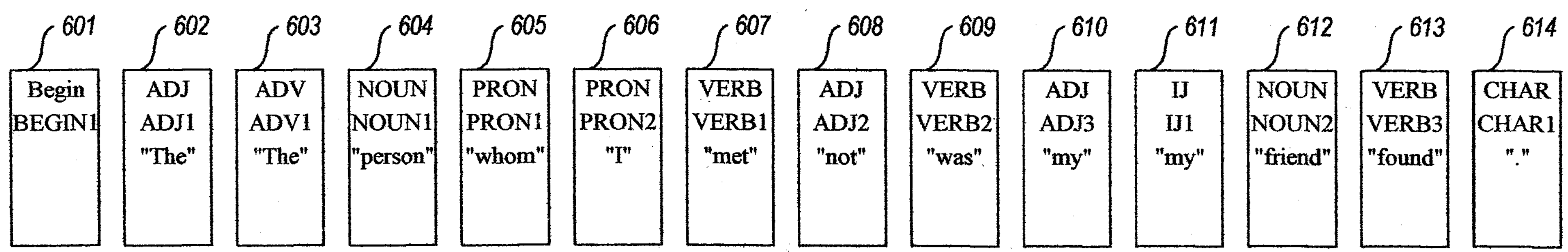
```
my
{
  Adj
           {Lemma      "I"
            Bits       Wa5 Det Poss Pers1 Def
                       Gen A0
            Infl       Adj-none }
  Ij
           {Lemma      "my } }

  Senses
           {Lemma      "I"
            Bits       Wa5 Closed Det Poss
                       Pers1 Def Gen A0
            Cat        Adj
            Infl       Adj-none
            Defin      "belonging to me"
            Exs        "my car"
                       "my mother"}

           {Cat        Ij
            Defin      "Used as an exclamation of surprise, pleasure, or dismay"
            Exs        "Oh, my!  What a tiring day!"}

           (more sense records)
}
```

(PRIOR ART)
Fig. 4

```
friend
{
  Noun
          {Lemma      "friend"
           Bits       Pers3 Sing Humn Anim
                      Count Conc Humn_sr N0
                      Wrdy
           Infl       Noun-default
           Vprp       (of to)
           Bitrecs
                {Bits         Humn Count Conc
                 Vprp         (of)  }

                {Bits         Humn Count Conc
                 Vprp         (to)  }  }

  Verb
          {Lemma      "friend"
           Bits       Inf Plur Pres T1
           Infl       Verb-default  }  }

  Senses
          {Lemma      "friend"
           Bits       Humn Conc
           Cat        Noun
           Defin      "A person whom one knows, likes, and trusts."}

          {Bits       T1
           Lemma      "friend"
           Cat        Verb
           Infl       Verb-default
           Defin      "To befriend."}

          (more sense records)
}
```

(PRIOR ART)

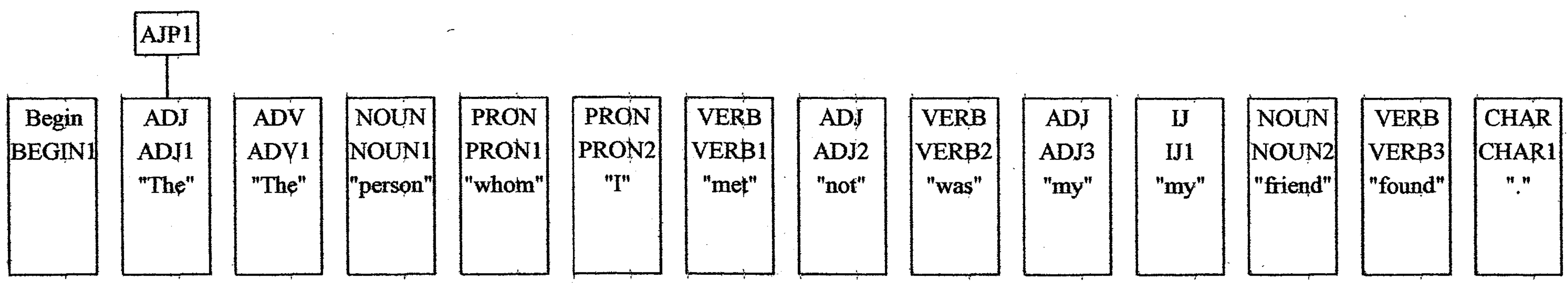
Fig. 5



(PRIOR ART)
Fig. 6

Rule: Adjective to Adjective Phrase

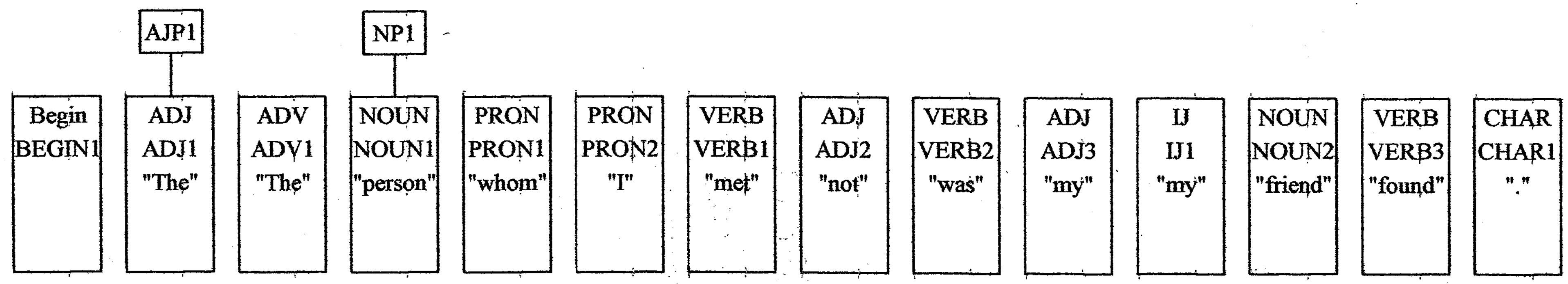
ADJ1 → AJP1



(PRIOR ART)
Fig. 7

Rule: Noun to Noun Phrase

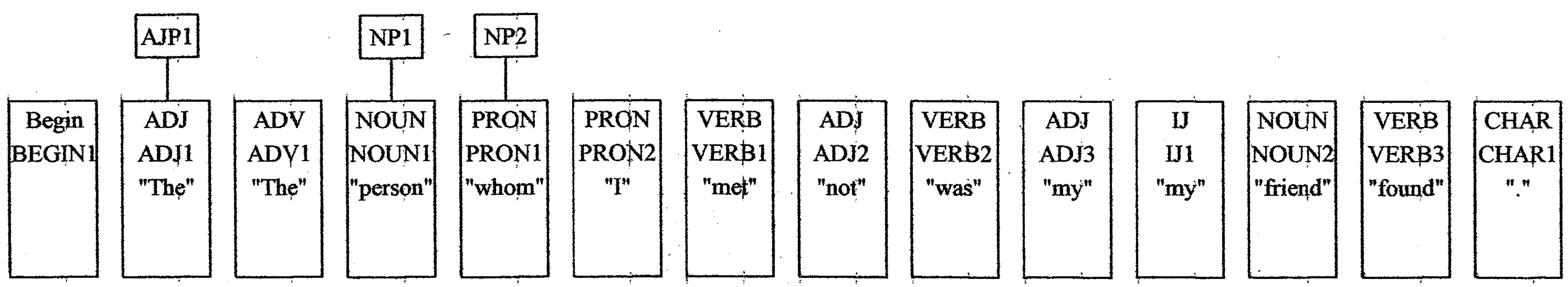
NOUN1 → NP1



(PRIOR ART)
Fig. 8

Rule: Pronoun to Noun Phrase

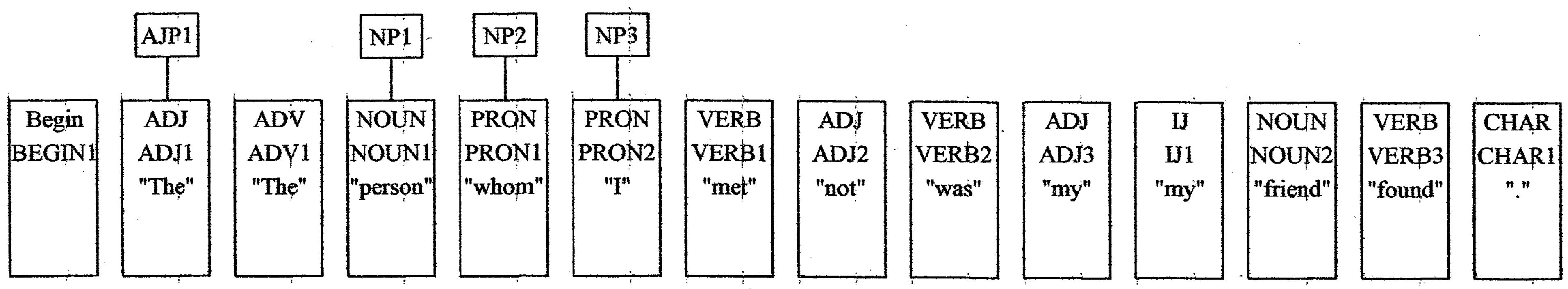
PRON1 → NP2



(PRIOR ART)
Fig. 9

Rule: Pronoun to Noun Phrase

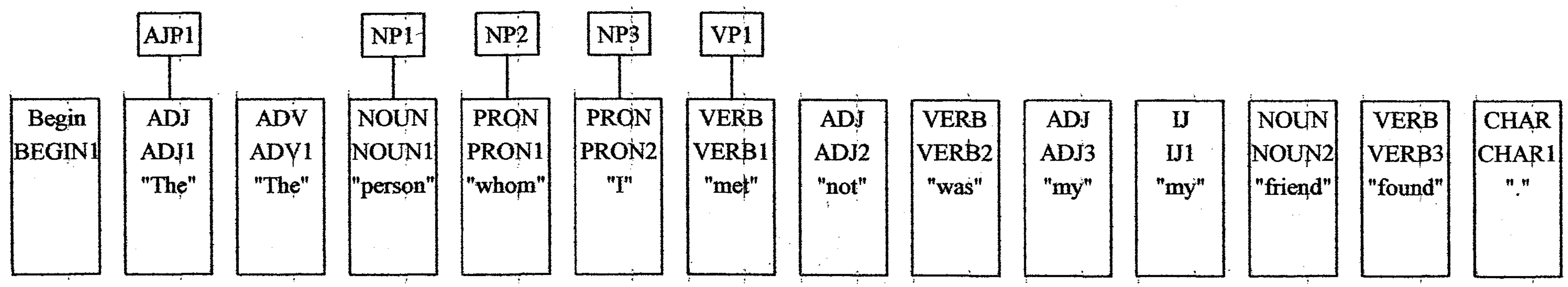
PRON2 → NP3



(PRIOR ART)
Fig. 10

Rule: Verb to Verb Phrase

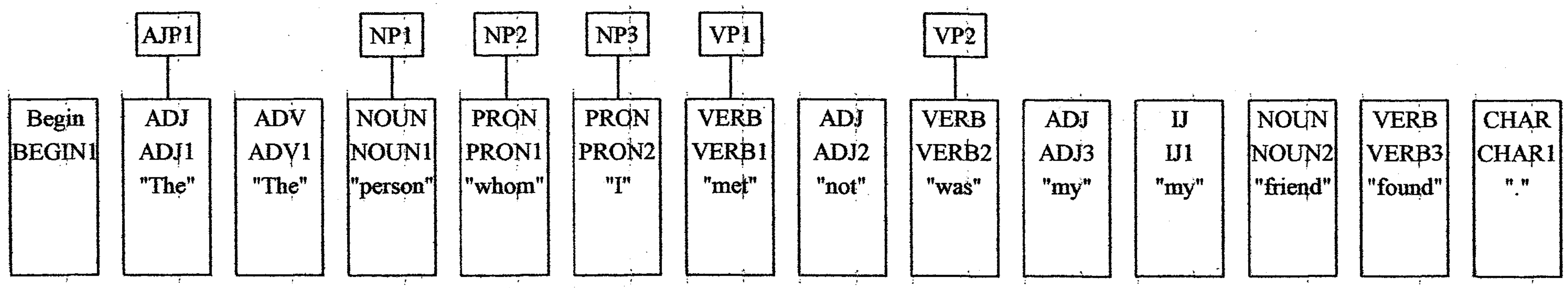
VERB1 → VP1



(PRIOR ART)
Fig. 11

Rule: Verb to Verb Phrase

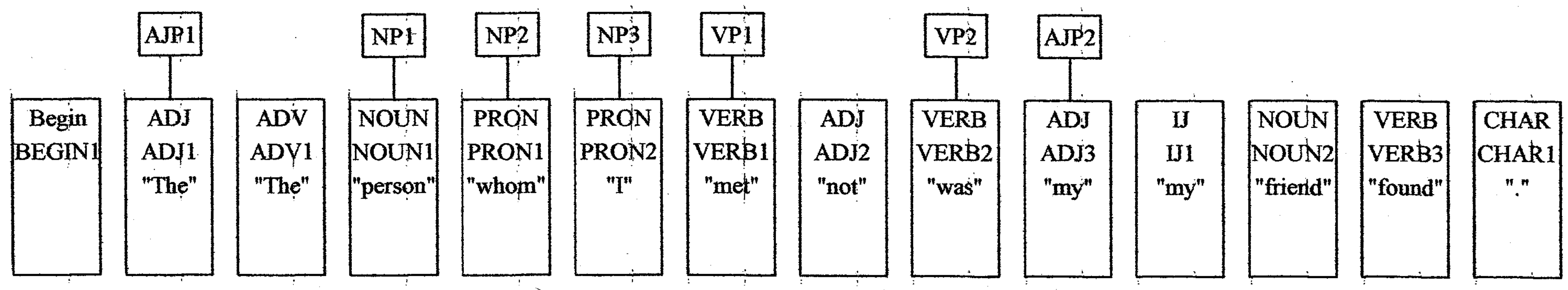
VERB2 → VP2



(PRIOR ART)
Fig. 12

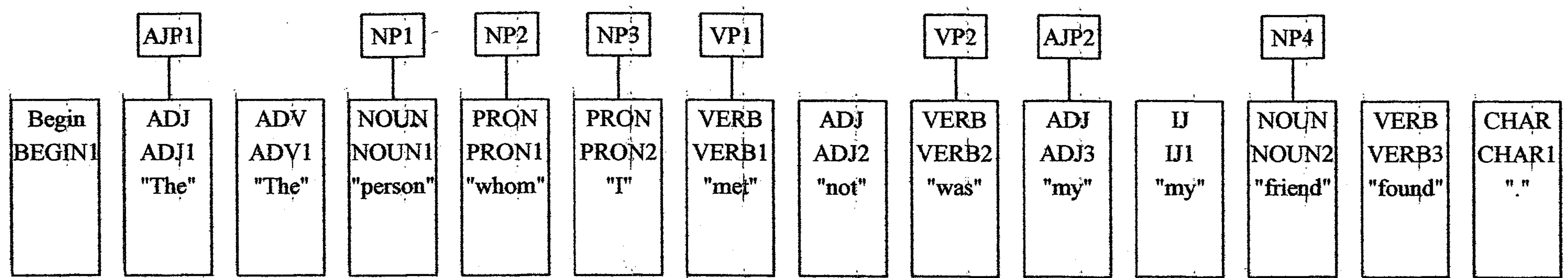
Rule: Adjective to Adjective Phrase

ADJ3 → AJP2



(PRIOR ART)
Fig. 13

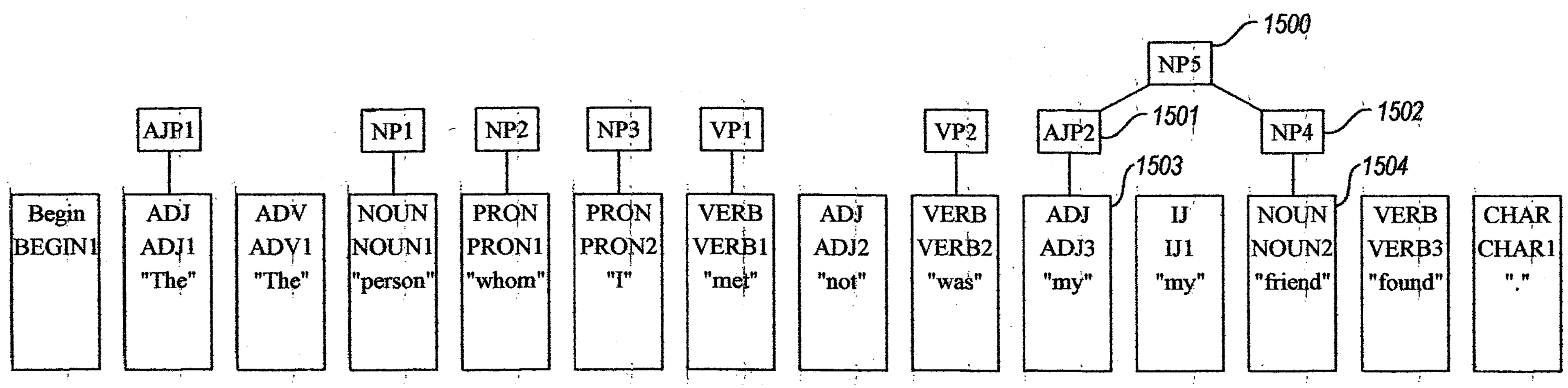
Rule: Noun to Noun Phrase



(PRIOR ART)
Fig. 14

Rule: Noun Phrase with Determer

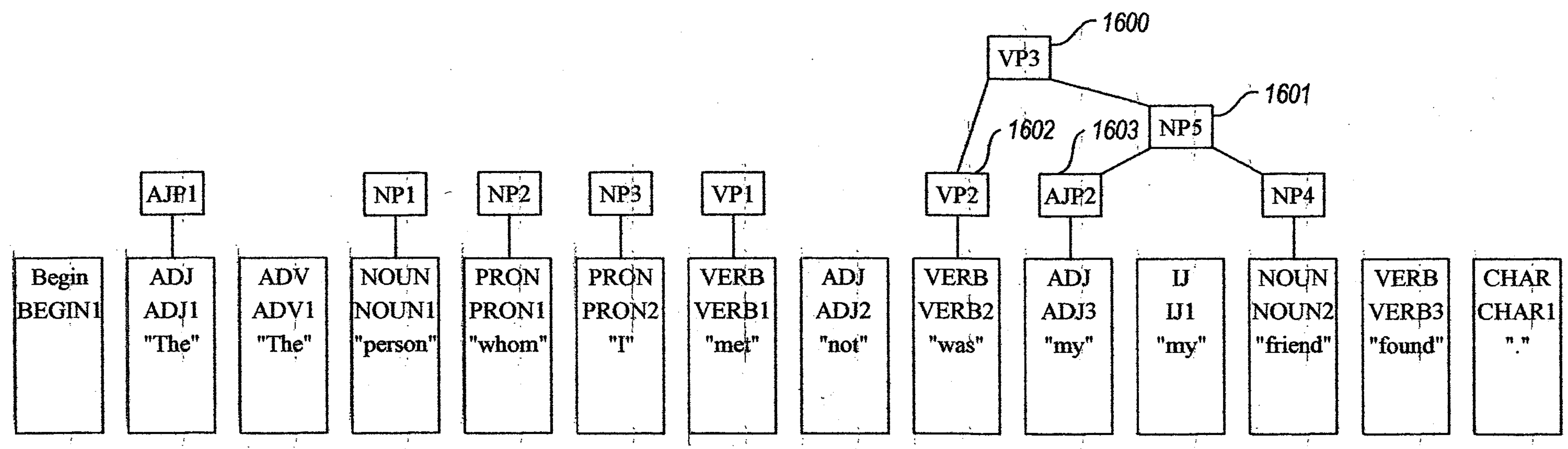
AJP2, NP4 → NP5



(PRIOR ART)
Fig. 15

Rule: Verb Phrase with Noun Phrase as Object of Transitive Verb

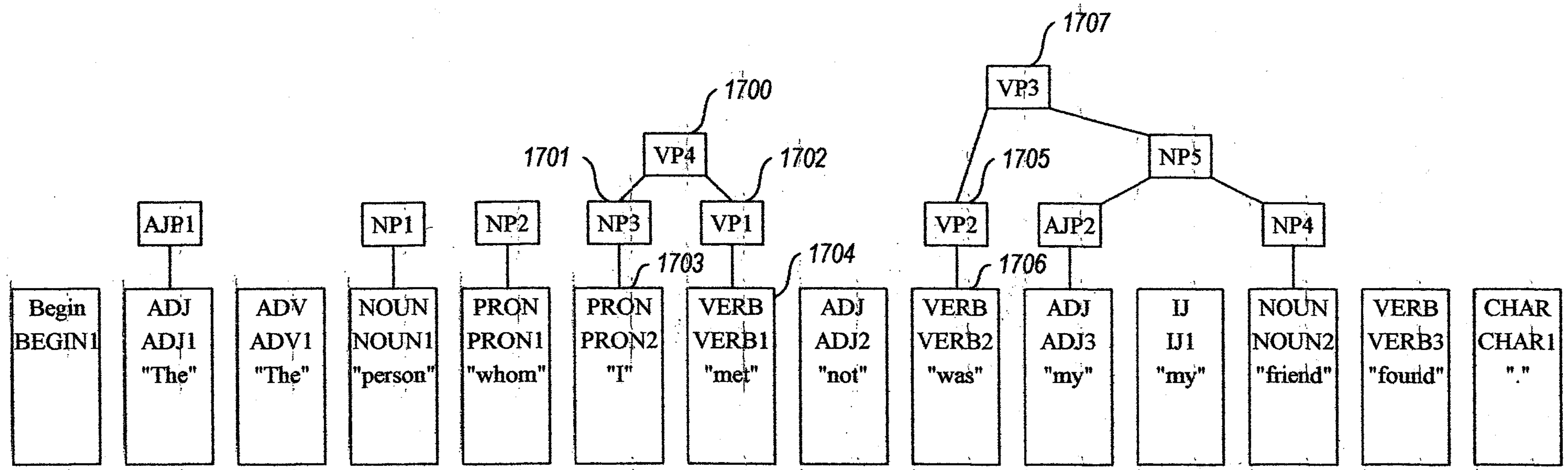
VP2, NP5 $\rightarrow$ VP3



(PRIOR ART)
Fig. 16

Rule: Verb Phrase with Noun Phrase as Subject

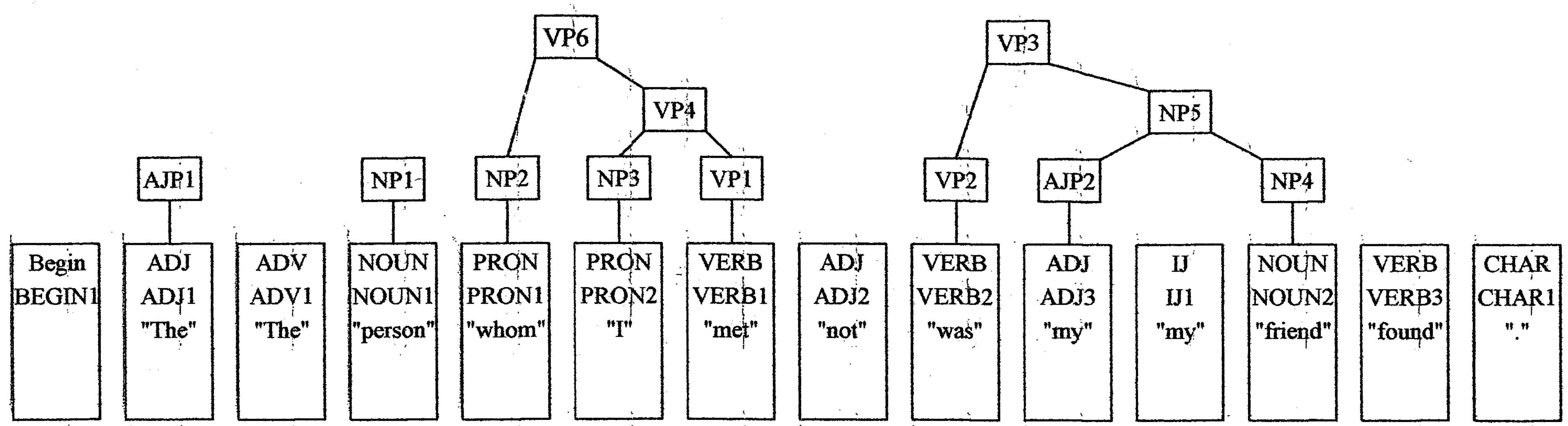
NP3, VP1 → VP4



(PRIOR ART)
Fig. 17

Rule: Topicalization

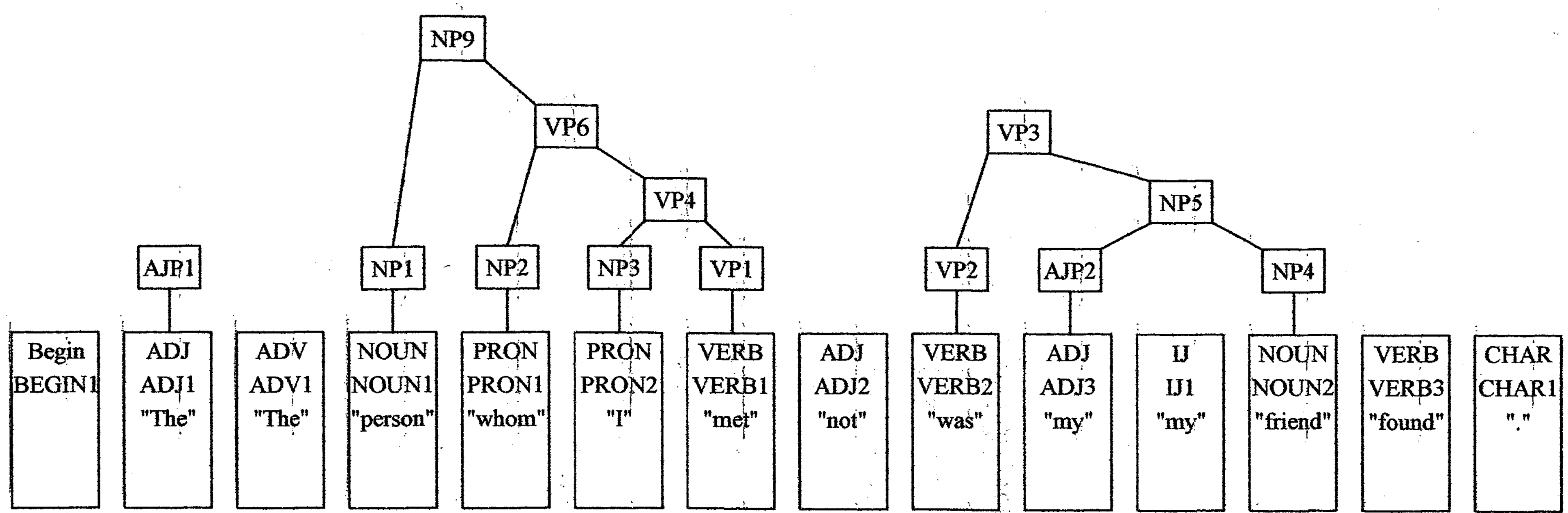
NP2, VP4 → VP6



(PRIOR ART)
Fig. 18

Rule: Noun Phrase with Relative Clause

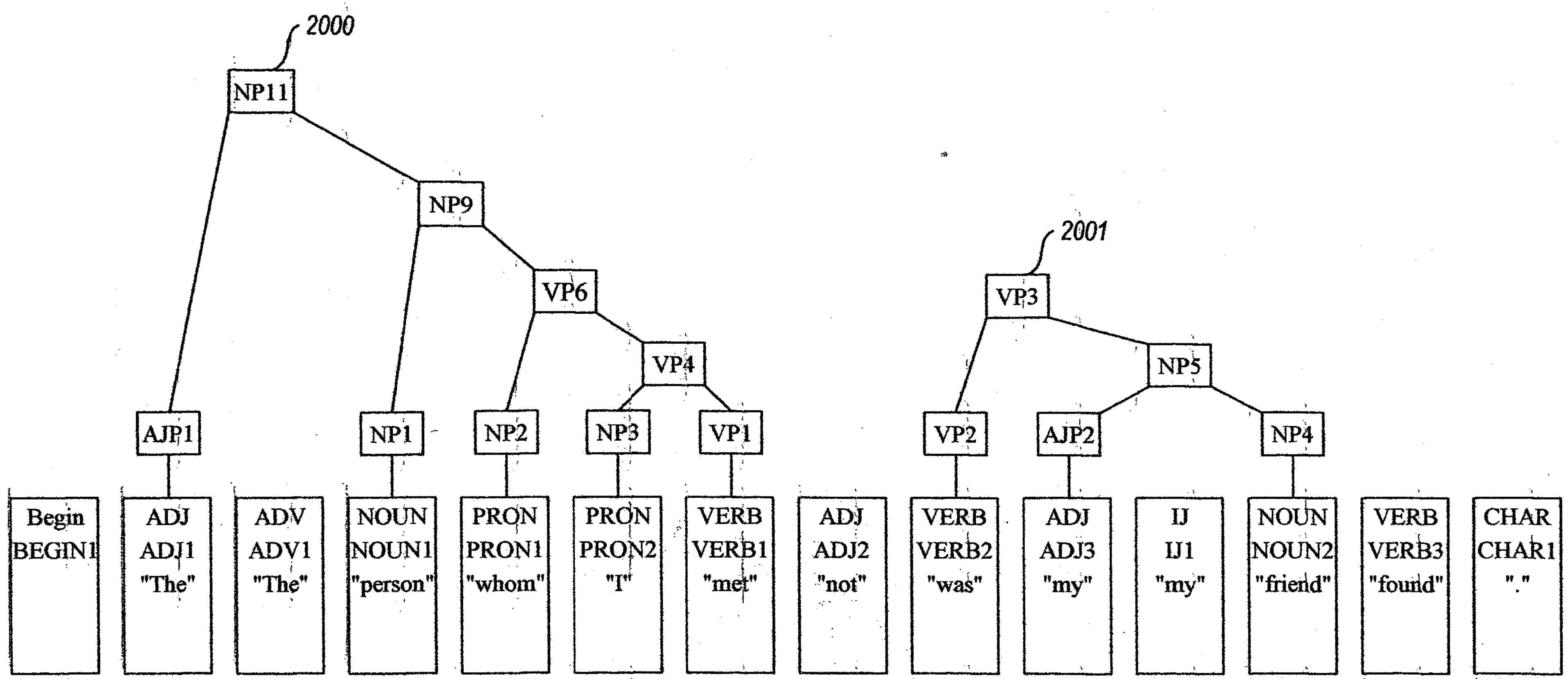
NP1, VP6 → NP9



(PRIOR ART)
Fig. 19

Rule: Noun Phrase with Determinate Quantifier

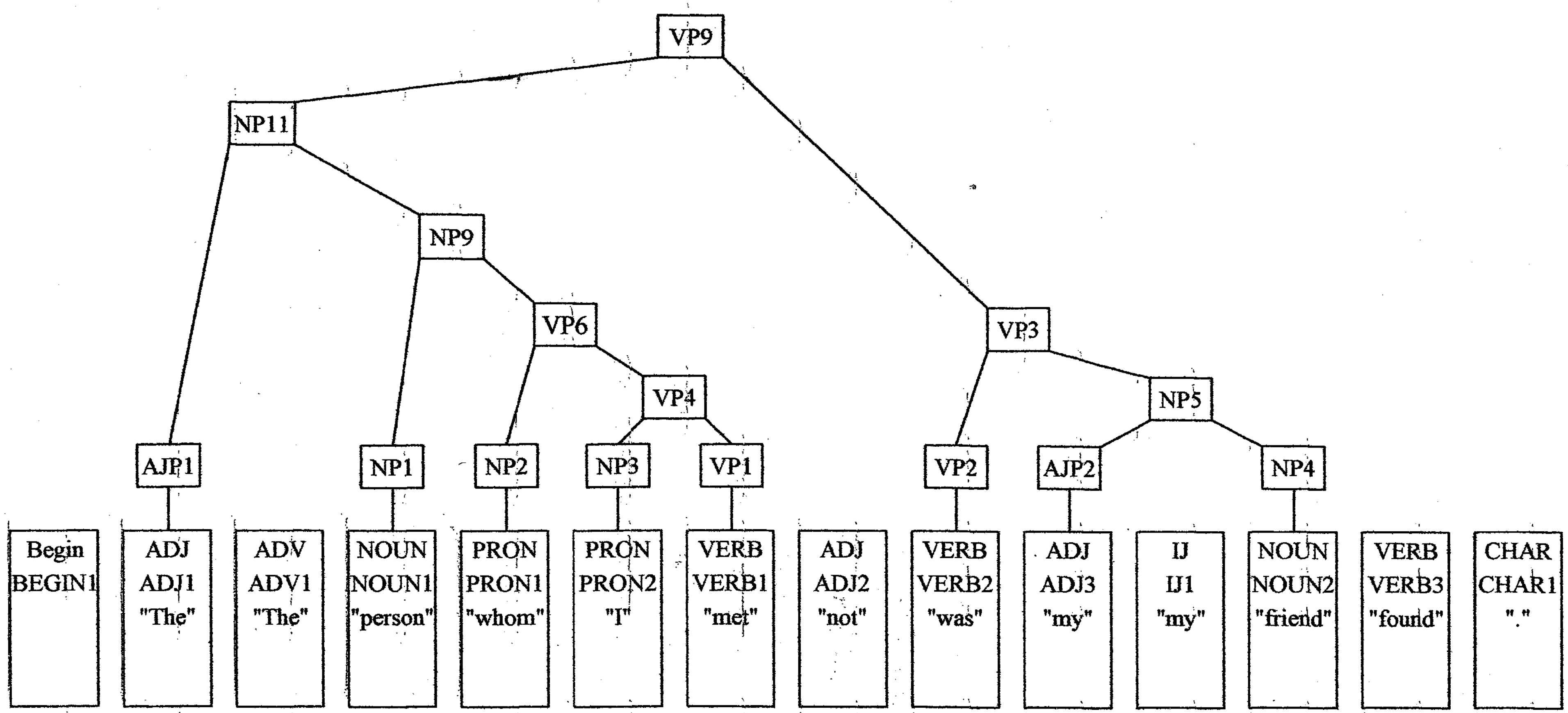
AJP1, NP9 → NP11



(PRIOR ART)
Fig. 20

Rule: Verb Phrase with Noun Phrase Subject

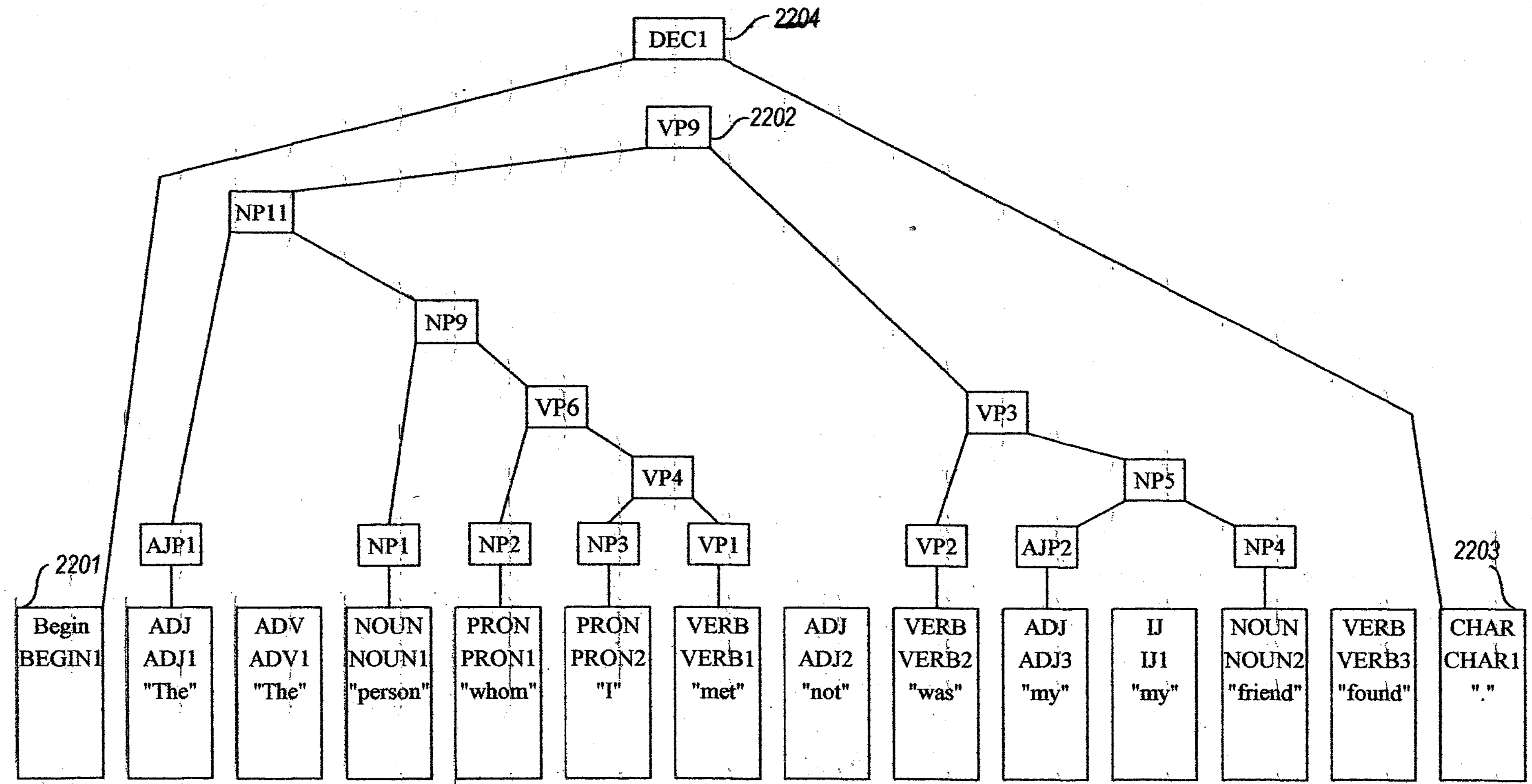
NP11, VP3 → VP9



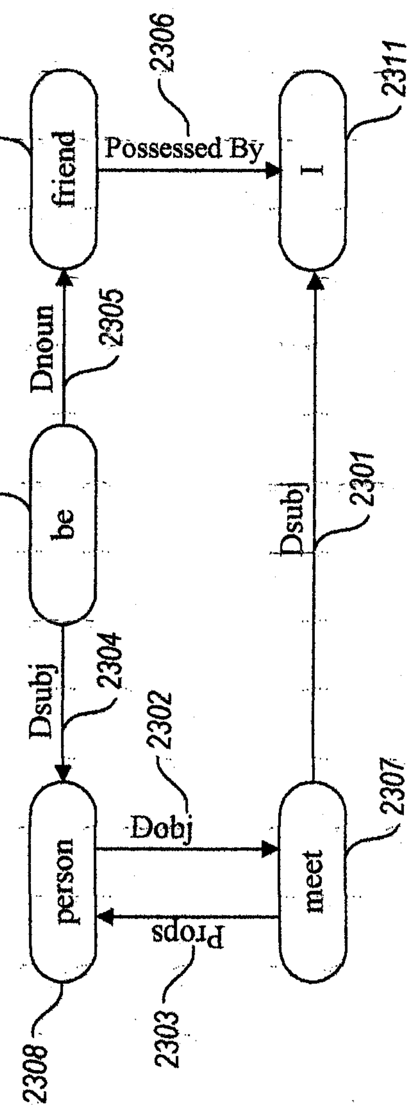
(PRIOR ART)
Fig. 21

Rule: Declarative Sentence from Begin + Verb Phrase + "."

BEGIN1, VP9, CHAR1 → DEC1



(PRIOR ART) Fig. 22



(PRIOR ART)
Fig. 23

#16 B

PATENT

I hereby certify that on the date specified below, this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Drawing Review Branch, Assistant Commissioner for Patents, Washington, DC 20231.

July 13, 1999
Date

Steven D. Lawrenz

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants	:	George Heidorn and Karen Jensen
Application No.	:	08/674,610
Filed	:	June 28, 1996
For	:	METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

OIPE JC78 JUL 19 1999 PATENT & TRADEMARK OFFICE

Batch No.	:	S42
Art Unit No.	:	2747
Date of Notice of Allowance	:	April 27, 1999
Docket No.	:	661005.447
Date	:	July 13, 1999

Drawing Review Branch
Assistant Commissioner for Patents
Washington, DC 20231

FILING FORMAL DRAWINGS AFTER ALLOWANCE

Sir:

Enclosed are 69 sheets of formal drawings, Figures 1-59, for filing in the above-identified application. Enclosed Figures 1-23 have been changed in accordance with the Request for Drawing Change filed herewith.

Respectfully submitted,
George Heidorn et al.
SEED and BERRY LLP

Steven D. Lawrenz
Registration No. 37,376

SDL:brg
Enclosures:
- Postcard
- Copy of Notice of Allowance
- Formal Drawings (Figures 1-59)

6300 Columbia Center
701 Fifth Avenue
Seattle, Washington 98104-7092
(206) 622-4900
Fax: (206) 682-6031

wpn/MS/661005/447/Forms/Filing Drawings After Allow

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office

NOTICE OF ALLOWANCE AND ISSUE FEE DUE

RECEIVED
MAY 0 3 1999
SEED & BERRY LLP

OIPE JC78
JUL 19 1999
PATENT & TRADEMARK OFFICE

LM41/0427

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

APPLICATION NO.	FILING DATE	TOTAL CLAIMS	EXAMINER AND GROUP ART UNIT		DATE MAILED
08/674,610	06/28/96	009	THOMAS, J	2747	04/27/99

First Named Applicant: HEIDORN, 35 USC 154(b) term ext. = 0 Days.

TITLE OF INVENTION: METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

ATTY'S DOCKET NO.	CLASS-SUBCLASS	BATCH NO.	APPLN. TYPE	SMALL ENTITY	FEE DUE	DATE DUE
2 661005.447	704-009.000	S42	UTILITY	NO	$1210.00	07/27/99

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED.

THE ISSUE FEE MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED.

HOW TO RESPOND TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is changed, pay twice the amount of the FEE DUE shown above and notify the Patent and Trademark Office of the change in status, or

B. If the status is the same, pay the FEE DUE shown above.

If the SMALL ENTITY is shown as NO:

A. Pay FEE DUE shown above, or

B. File verified statement of Small Entity Status before, or with, payment of 1/2 the FEE DUE shown above.

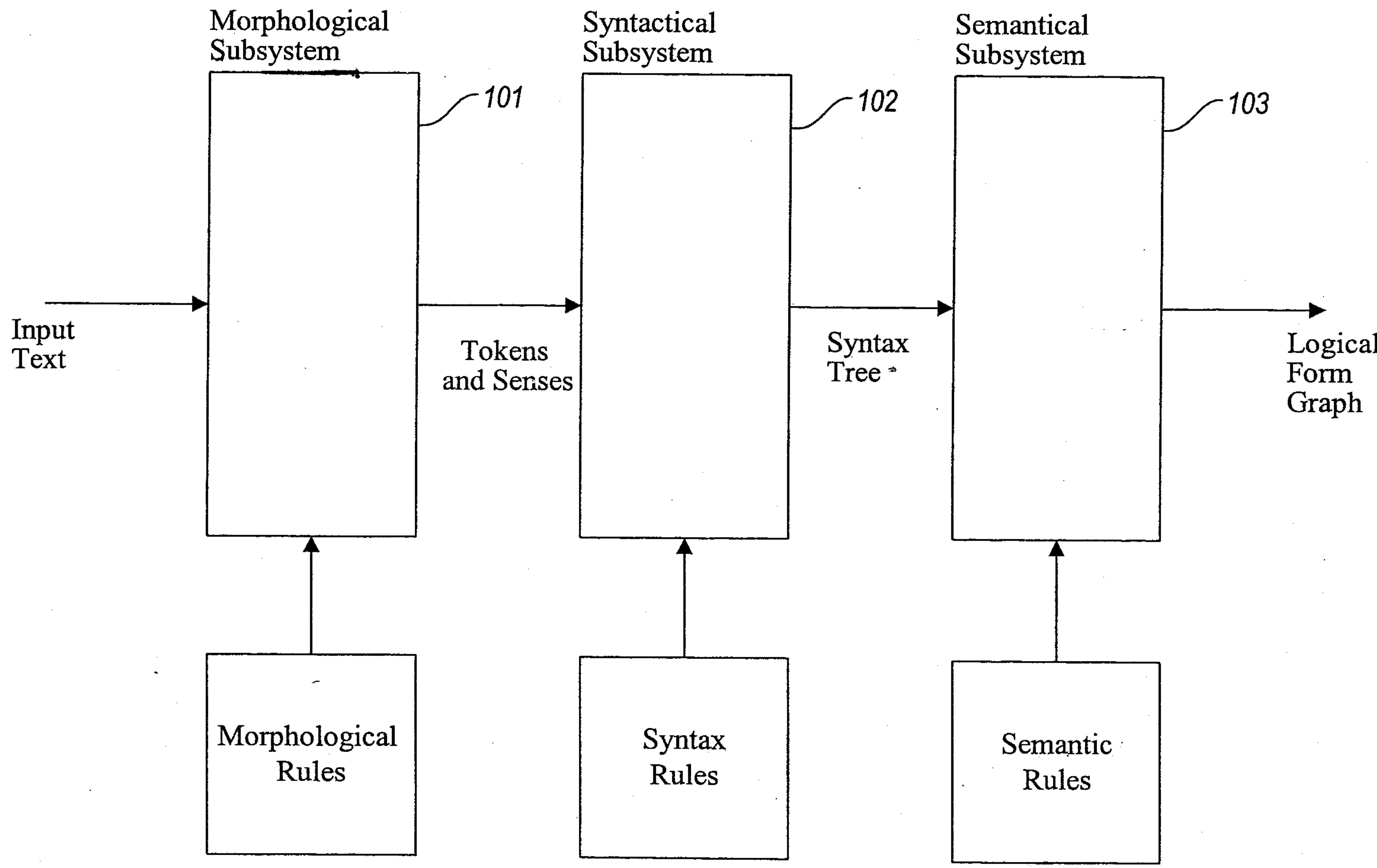
II. Part B-Issue Fee Transmittal should be completed and returned to the Patent and Trademark Office (PTO) with your ISSUE FEE. Even if the ISSUE FEE has already been paid by charge to deposit account, Part B Issue Fee Transmittal should be completed and returned. If you are charging the ISSUE FEE to your deposit account, section "4b" of Part B-Issue Fee Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give application number and batch number. Please direct all communications prior to issuance to Box ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

YOUR COPY

PTOL-85 (REV. 10-96) Approved for use through 06/30/99. (0651-0033)



(PRIOR ART)
Fig. 1

5966686

```
                                                            201
the  203
{
 Adj  204              205
      {Lemma           "the"
       Bits            Sing Plur Wa6 Det Art
                       B0 Def}  206
 Adv  212
      {Lemma           "the"
       Bits            Wa5}

 Senses  207
      {Bits  208       Sing Plur Wa6 Closed
                       Det Art Def
       Lemma           "the"
       Cat             Adj
       Infl            Adj-nil
       Defin           "used when it is clearly understood who or what is meant"
       Exs             "We have a cat and a dog. The cat (= our cat) is black and the dog
                       (= our dog) white."
                       "the history of China (= Chinese history)"
                       "The Danes that I know work very hard."
                       "Take these letters to the post office (it is understood that you know
                       which post office and where it is)"}  209
210
      {Lemma           "the"
       Cat             Adv
       Defin           "To that extent; by that much"
       Exs             "the sooner the better."}  211

     (more sense records)  213
}
                                                            202
person
{
 Noun
      {Lemma           "person"
       Bits            Pers3 Sing Humn Mass
                       Anim Count Conc C9
                       Humn_sr
       Infl            Noun-default}
 Senses
      {Lemma           "person"
       Cat             Noun
       Defin           "A living human being."
       Exs             "chairperson"
                       "spokesperson"
                       "salesperson."}

     (more sense records)
}
```

(PRIOR ART) Fig. 2

```
whom
{
 Pron
          {Lemma      "who"
           Bits       Pers3 Sing Plur Rel Wh
                      Humn Obj Anim}
 Senses
          {Lemma      "who"
           Bits       Pers3 Sing Plur Rel Wh
                      Closed Humn Obj Anim
           Cat        Pron
           Defin      "(the object form of who, used esp. in writing and careful speech)"
           Exs        "With whom?"
                      "The man with whom he talked."
                      "You saw whom?"
                      "Whom did they see?"
                      "the man (whom) they saw arriving"
                      "a man (whom) you may know of"}
          (more sense records)
 }
```

```
i
{
 Noun
          {Lemma      "i"
           Bits       Pers3 Sing TakesAn
           Infl       Noun-irreg}
 Pron
          {Lemma      "I"
           Bits       Sing Nom TakesAn Persl
                      Humn Anim LexCap}
 Senses
          {Lemma      "i"
           Cat        Noun
           Infl       Noun-irreg
           Defin      "The ninth letter of the modern English alphabet."}

          {Lemma      "I"
           Cat        Pron
           Defin      "Used to refer to oneself as speaker or writer."}
          (more sense records)
}
```

```
met
{
  Verb
          {Lemma      "meet"
           Bits       Sing Plur Past
                      Pastpart
           Infl       Verb-meet}
  Senses
          {Lemma      "meet"
           Bits       Past Pastpart
           Cat        Verb}
}
```

(PRIOR ART) Fig. 3

```
was
{
  Verb
          {Lemma     "be"
           Bits      Pers3 Sing Past Pers1
           Infl      Verb-be } }

  Senses
          {Lemma     "be"
           Bits      Past Pastpart
           Cat       Verb}

          (more sense records)
}
```

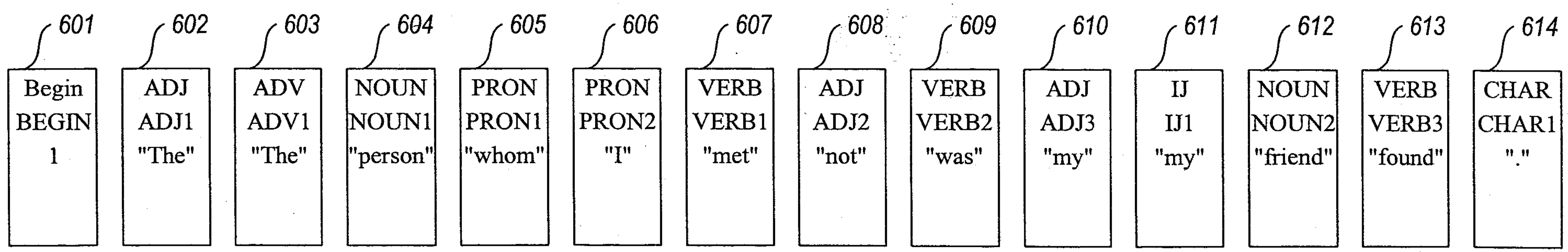
```
my
{
  Adj
          {Lemma     "I"
           Bits      Wa5 Det Poss Pers1 Def
                     Gen A0
           Infl      Adj-none }
  Ij
          {Lemma     "my } }

  Senses
          {Lemma     "I"
           Bits      Wa5 Closed Det Poss
                     Pers1 Def Gen A0
           Cat       Adj
           Infl      Adj-none
           Defin     "belonging to me"
           Exs       "my car"
                     "my mother"}

          {Cat       Ij
           Defin     "Used as an exclamation of surprise, pleasure, or dismay"
           Exs       "Oh, my!  What a tiring day!"}

          (more sense records)
}
```

(PRIOR ART)
Fig. 4

```
friend
{
  Noun
          {Lemma      "friend"
           Bits       Pers3 Sing Humn Anim
                      Count Conc Humn_sr N0
                      Wrdy
           Infl       Noun-default
           Vprp       (of to)
           Bitrecs
                {Bits          Humn Count Conc
                 Vprp          (of)   }

                {Bits          Humn Count Conc
                 Vprp          (to)   }   }

  Verb
          {Lemma      "friend"
           Bits       Inf Plur Pres T1
           Infl       Verb-default  }  }

  Senses
          {Lemma      "friend"
           Bits       Humn Conc
           Cat        Noun
           Defin      "A person whom one knows, likes, and trusts."}

          {Bits       T1
           Lemma      "friend"
           Cat        Verb
           Infl       Verb-default
           Defin      "To befriend."}

        (more sense records)
}
```

(PRIOR ART)

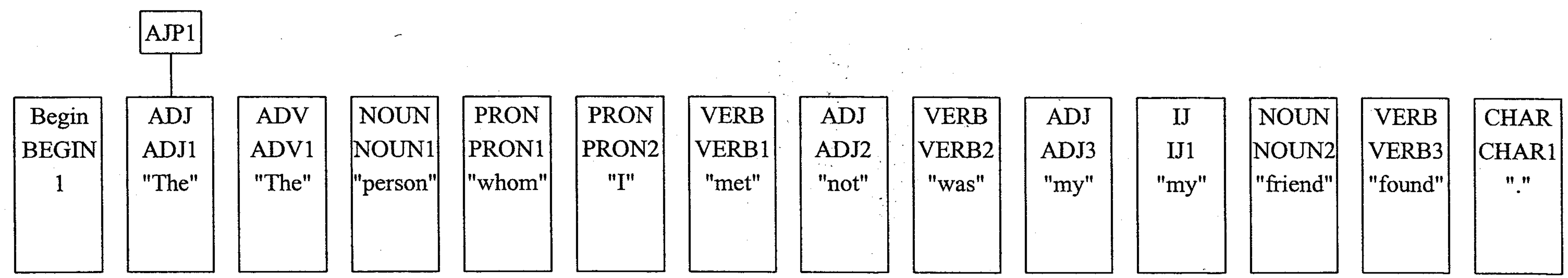
Fig. 5



(PRIOR ART)
Fig. 6

Rule: Adjective to Adjective Phrase

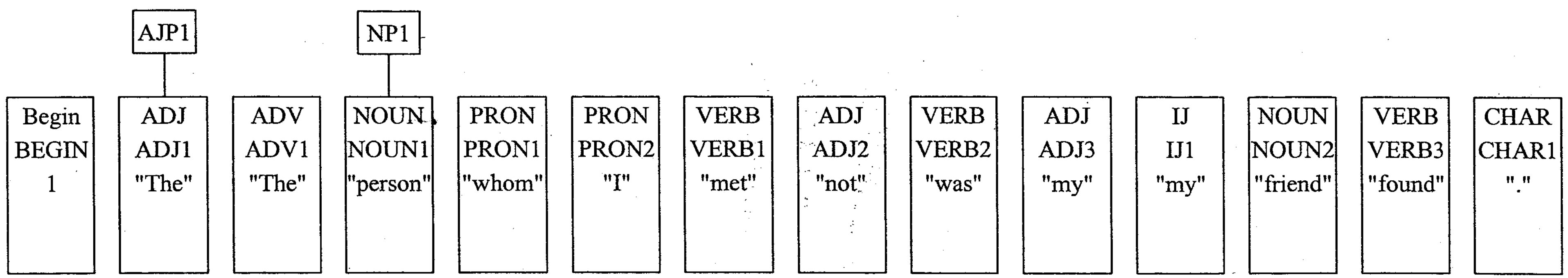
ADJ1 → AJP1



(PRIOR ART)
Fig. 7

Rule: Noun to Noun Phrase

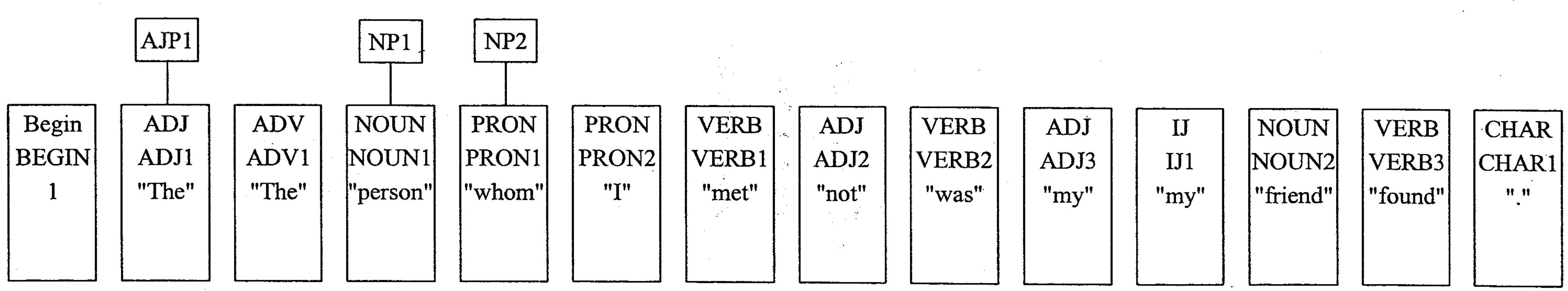
NOUN1 → NP1



(PRIOR ART)
Fig. 8

Rule: Pronoun to Noun Phrase

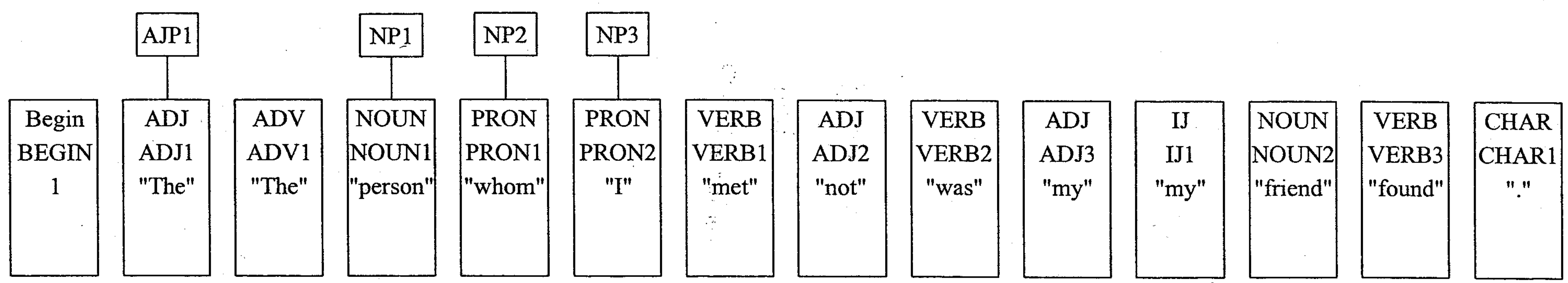
PRON1 → NP2



(PRIOR ART)
Fig. 9

Rule: Pronoun to Noun Phrase

PRON2 → NP3

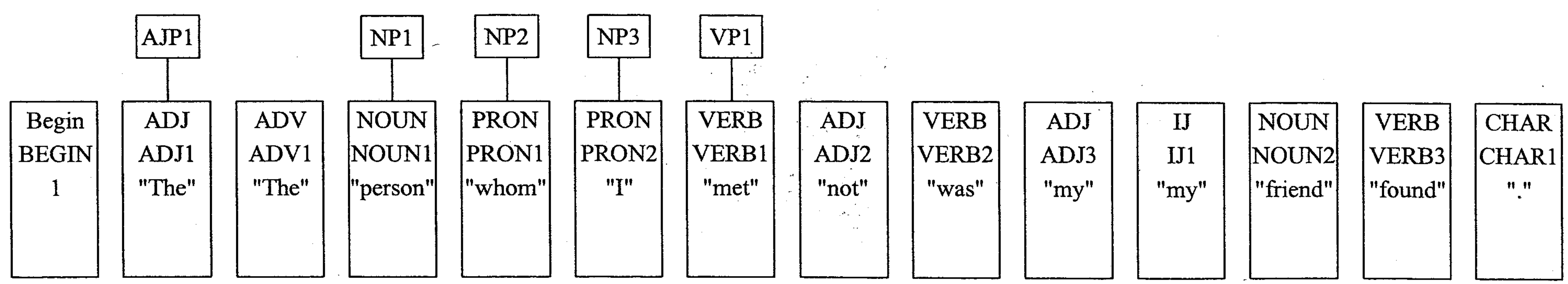


(PRIOR ART)

Fig. 10

Rule: Verb to Verb Phrase

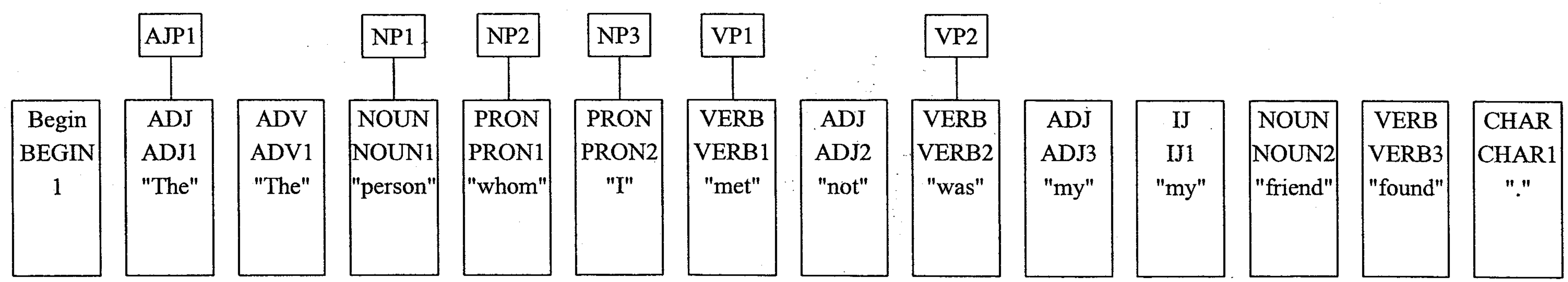
VERB1 → VP1



(PRIOR ART)
Fig. 11

Rule: Verb to Verb Phrase

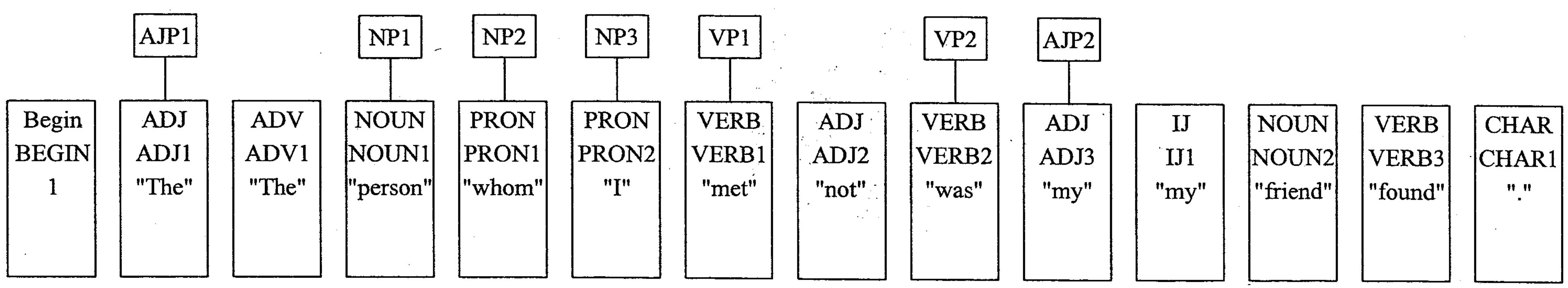
VERB2 → VP2



(PRIOR ART)
Fig. 12

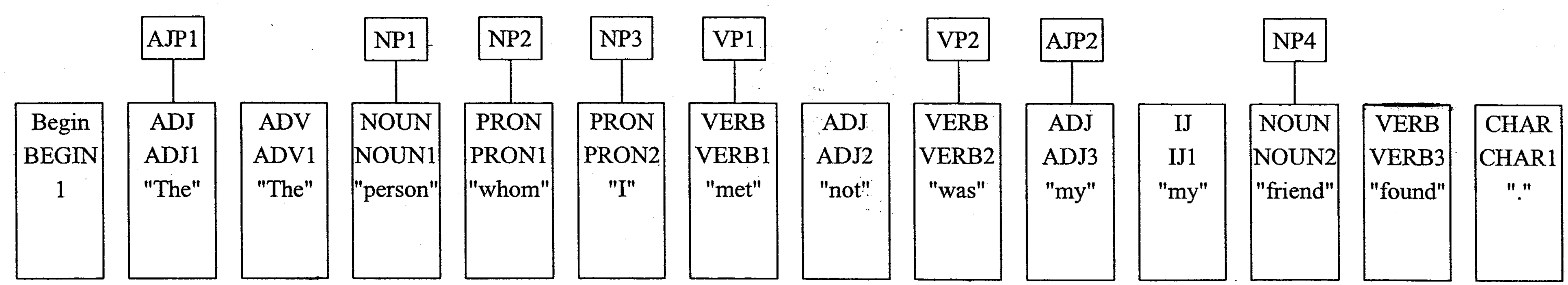
Rule: Adjective to Adjective Phrase

ADJ3 $\rightarrow$ AJP2



(PRIOR ART)
Fig. 13

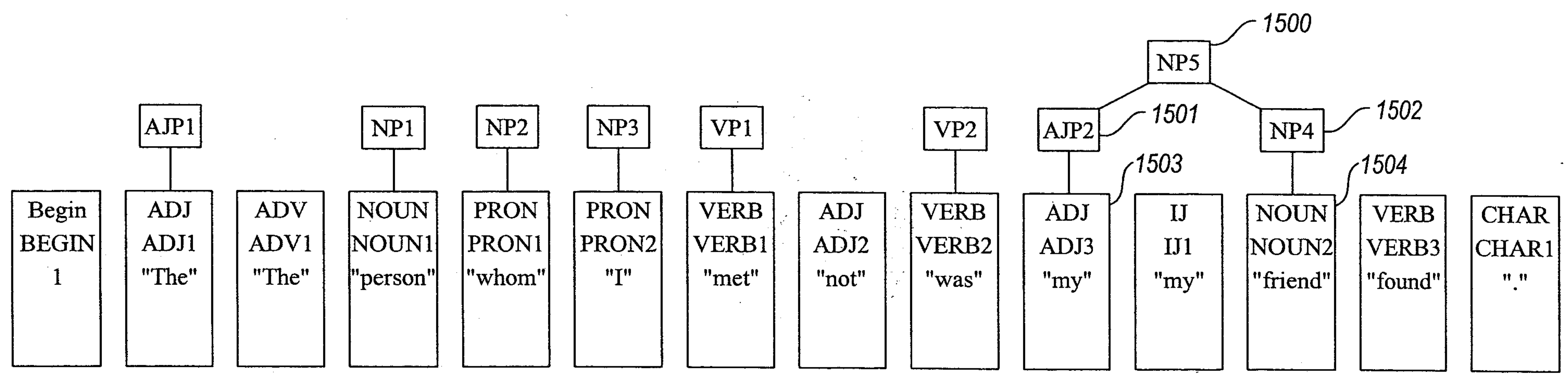
Rule: Noun to Noun Phrase



(PRIOR ART)
Fig. 14

Rule: Noun Phrase with Determer

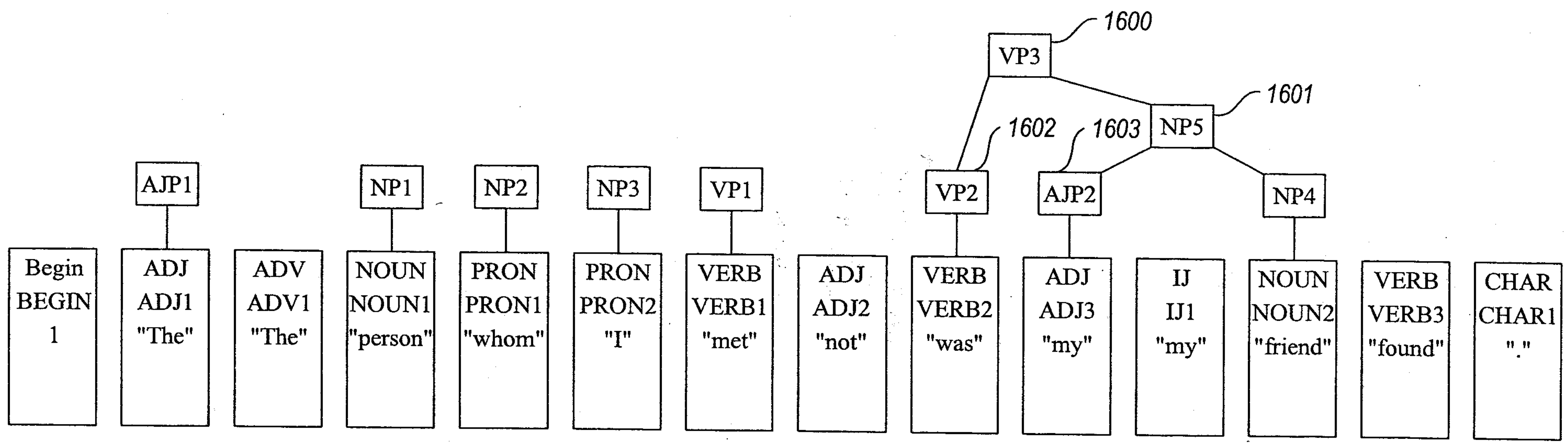
AJP2, NP4 → NP5



(PRIOR ART)
Fig. 15

Rule: Verb Phrase with Noun Phrase as Object of Transitive Verb

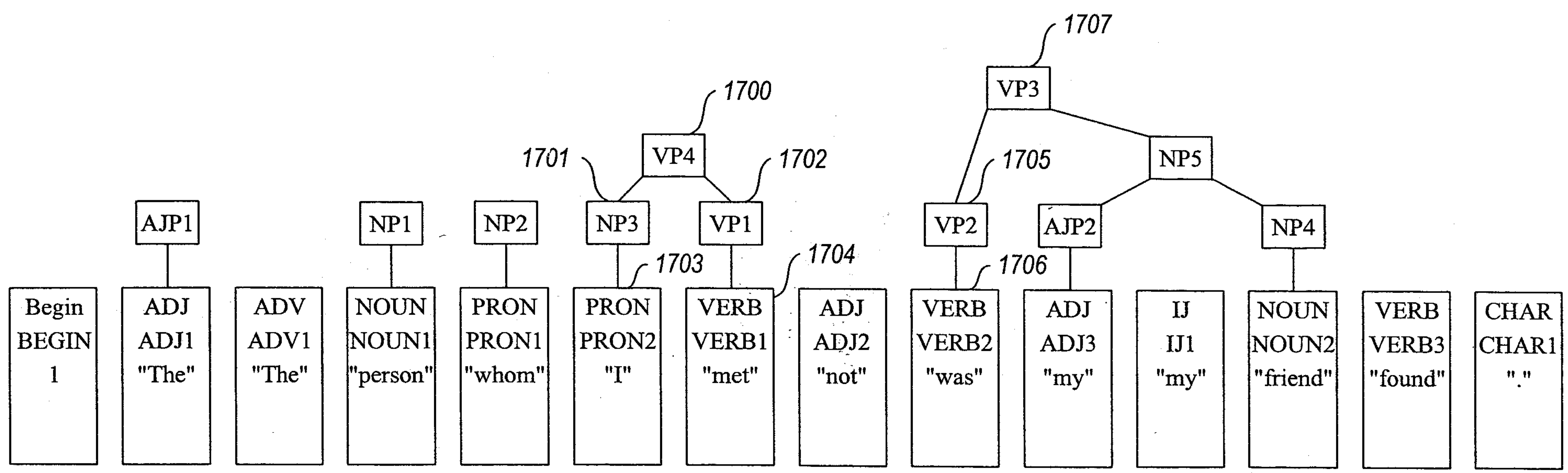
VP2, NP5 $\rightarrow$ VP3



(PRIOR ART)
Fig. 16

Rule: Verb Phrase with Noun Phrase as Subject

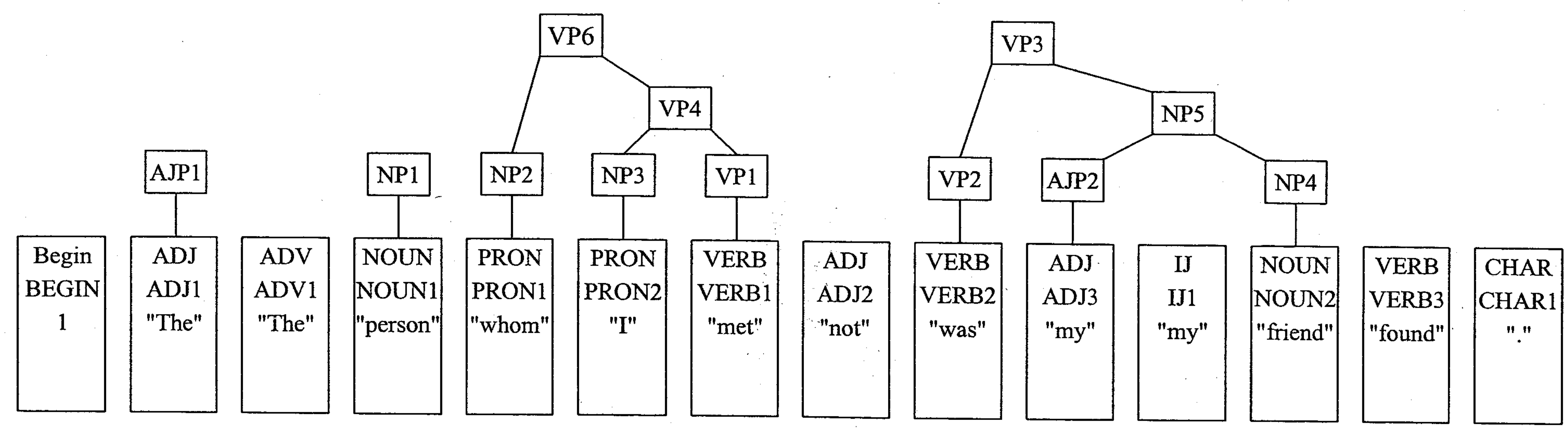
NP3, VP1 → VP4



(PRIOR ART)
Fig. 17

Rule: Topicalization

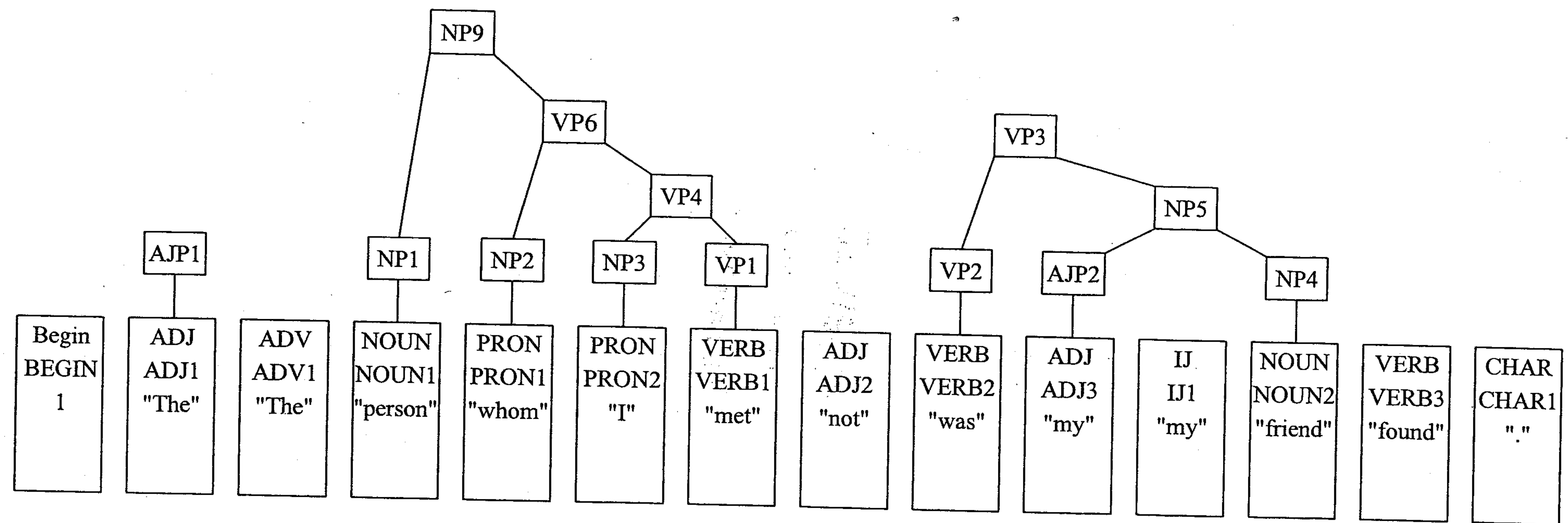
NP2, VP4 → VP6



(PRIOR ART)
Fig. 18

Rule: Noun Phrase with Relative Clause

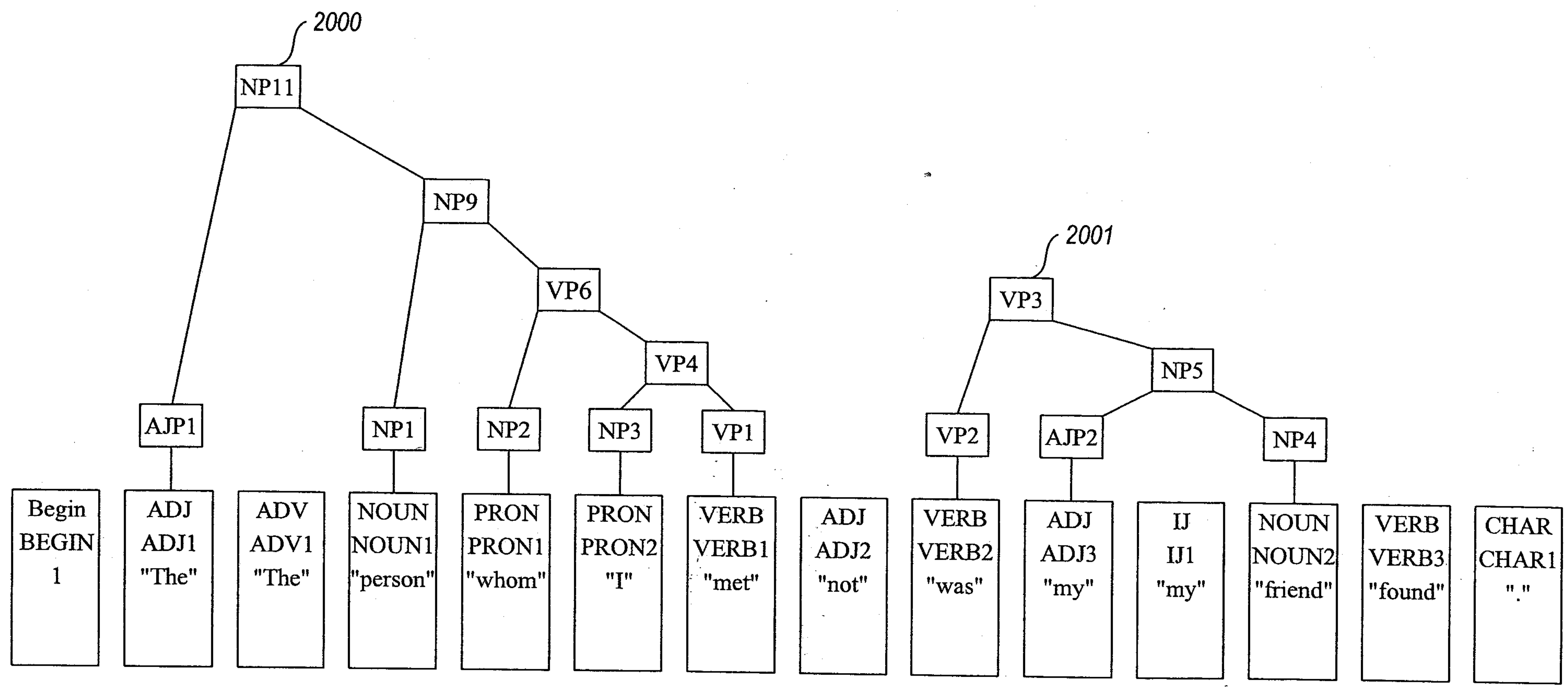
NP1, VP6 $\rightarrow$ NP9



(PRIOR ART)
Fig. 19

Rule: Noun Phrase with Determinate Quantifier

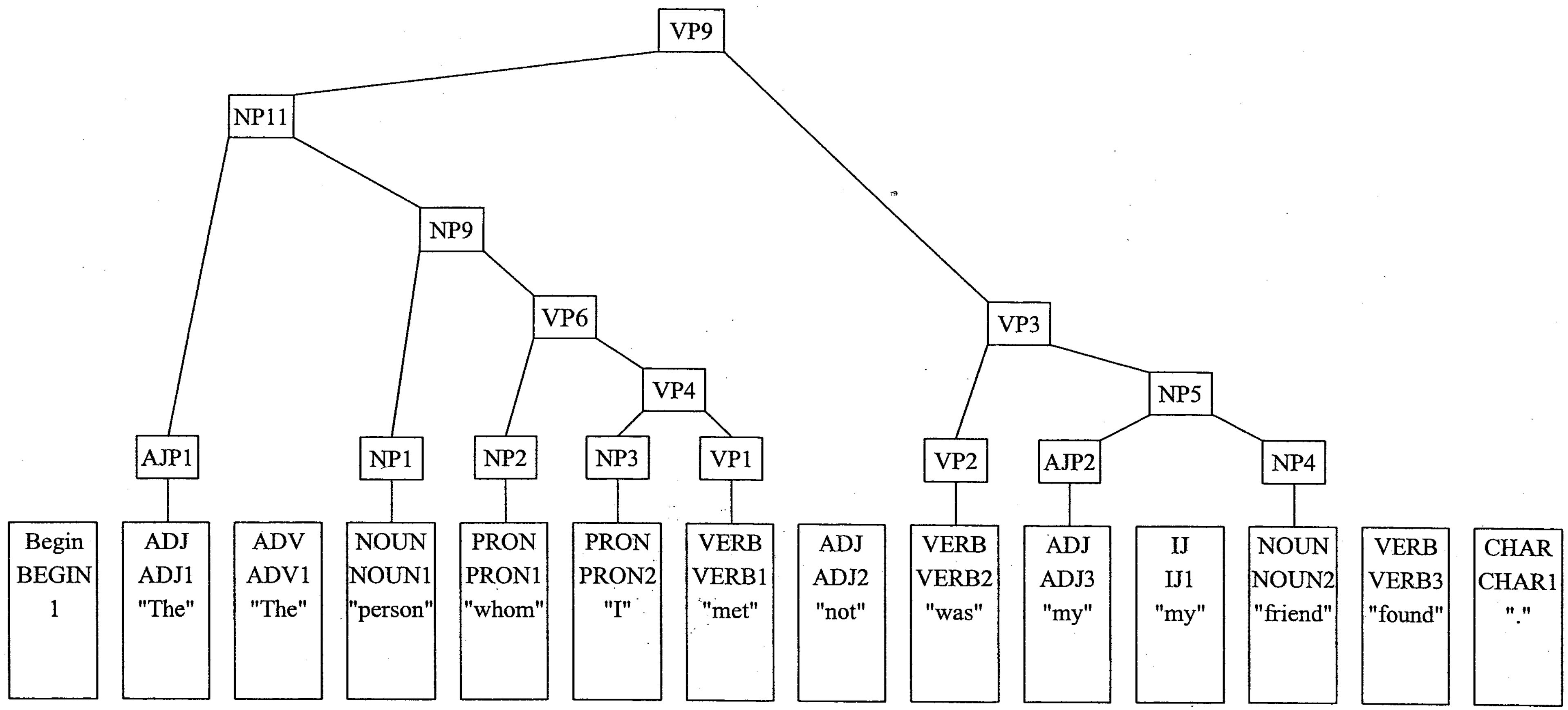
AJP1, NP9 → NP11



(PRIOR ART)
Fig. 20

Rule: Verb Phrase with Noun Phrase Subject

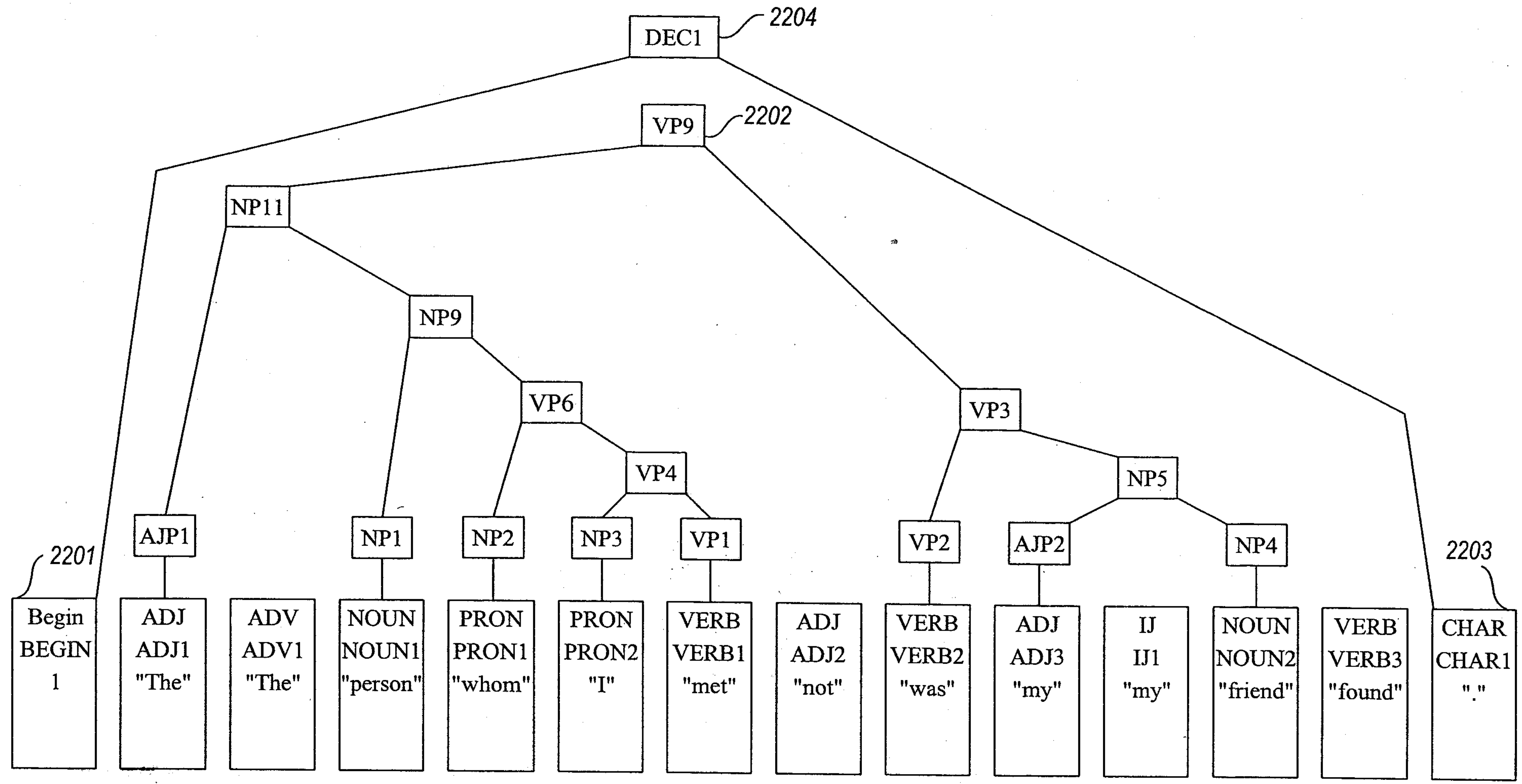
NP11, VP3 $\rightarrow$ VP9



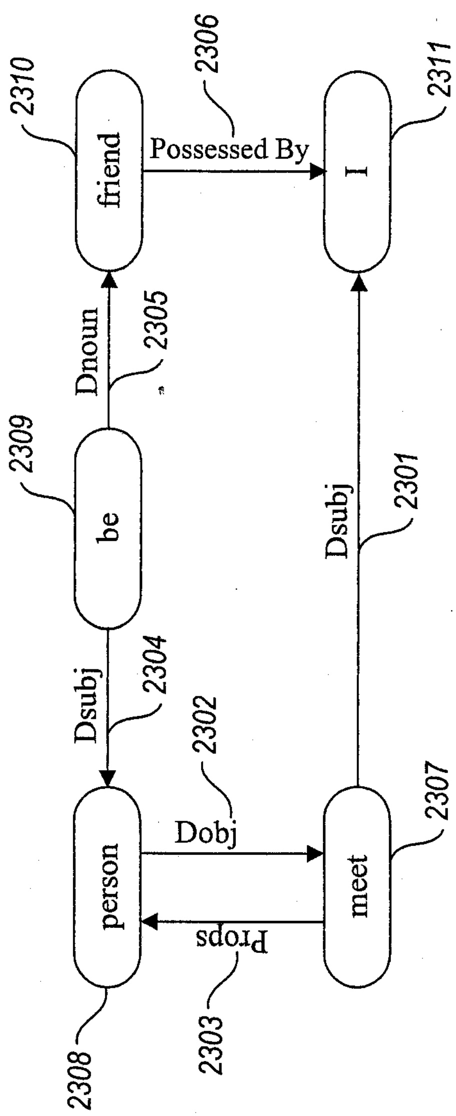
(PRIOR ART)
Fig. 21

Rule: Declarative Sentence from Begin + Verb Phrase + "."

BEGIN1, VP9, CHAR1 → DEC1



(PRIOR ART) Fig. 22



(PRIOR ART)
Fig. 23

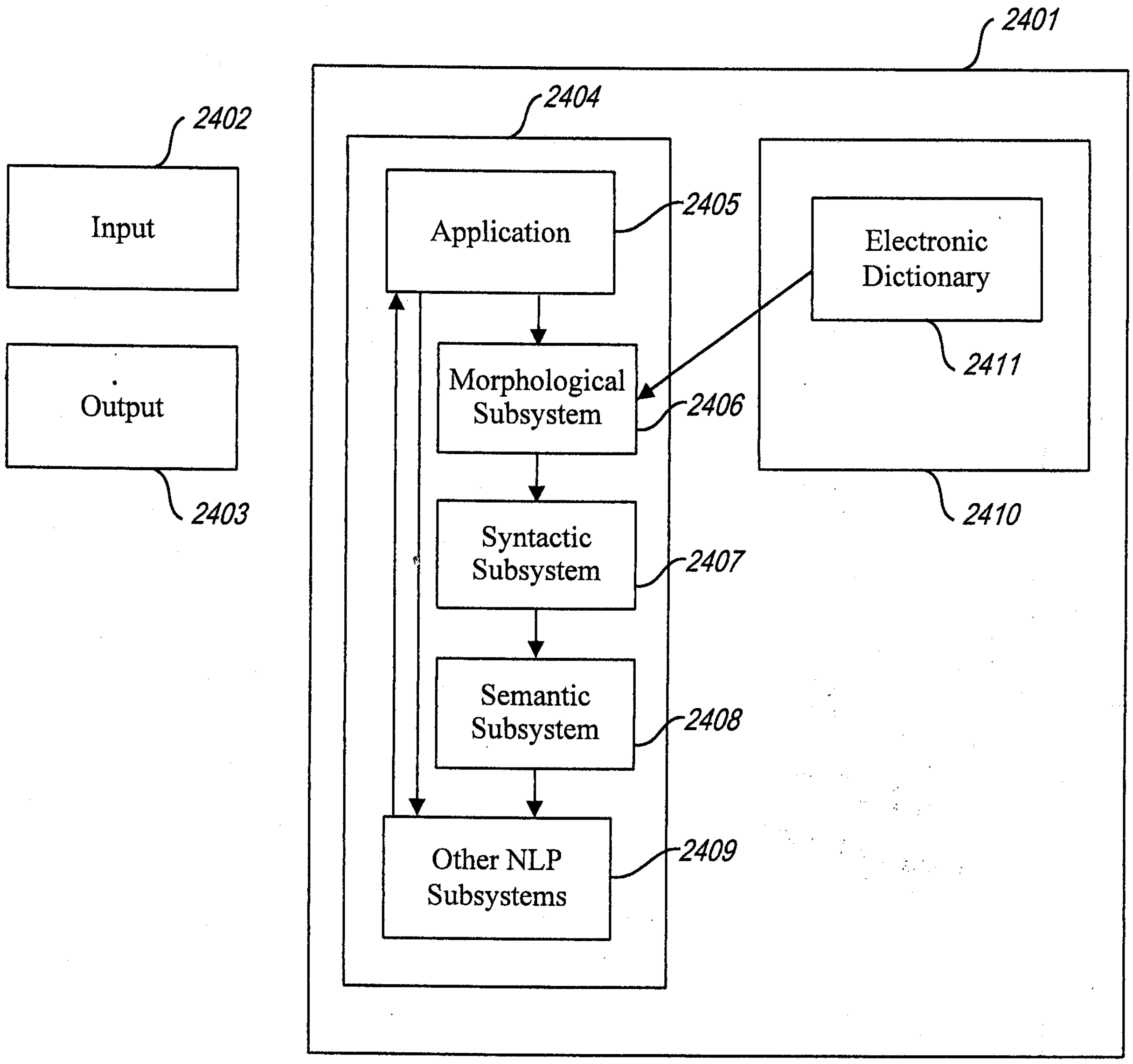

2401
2402
Input
Output
2403
2404
Application
2405
Morphological Subsystem
2406
Syntactic Subsystem
2407
Semantic Subsystem
2408
Other NLP Subsystems
2409
Electronic Dictionary
2411
2410

Fig. 24

The New Semantic Subsystem

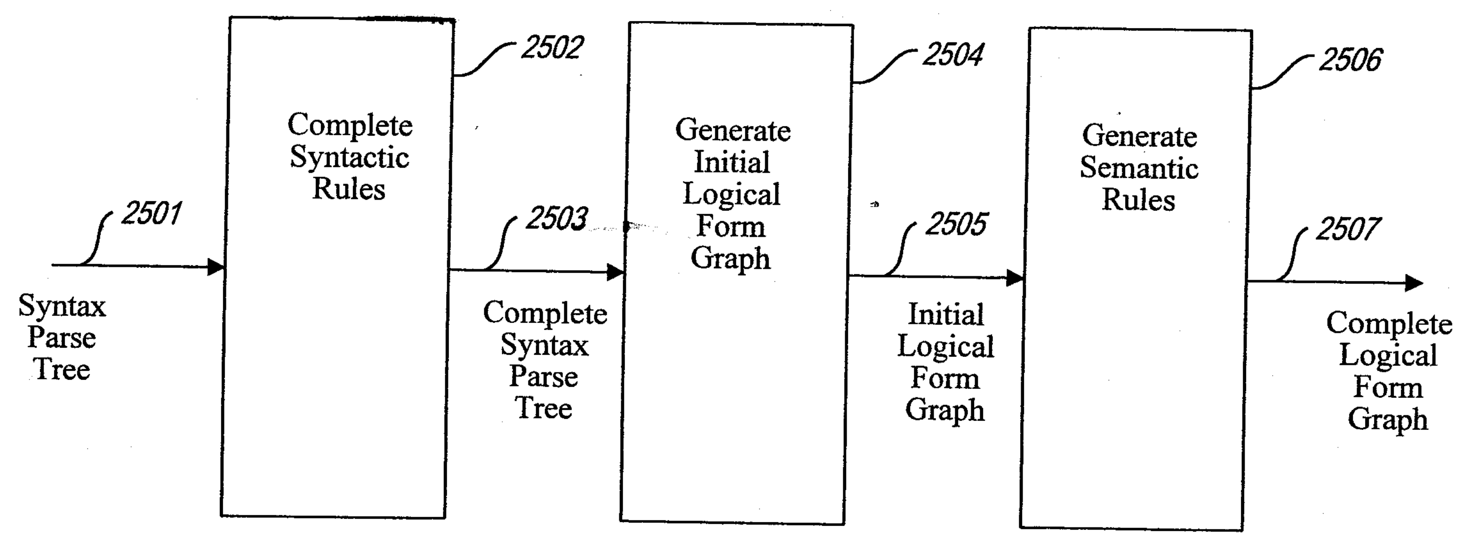


Fig. 25

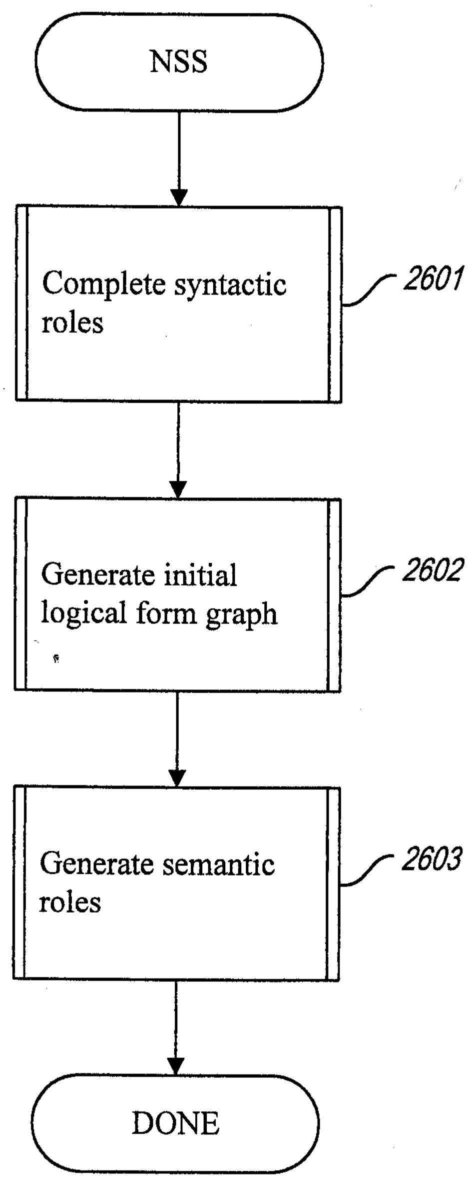

NSS
Complete syntactic roles
2601
Generate initial logical form graph
2602
Generate semantic roles
2603
DONE

Fig. 26

1. PrLF_NPQuantOf: for NPs like "a number of books," makes "books" the head and "a number of" the modifier

2. PrLF_PPQuantOf: same but for PPs, like "with a number of books"

3. PrLF_notAnaphora: prepares to fill VP anaphora like "John thought he would go but Jim though not __________"

4. PrLF_soAnaphora: prepares to fill VP anaphora like "Mary wondered if it was true but Jane knew so __________"

5. PrLF_toAnaphora: prepares to fill VP anaphora like "Chris wanted to go but Pat didn't want to _____"

6. PrLF_You: supplies the understood "you" in commands like "(You) please close the door"

7. PrLF_HowAbout: supplies the understood "you" in constructions like "How about (you) closing the door"

8. PrLF_We: supplies the understood "we/us" in constructions like "Let's (us) go to the movies"

9. PrLF_I: supplies the understood "I" in, for example, "(I) thank you" or "(I) Have not yet received your letter"

10. PrLF_SubjectMods: connects "we" and "all" in, e.g., "We are all reading the book"; connects "he" and "hungry" in, e.g., "He arrived hungry"

11. PrLF_RightShift: connects "the man" and "who was my friend" in, e.g., "The man arrived who was my friend"

12. PrLF_InfclPP: prepares for correct interpretation in constructions like "a person on whom to rely"

13. PrLF_QuantifierEllipsis: having to do with the resolution of pronoun references

14. PrLF_PossessivePronHead: having to do with the resolution of pronoun references

15. PrLF_PossibleCorefsOfProns: having to do with the resolution of pronoun references

16. PrLF_VPAnaphora: identifies and fills missing arguments in all cases of VP anaphora, e.g., "Sarah likes basketball and I do too"

17. PrLF_DistCoords: distributes elements across coordinated structures, like "They washed _____ and dried the dishes"

Fig. 27

PrLF_You

If the Syntax Record

has the attribute "Infinitive"
and does not have the attribute "Subject"
or has the attribute "Verb Phrase Invert" and does not have any of the attributes "Object2," "Yes/No/Question," or "Old Subordinate Clause"
and does not meet the "There Subject Test"
and does not have the "Coordinate Constructions" attribute
and does not have any premodifiers with the node type "Auxiliary Phrase" or the attribute "Modal Verb"
and does not have any premodifiers with the lemma "let" or the node type "Adverbial Phrase,"
and does not have the node type "Abbreviated Clause," "Auxiliary Phrase," "Complement Clause," "Infinitive Clause," "Noun Relative," "Past Participle Clause," or "Relative Clause"
and does not have a parent with the node type "Past Participle Clause"
and if the head of the parent has node type "Conjunction,"
then the parent does not have a "Subject" attribute and does not have the node type "Auxiliary Phrase," "Complement Clause," "Infinitive," "Noun Relative," or "Relative Clause"
and if there is an Auxiliary Attribute on its Head
then for all its Premodifiers their Lemma must not be "neither" nor "so,"
and if it has a Do Modifier,
then it must have an Infinitive attribute and either there must not be a Modal on the First Verb Attribute, or the Lemma of its First Verb must be either "dare" or "need,"
and it if has a Perfective attribute,
then its Lemma must be do,
and if it has a Verb Phrase Invert attribute,
then either there must not be an L9 attribute
or there must not be a Comma attribute and for all of its Premodifiers their node type must not be equal to "Prepositional Phrase" and for all of its Premodifiers their node type must either not be "Adverbial Phrase" or there must be a Comma attribute or the node type of their Head must be an Interjection,
and has neither "ect" nor "ect." as its Lemma,
and if its Lemma is "suffice,"
then the Lemma of its Object1 cannot be "it,"
and if its Lemma is "thank,"
then the Lemma of its Object1 cannot be "you,"

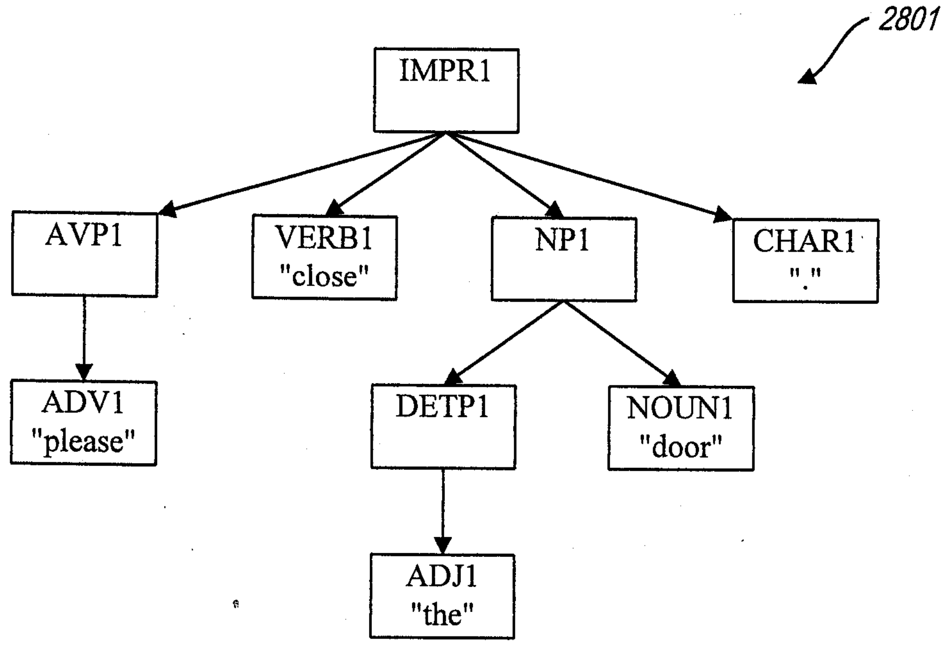
Then

create a pronoun record for the lemma "you";
make the Subject attribute of the syntax record be a copy of the pronoun record and set the Segtype to be "NP," set the node type to be Segtype, and set the head attribute to be the pronoun record;
and set the premodifiers of the syntax record to be the value of the subject attribute plus all of the original premodifiers and set the Undersubject attribute flag.

Fig. 28A

Sentence represented by parse tree: "Please close the door."

Syntax parse tree generated by syntactic subsystem:



Rule PrLF_You

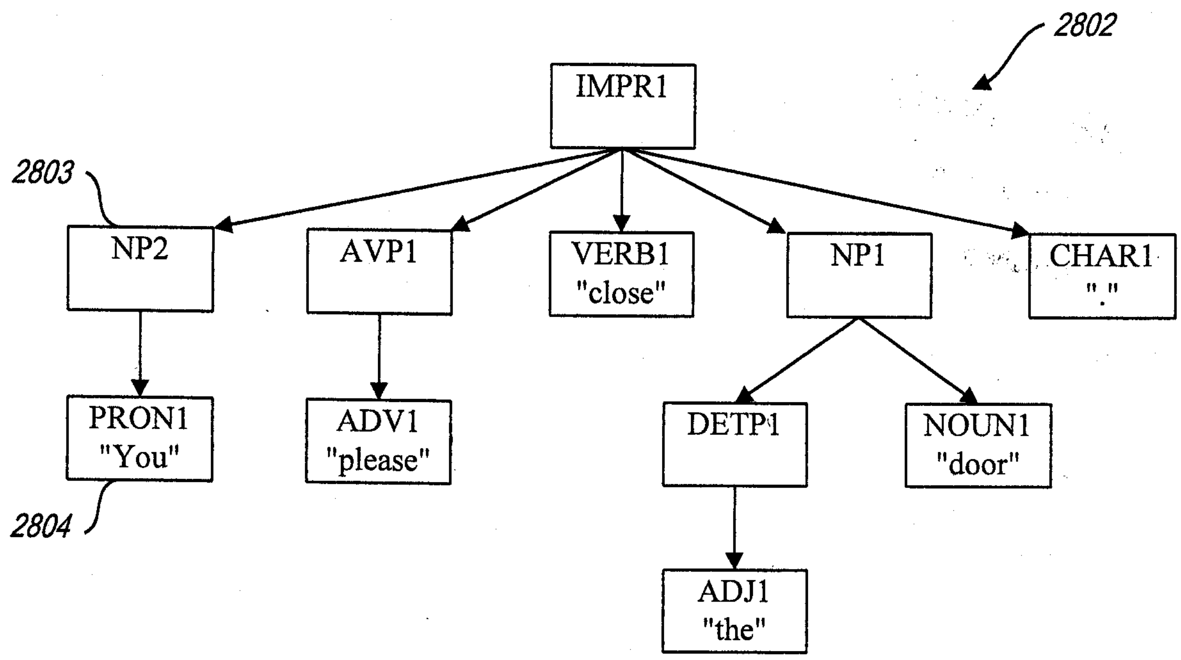


Fig. 28B

1. TrLF_LongDist1: locates NPs that are removed from their semantic heads and reattaches them, e.g., "Who did John say that Mary likes_(who)_?"

2. TrLF_LongDist2: performs the same kind of long-distance attachment for AJPs, INFCLs, PPs, PRPRTCLs, PTPRTCLs, SUBCLs

3. TrLF_PhrasalVerb: defines semantic objects of certain verbs when they appear hidden inside PPs: "his hat" is really the semantic object of "took off" in "He took off his hat"

4. TrLF_ControlwNP: e.g., in "Chris told Pat what to eat," "Pat" is really the subject of "eat" and "what" is its object

5. TrLF_ControlwAJP: e.g., in "I find this difficult to believe," "this" is really the object of "believe"

6. TrLF_ForInfcl: used in "for-to" constructions, e.g., in "For Mary to talk to John is easy," "Mary" is really the subject of "talk"

7. TrLF_ForInfclCoords: used in "for-to" constructions that have coordinated PPs

8. TrLF_MoveProp: given our strategy for attachment, it is sometimes necessary to move clauses from a lower to a higher level so that the proper argument structure can be assigned

9. TrLF_ControlatVP: e.g., in "Farmers grow food by using salt water," "farmers" is really the subject of "use salt water"

10. TrLF_PropsAsArgs: some clauses (propositions) can be arguments, e.g., in "Has he to answer the letter?" the object of "has" is "to answer the letter"

11. TrLF_Extraposition: e.g., in "It makes me happy to meet you," the real subject of "makes" is "to meet you" -- "it" is an empty word and must drop out

12. TrLF_FillCoords: fills in missing arguments in coordinated structures

13. TrLF_RedefineSubject: e.g., in "What is John's address?" we interpret "John's address" as the logical subject even though it is not in canonical subject position

Fig. 29

TrLF MoveProp

If the syntax Record

has either a node type of Abbreviated Clause, Infinitive Clause, Present Participle Clause, Past Participle Clause

or if it has a Gerund attribute and an Object of a Prepositional Phrase and

if it has Premodifiers,

then the node type of all Premodifiers must be either Auxiliary Phrase, Adverbial Phrase, or Prepositional Phrase,

and the node type of the Head attribute of the Parent is not "verb"

and this syntax record is the last of the post modifiers of its parent

and this syntax record is not in the coordinates attribute of its parent

and among the ancestors of the parent there is a record whose node type of the Head is "Verb" but none of those ancestors can have a Coordinates attribute (this record will later be referred to as "same ancestor")

and there should be no For To Prepositional Phrase attribute on the parent,

and if the node type equals Infinitive Clause,

then there must be either no WH attribute on PP obj of the parent or the syntax record is not equal to the Nominal Relative of the parent,

and if the node type is either Present Participle or Past Participle,

then its Parent does not have an Object of a Prepositional Phrase,

and if the node type is a Present Participle Clause,

then there must be an 'ING' Complement on the same ancestor

and if the node type is a Past Participle Clause,

then there must be a V8 (code from Longman's dictionary) attribute on the same ancestor and if there is an X1 attribute on the syntax record then there must not be an Object 1

and there is no B3 attribute on its parent,

and this syntax record must follow the head of the same ancestor or there is a passive attribute on the same ancestor

and if the Lemma of the Parent is 'certain'

then the node type of the parent must not be an Adjective Phrase

and if the Lemma of the Preposition is either "as" or "of,"

then there must be a To Noun attribute of its Parent

and if the Lemma of the same ancestor is either "be" or "become"

then either the node type of the Parent must be an Adjective Phrase

or there must be a WH attribute on the Parent

or there must be both a To Noun attribute on parent and no There Subject Test on the same ancestor

or the Lemma of the Parent must be one of the following: "delight," "horror," "joy," "pleasure," "riot," "shame," "surprise," "terror,"

Fig. 30A

TrLF MoveProp

Then

the syntax record whose attributes will be changed is the same ancestor syntax record (see above);

if the Parent of the syntax record has the Subject attribute and the Parent of syntax record also has the Object attribute,

then delete the object attribute from the ancestor;

if the Parent of the syntax record has the Subject attribute and the Parent of the Syntax Record does not also have the Object 1 attribute,

then set the Subject attribute of same ancestor to be the syntax record;

if the same ancestor has

the DI (Longman code) attribute and there is an Object Complement attribute and no Indirect Object attribute and there is a To Infinitive on the syntax record and the Parent of syntax record is the Object

and there is no WH attribute on the Parent of Syntax Record

and either there is an Animate attribute on Parent of syntax record

or there is a Case attribute on Parent of Syntax Record and the Lemma of the Parent of the syntax record is not "it"

or there is a Human attribute on the Parent of Syntax Record

or there is a Proper Name attribute on Parent of syntax record,

then make the Indirect Object Attribute on same ancestor equal to that of the Parent of syntax record;

if there is a To Infinitive attribute on the syntax record and no Passive attribute on same ancestor,

then make the Predicate Complement attribute equal to the syntax record;

if the Parent of syntax record is in the Propositions attribute of same ancestor,

then take that Propositions list and replace the Parent of the syntax record with the syntax record itself in the propositions list;

delete the Infinitive attribute of the Parent of the syntax record;

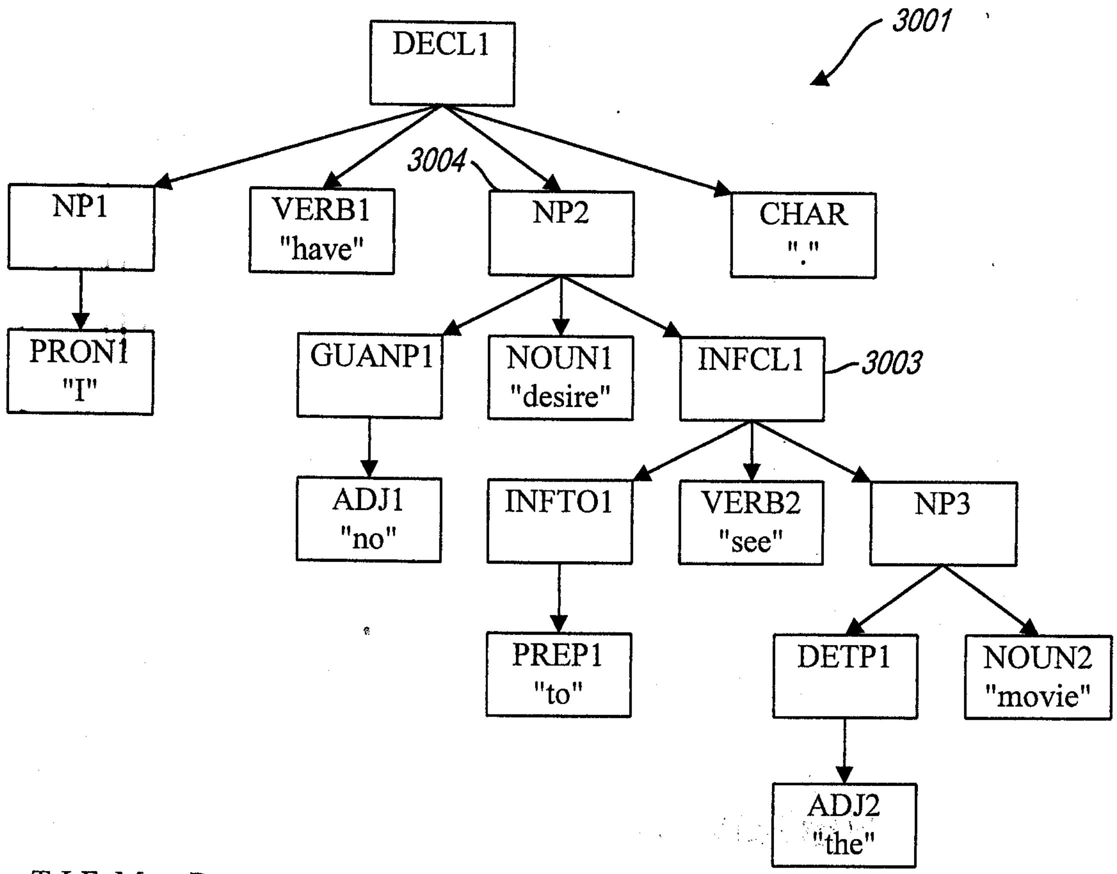
delete the Alternatives attribute on the syntax record;

reattach the syntax record to the same ancestor.

Fig. 30B

Sentence represented by parse tree: "I have no desire to see the movie."

Syntax parse tree prior to applying rule TvLF_MoveProp:



Rule TrLF_MoveProp:

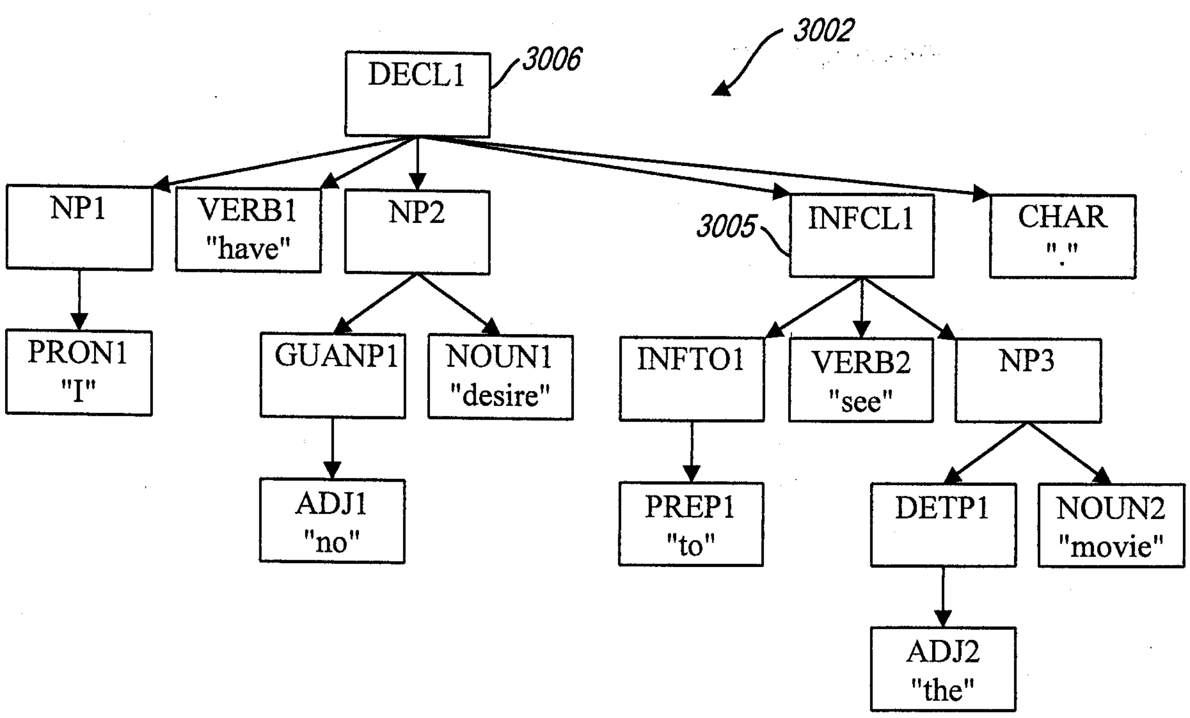


Fig. 30C

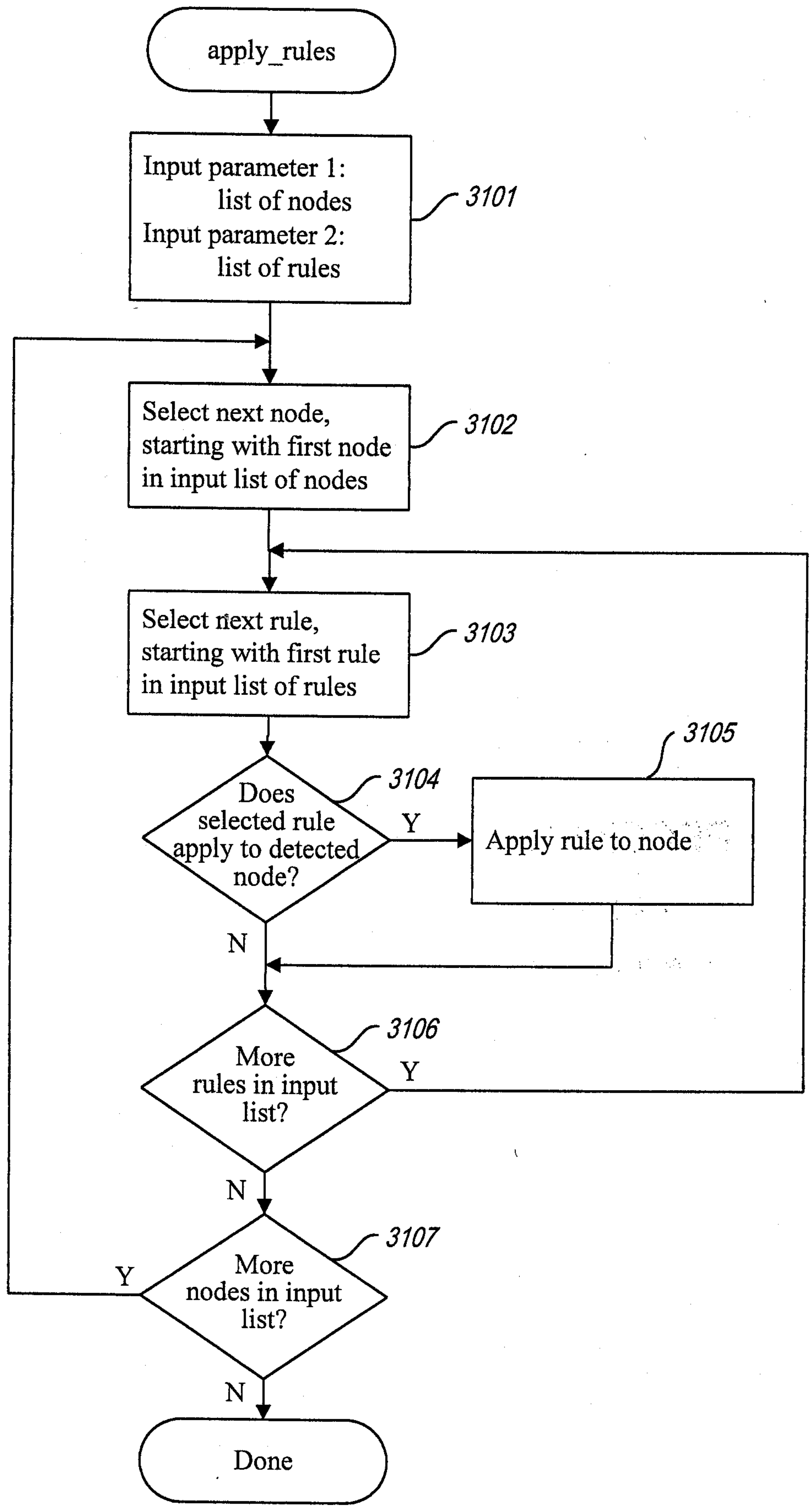

apply_rules
Input parameter 1:
list of nodes
Input parameter 2:
list of rules
3101
Select next node,
starting with first node
in input list of nodes
3102
Select next rule,
starting with first rule
in input list of rules
3103
Does selected rule apply to detected node?
3104
Y
Apply rule to node
3105
N
More rules in input list?
3106
Y
N
More nodes in input list?
3107
Y
N
Done

Fig. 31

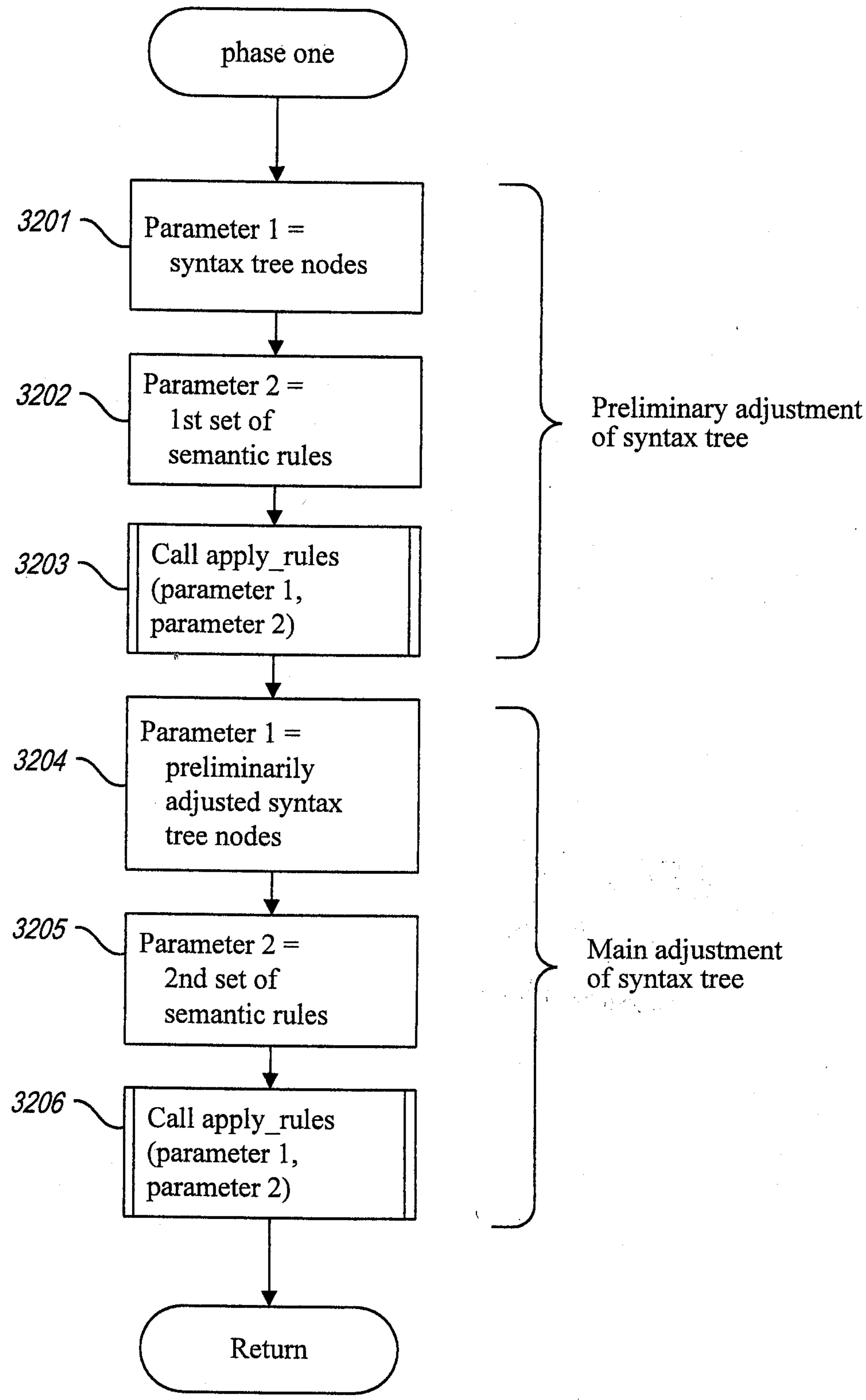

phase one
3201
Parameter 1 = syntax tree nodes
3202
Parameter 2 = 1st set of semantic rules
3203
Call apply_rules (parameter 1, parameter 2)
Preliminary adjustment of syntax tree
3204
Parameter 1 = preliminarily adjusted syntax tree nodes
3205
Parameter 2 = 2nd set of semantic rules
3206
Call apply_rules (parameter 1, parameter 2)
Main adjustment of syntax tree
Return

Fig. 32

1. SynToSem1: creates semantic nodes and a basic semantic graph in es
2. SynToSem2: creates the top-level semantic node and graph for fitted parses
3. SynToSem3: creates semantic nodes for a special subclass of elements in fitted parses

Fig. 33

Rule SynToSem1

If

the Syntax Record
has a Head and
there is no Subordinate Conjunction and
there is no Correlative and
there is no "It subject" and
there is no "There subject" and
there is no Ancestor of the Head for which it is true that that node
is the Emphatic of its Parent and is not a fraction and the head node
is not a verb and
if the segment is the Relative Pronoun of its Parent,
then there must not be a Nominal Relative on the Object of its Parent
and for all of its Parents last records there must not be a VPDone attribute and
if the lemma equals 'that'
then there must not be an Extra Position on the Parent of the Parent and
the node type is not "Auxiliary Phrase," "To Infinitive," "Determiner Phrase,"
or "Tag" or
there is a Possessive attribute or
there is an EVR attribute or
the Lemma equals "other" or
there are Coordinates and for all of those coordinates there is either a
Possessive attribute or an EVR attribute or the lemma is "other" and
if the node type is "Adverb Phrase"
then if the node type of Parent equals Prepositional Phrase
then the segment must not be the first of the Premodifiers of its Parent
and
either the Lemma must not be equal to 'well' or there must not be any Degree
attribute or there must not be any Weak Obligation on the Parent and
If the node type of the Head is a Conjunction or a Preposition,
then the segment node must not be a Conjunction of the Parent and the
segment node must not be a Preposition of the Parent and
If the node type is a Conjunctive Phrase
then there must not be any Coordinates of the Parent or there must not be a
Coordinate Conjunction attribute and
If the node type is a Quantifier Phrase,
then the Lemma of the Head must not be "no" and
If the word could have been an Interjection
then the node type must not be an Adverb Phrase or
there must be Premodifiers or
there must be no comma or
the segment must be the Post Adverbial of the Parent or
the number of Post Modifiers must be greater than one and
If there is an Intensifier attribute
then either the node type of Head of Parent is a "verb" or
the node type of Parent equals "fitted" or
there is an Adverbial Phrase attribute or
there is a WH marker and a Nominal Relative on the Parent and
If there is a Preposition attribute,
then there must be an Object of the Prepositional Phrase or
there is a Particle attribute on the Parent or
the word also could have been an Adverb and

Fig. 34A

Rule SynToSem1

If the Lemma is "also," "so," or "too,"
    then there must not be a VPDone attribute on the Parent and
If the Lemma is "as" or "than"
    then there must not be a Comparative on the Parent and
If the Lemma equals "for"
    then there must not be a "for to" Preposition on the Parent and
If the Lemma equals "it"
    then if there is a Topic Clause on the Parent
        then the segment must be equal to the Subject of the Parent or
        the segment must be equal to the Object of the Parent and
If the Lemma equals "it"
    then the segment must not be in the Premodifiers of the Parent or
    If there is an Extra Position on the Predicate Adjective of the Parent
        then there must not be a Right Shift attribute on the Parent and
        if there is a WH Question attribute on the Parent
            then there is no "To Infinitive" attribute on the
            Predicate Compliment of the Parent and it's
            not the case that for any of the Post Modifiers of the
            Parent that there is a "For to" prepositional phrase
            on the first of the Premodifiers and
If the Lemma equals "let"
    then the node type is not equal to "Adverb Phrase" and
If the Lemma equals "not"
    then there must be a Coordinate Conjunction on the Parent and
If the Lemma equals "there"
    then there must not be any Skipover attribute and
    either there must not be any "Yes No" question on the parent or
        there must not be a Copulative on the Parent or
        there must be a T1 attribute on the Parent or
        the first token integer must be greater than the first token integer of
        the Subject of the Parent and
If the Lemma is "whether" or "whether or not"
    then the node type of the Nominative Relative must not be an
    Infinitive Clause" and
the Lemma must note be "etc," "etc.," "the," "hm," "mm," "uh," or "um"

Then

(If syntax node was kept, then create a corresponding semantic node.)
If the node type of the syntax node is a Noun Phrase and
    there are Bases on the syntax node and
    there is a Subject or an Object on the syntax node,
then make the Predicate equal to the Lemma of the first Basis of the syntax node
Else if there is a Proper Noun attribute on the syntax node and
    if there is a dictionary entry for that word,
then make the Predicate equal to that dictionary entry
Else set the Predicate equal to the Lemma of the syntax node
If the word could have been a Verb and has a Present Participle attribute and
    if for any of the Premodifiers of the syntax node there is a Possessive or
    if the Lemma of the Preposition of the first of the Postmodifiers of the syntax node is
    "by," "for," "of," or "to"

Fig. 34B

Rule SynToSem1

then make the Predicate equal to the Lemma of the Verb entry of the Part of Speech
Record

Copy the appropriate fields from syntax node to the semantic node.

Go through each of the Premodifiers of syntax record and examine each Premodifier

For each record of Premodifiers of the syntax record

if there is a semantic node on the record and
if the semantic node of the record is not in the temporary modifiers attributes of this semantic record and there is no Skipover attribute on the record and the record is not equal to the Preposition of the Parent of the record and the record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record, or Coordinate Subordinate Clauses
then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record

For each record of the Postmodifiers of the syntax record

if there is a semantic node on record and
if the semantic node of record is not in the Temporary Modifiers attributes of this semantic record and there is no Skipover attribute on record and record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record or Coordinate Subordinate Clauses
then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record

If there are Coordinates of the syntax record and no Coordinates of the Prepositional Phrase on that syntax record and no Coordinate Subordinate Clauses
then

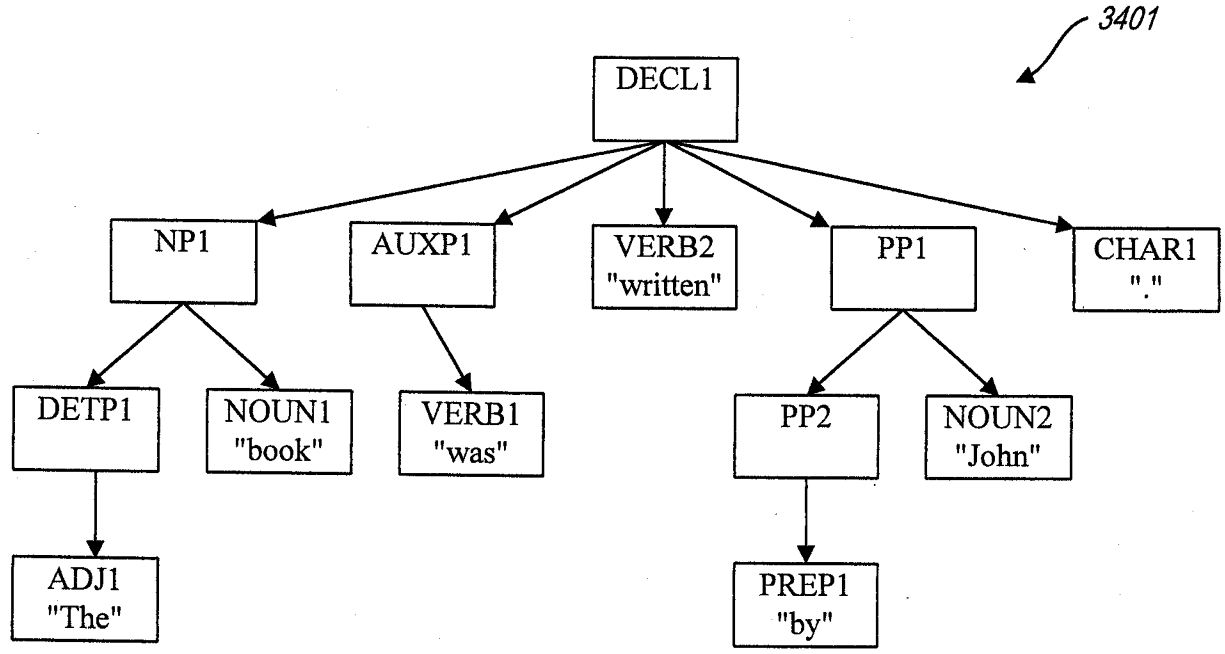
for each of the Coordinates of syntax record

if there is a Semantic node on record,
then add that Semantic node to Coordinates attribute on this new Semantic record.

Fig. 34C

Sentence represented by syntax parse tree: "The book was written by John."

Syntax tree prior to application of rule SynToSem1:



Rule SynToSem1:

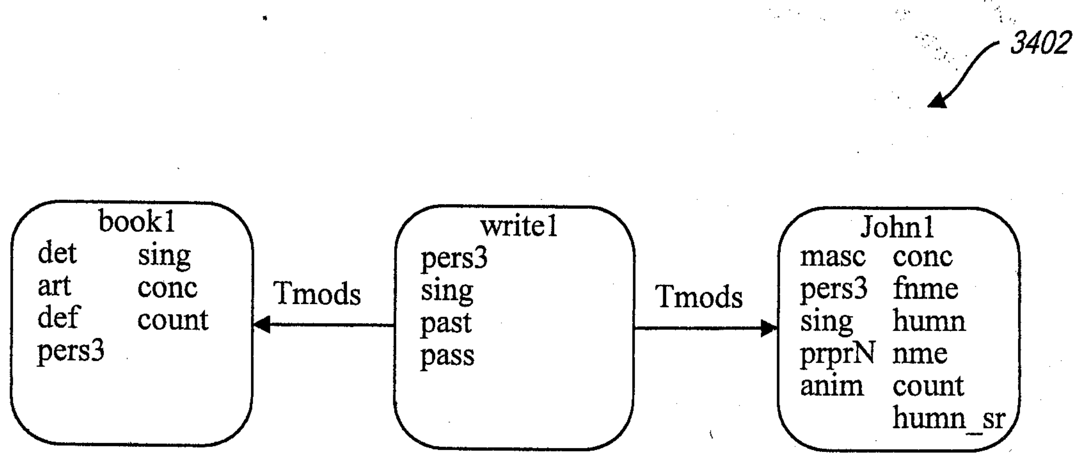


Fig. 34D

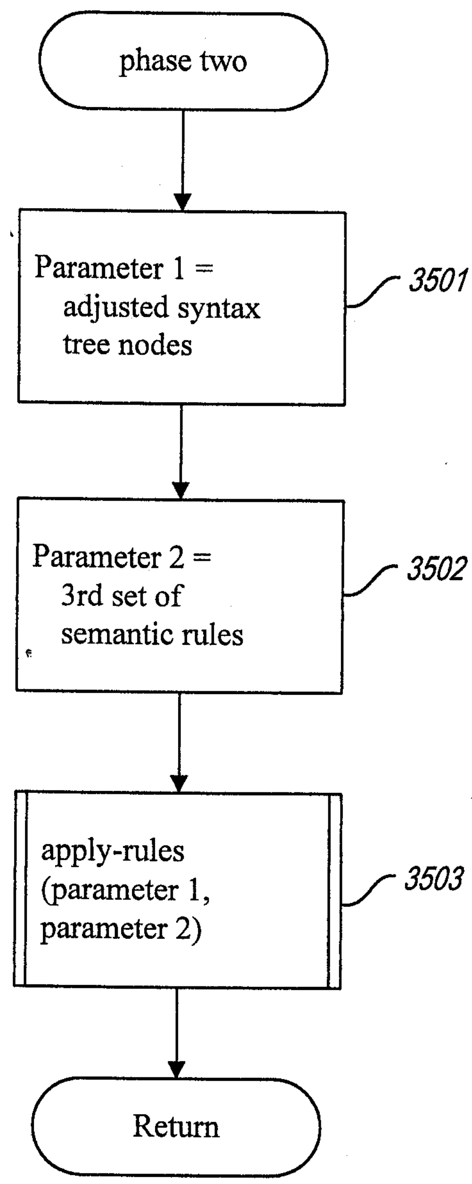

phase two
Parameter 1 = adjusted syntax tree nodes
3501
Parameter 2 = 3rd set of semantic rules
3502
apply-rules (parameter 1, parameter 2)
3503
Return

Fig. 35

1. LF_Dsub1: creates the Dsub (deep subject) label for subjects of clauses in the active voice

2. LF_Dsub2: for passive-voice clauses, if there is a "by"-PP, identifies this PP as the Dsub of the action

3. LF_Dobj1: creates the Dobj (deep object) label for, e.g., direct objects of clauses in the active voice

4. LF_Dobj2: for passive clauses, identifies the syntactic subject as the deep object of the action

5. LF_Dobj3: for clauses like "The door opened," identifies "the door" as the logical object of the action

6. LF_Dobj4: for constructions like "the nomination of the candidate," identifies "the candidate" as the logical object of an action of nominating

7. LF_Dind1: creates the Dind (deep indirect object) label for, e.g., "Mary" in "John gave Mary the book"

8. LF_Dind2: identifies the deep indirect object ("Mary") in paraphrases like "John gave the book to Mary"

Fig. 36

9. LF_Dind3: chooses the right deep indirect object in trickier constructions like "The book was given her"; "She was given the book"

10. LF_Dnom: creates the Dnom (deep nominative) label for predicate nominative, e.g., "our friends" in "They are our friends"

11. LF_Dcmp1: identifies the complement ("president"; "italic") in, e.g., "elect Tom president"; "make the word italic"

12. LF_Dcmp2: identifies the complement in trickier constructions, e.g., in "He gave Tom a place to call his own," "his own" is the Dcmp of "call"

13. LF_Dadj: creates the Dadj label for predicate adjectives, e.g., "blue" in "The sky is blue"

14. LF_CausBy: creates a causative relation where appropriate, e.g., "why" in "Why did you say that?"

15. LF_LocAt: creates a locative relation where appropriate, e.g., "where" in "Where did you find that?"

16. LF_TmeAt: creates a temporal relation where appropriate, e.g., "what day" in "What day did you read that?"

17. LF_Manr: creates a manner relation where appropriate, e.g., "how" in "How did you do that?"

Fig. 37

18. LF_Ptcl: creates a Ptcl node to refer to particles in phrasal verb constructions

19. LF_PrpCnjs: creates temporary relations for PPs and subordinate clauses by naming these elations with the word that is the preposition or conjunction

20. LF_PrpCoord: handles cases of coordinated PPs or subordinate clauses

21. LF_Props: lists remaining clausal adjuncts for any given node

22. LF_Ops: identifies logical operators in noun phrases, e.g., "all" in "all my children"

23. LF_Nadj: lists remaining adjectives that premodify nouns

24. LF_Mods: lists remaining non-clausal modifiers for any given node

Fig. 38

Rule LF_Dobj2

If the Semantic Record

doesn't already have a Deep Object,

and has a Passive attribute,

and has a Subject on its syntactic record (SynNode), and this Subject (which is a syntactic record) has a SemNode attribute (i.e., it has a corresponding semantic record)

and there are no Coordinates

and if there is a Predicate Complement attribute on its syntactic record, then the node type is not "COMPCL" (i.e., it is not a complement clause, as in: "some people were convinced <u>that he had written a book</u>"

and if the SynNode record has either a D5, D6, ObjC, or Psych feature[2]

then either the Object of the SynNode is not a noun phrase,

or the SynNode has an X1[3] feature (as in: He was named Arles")

or the Object of the SynNode has an Animate feature

or there is a Case feature on the Object of the SynNode and its Lemma is not "it"

Then,

give the Semantic record a Dobj attribute with, as its value, the semantic record corresponding to the Subject on the syntactic record

and, remove what is now the value of Dobj attribute from the list of Tmods

---

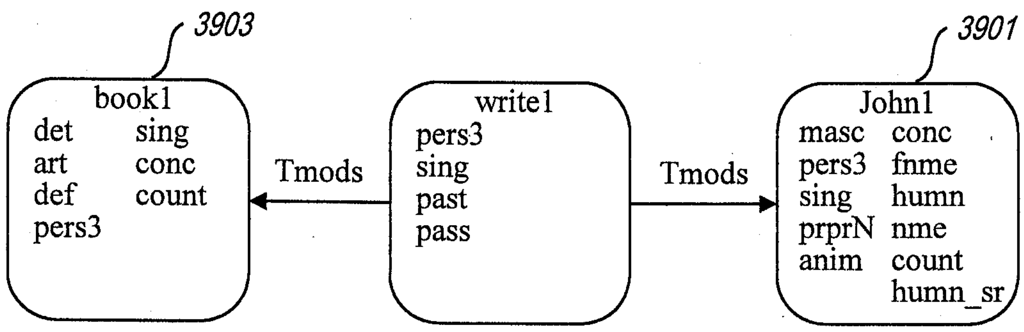
[2] D5, and D6 are features from Longman's Dictionary of Contemporary English; ObjC is verb subcategory for verbs which show object control (e.g., I want Harry to wash the car) and Psych is a verb subcategory for verbs like "scare" "excite".

[3] X1 is a feature from Longman's Dictionary of Contemporary English.

Fig. 39A

Sentence represented by the logical form: "The book was written by John."

Logical form prior to application of rule LF_Dobj2:



Rule LF_Dobj2:

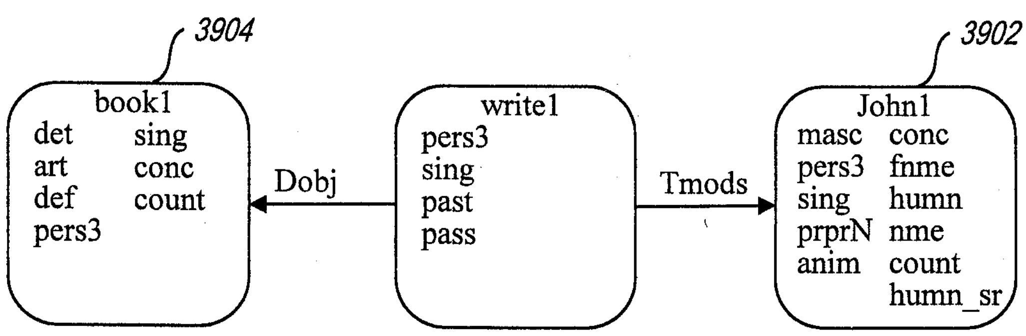


Fig. 39B

1. PsLF_RelPro: identifies proper referents for relative pronouns, e.g., "who" refers to "the man" in "the man who came to dinner"

2. PsLF_ReciprocalAnaphora: handles reciprocal pronouns like "each other" and "one another"

3. PsLF_ReflexiveAnaphora: handles reflexive pronouns like "myself, yourself, him/herself," etc.

4. PsLF_PronAnaphora: identifies possible NP referents for most pronouns

5. PsLF_ProtoAnaphora: handles special cases of pronouns which can agree with just about any NP

6. PsLF_NumberEllipsis: handles reference for number words, e.g., "A bird in the hand is worth two (birds) in the bush"

7. PsLF_FillInHead: adds "DUMMY" as a head word in special cases of unclear referents

8. PsLF_NumberCritique: takes note of pronouns that disagree in number with their referents

9. PsLF_FillDsub: fills in "x" as a placeholder for the deep subject in cases where that is missing, e.g., in passives like "The door was opened"

10. PsLF_UnifyProns: if two pronoun nodes refer to the same referent, this rule unifies them

11. PsLF_UnifyCopies1: unifies some nodes that should be identical

12. PsLF_UnifyCopies2: unifies other nodes that should be identical

13. PsLF_RaiseModality: deletes some verbs when they serve only an aspectual purpose, e.g., in "We used to go there," "used to" is deleted from the graph

14. PsLF_RaisePcs: makes fitted parses easier to read

Fig. 40

Rule PsLF_PronAnaphora

If the Semantic Record

has a Pers3 attribute, i.e., it is not either first (e.g., I or we) or second person (e.g., you)

and the node type of the head of its syntactic record is either "PRON" (pronoun) or the node type of the head of its syntactic record is "ADJ" (adjective) and it has a possessive attribute

and is not Reflexive

and none of the premodifiers of the Parent of its syntactic node has the Lemma "own"

and the Pred of this semantic record is not "each other" or "one another"

and does not have NonRef attribute (NonRef is an attribute set on words that cannot have a reference, such as true numbers, as in: One plus one is two.

and does not have a Negation attribute

and if it has an Indefinite attribute, then there must also be a Definite attribute

and is not a Wh- word (it does not have a WH attribute)

and is not a Relative

and is not a Distal (Distl) or a Proxal (Proxl) determiner (e.g., "this" "that")

Then

add a FindRef attribute to the semantic record

for each of the records in the list of possible referents;[1]

if

the possible referent has a corresponding semantic record

and the possible referent is not the same as this record (i.e., the antecedent of a noun phrase can not be the noun phrase itself)

and if the head of both the possible referent and of this record's SynNode are pronouns (i.e., have the node type "PRON" as their head), then the possible referent must precede this record (no forward reference to a pronoun; an example of forwards (cataphoric) reference is: with his hat on, the teacher left the room, where "his" refers forward to "teacher"

and if the possible referent is the ancestor of the syntactic record of this record, then that ancestor must have a Prp attribute (i.e., must have a postmodifying Prepositional phrase), and its preposition must be either "in", "to", "for", or "by"

and there is no Time or Space feature on the possible referent

and this record and the possible referent agree in number

and this record and the possible referent agree in gender

and if the Lemma of the SynNode is "they" and the possible referent can be a Mass noun (i.e., the possible referent has a Mass feature),

then the possible referent must also be a Count noun (i.e., it must also have a Count feature).

and if the Lemma of the SynNode is "they" and the possible referent has a Sing feature (can be Singular), and the possible referent does not have a Plur feature (i.e., it cannot be Plural),

then the possible referent is either a Count noun, or the possible referent is a Coordinated noun phrase, or it has a Universal feature, or the possible referent is indefinite and has no possessive, or the possible referent has a Proxal feature,

[1] this list is created in a PrLf rule, so, after syntactic processing but before most logical form processing (it is a list of syntactic records). This is a list of all the words in the sentence which can be referred to, i.e., most of the nouns and pronouns in the sentence

Fig. 41A

Rule PsLF_PronAnaphora

and if there is an ancestor of the possible referent that has a Coords attribute (i.e., has coordinate constituents) (but before there is an ancestor with a Subject attribute) then this ancestor is the same as the ancestor of this record that has a Coords attribute (but before there is an ancestor with a Subject attribute)

then if this record is a possessive (e.g., "his" in "John saw his son")

add the possible referent to the list of possible referents (the value of the Refs attribute)

if:

the possible referent is a genitive

and node type of the head of the possible referent is not a Noun

and the possible referent precedes this record (i.e., the semantic record being processed in this rule

or if:

the possible referent is not the first of this record's Parents

and the first of the Parents of the possible referent is not the first of this record's Parents

and if the possible referent follows this record and if any of the possible referent's ancestors have Coordinate constituents, then there should be no ancestor of this record for which the Parent has Coordinate constituents and for which the Parent is the same as the ancestor of the possible referent that has Coordinate constituents (but before there is ancestor whose node type is "NP")

or else if the node type of Parent of this record's syntactic record is "TAG" (i.e., if the pronoun is in a tag question)

add the possible referent to the list of possible referents (the value of the Refs attribute)

if:

the possible referent is the Subject of the Parent of the Parent of this record (e.g., "they" refers to "someone" in: Someone painted in here, didn't they?)

or else:

if

this record is a prepositional phrase

and this record precedes the Subject of this record's Parent

and the possible referent is the Subject of this record's Parent

then add the possible referent to the list of possible referents (the value of the Refs attribute);

else if

this record is not possessive

and this record precedes the possible referent

and node type of the head of the possible referent is "NOUN" and is not a Dummy noun (i.e., one that cannot be a possible referent)

and if this record is not one of possible referent's ancestors

and if it is not the case that there is an ancestor of this record that has Coordinate constituents and the Lemma of that ancestor is "but" and that ancestor is also an ancestor of this record which has Coordinate constituents

then add the possible referent to the list of possible referents (the value of the Refs attribute)

Fig. 41B

Rule PsLF_PronAnaphora

else if

the possible referent is a Prepositional Phrase
and the Parent of the possible referent is not the Parent of this record's syntactic record
and if the Parent of the possible referent is an Adjective Phrase, then the Parent of the possible referent precedes this record

then add the possible referent to the list of possible referents (the value of the Refs attribute)

else if

there is no ancestor of the possible referent for which the Lemma is "be" (but before there is an ancestor with a Subject) that is the same as the ancestor of this record for which the Lemma is "be" (but before there is an ancestor with a Subject)
and none of the Parents on the semantic record of the possible referent is the same as the possible referent
and if this record precedes the possible referent, then the Head of the possible referent is not either a Noun or an Adjective

then add the possible referent to the list of possible referents (the value of the Refs attribute)

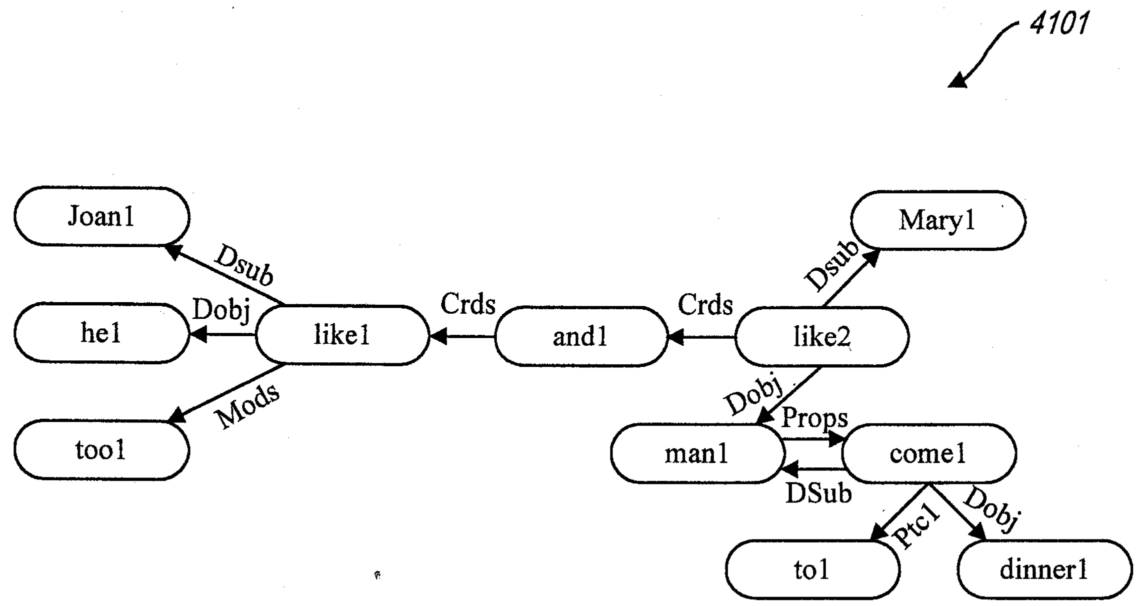
if the possible referent was added to the list of possible referents (the value of the Refs attribute)

then add of RefOf attribute to the possible referent and add this record to that list (provide cross pointers: this record gets a Ref attribute pointing to possible referents, and the possible referents each get a RefOf attribute, pointing back to this record.

Fig. 41C

Sentence represented by logical form: "Mary likes the man who came to dinner, and Joan likes him too."

Logical form prior to application of rule PsLF_PronAnaphora:



Rule PsLF_PronAnaphora:

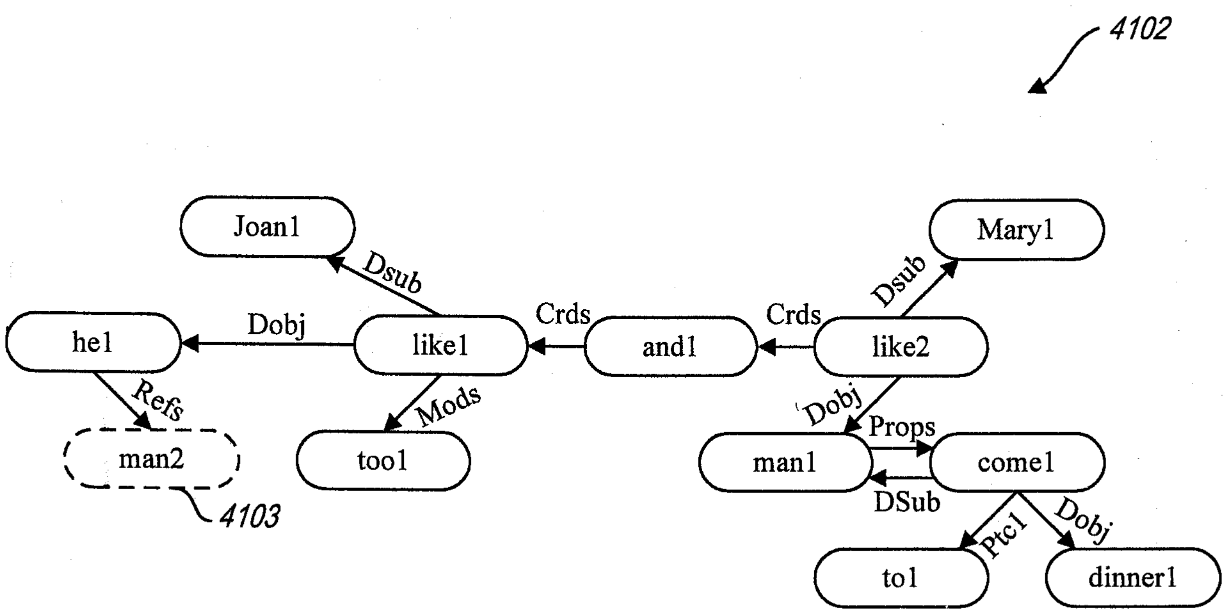


Fig. 41D

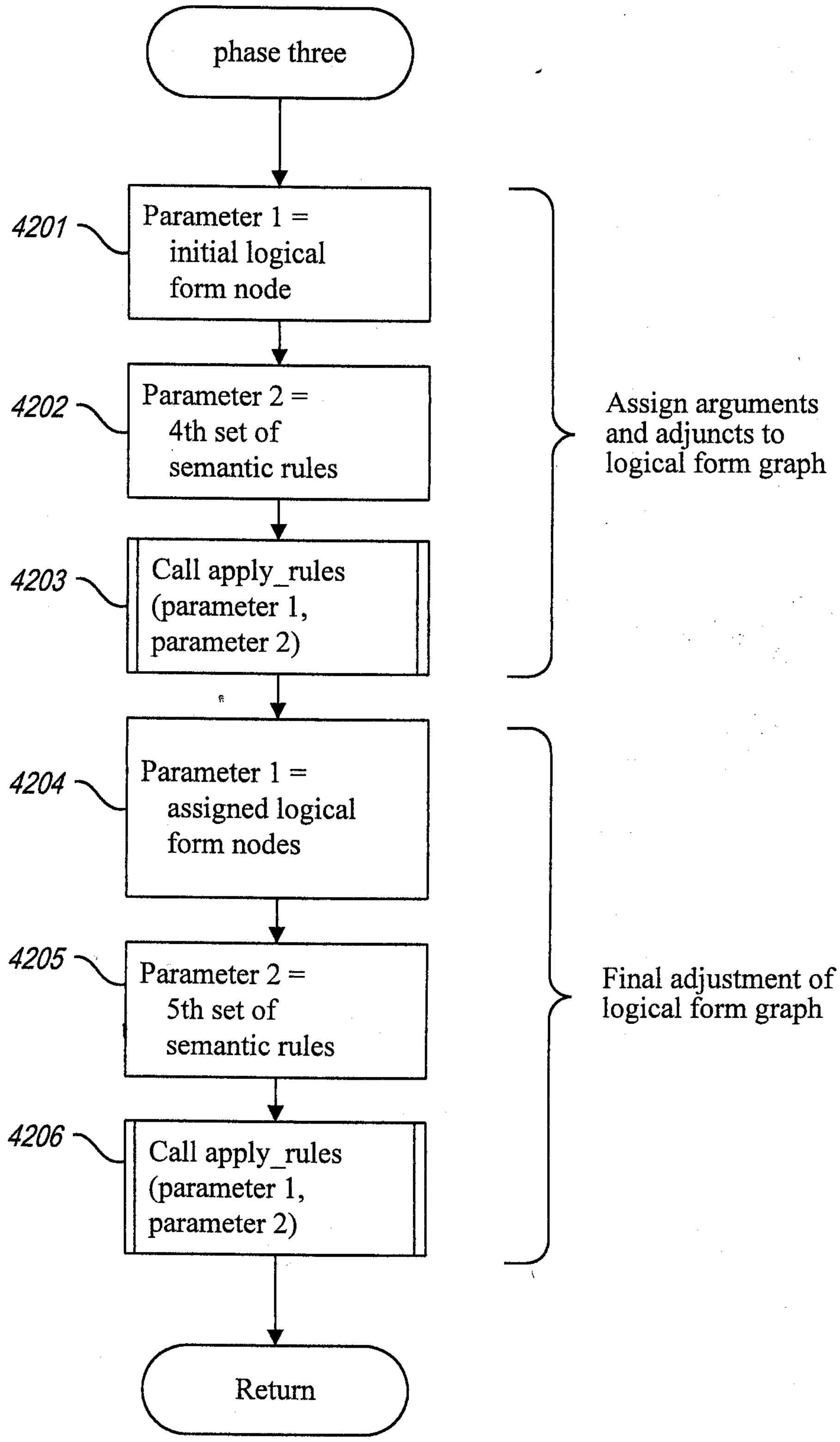

phase three
4201
Parameter 1 = initial logical form node
4202
Parameter 2 = 4th set of semantic rules
4203
Call apply_rules (parameter 1, parameter 2)
Assign arguments and adjuncts to logical form graph
4204
Parameter 1 = assigned logical form nodes
4205
Parameter 2 = 5th set of semantic rules
4206
Call apply_rules (parameter 1, parameter 2)
Final adjustment of logical form graph
Return

Fig. 42

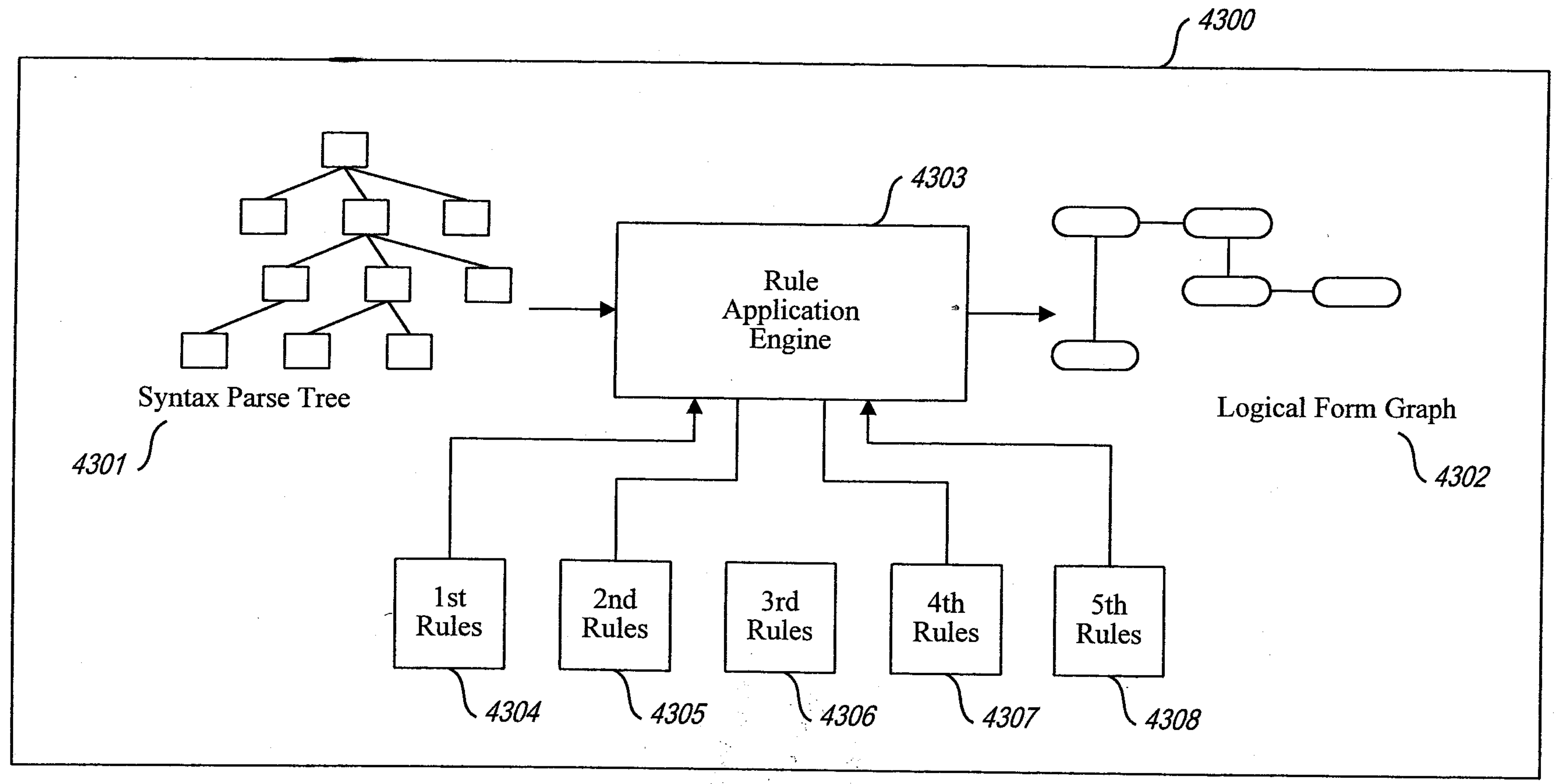

4300
Syntax Parse Tree
4301
Rule
Application
Engine
4303
Logical Form Graph
4302
1st
Rules
4304
2nd
Rules
4305
3rd
Rules
4306
4th
Rules
4307
5th
Rules
4308

Fig. 43

Rule: TrLF_LongDist1 modifies RELCL1 ("whom I met")

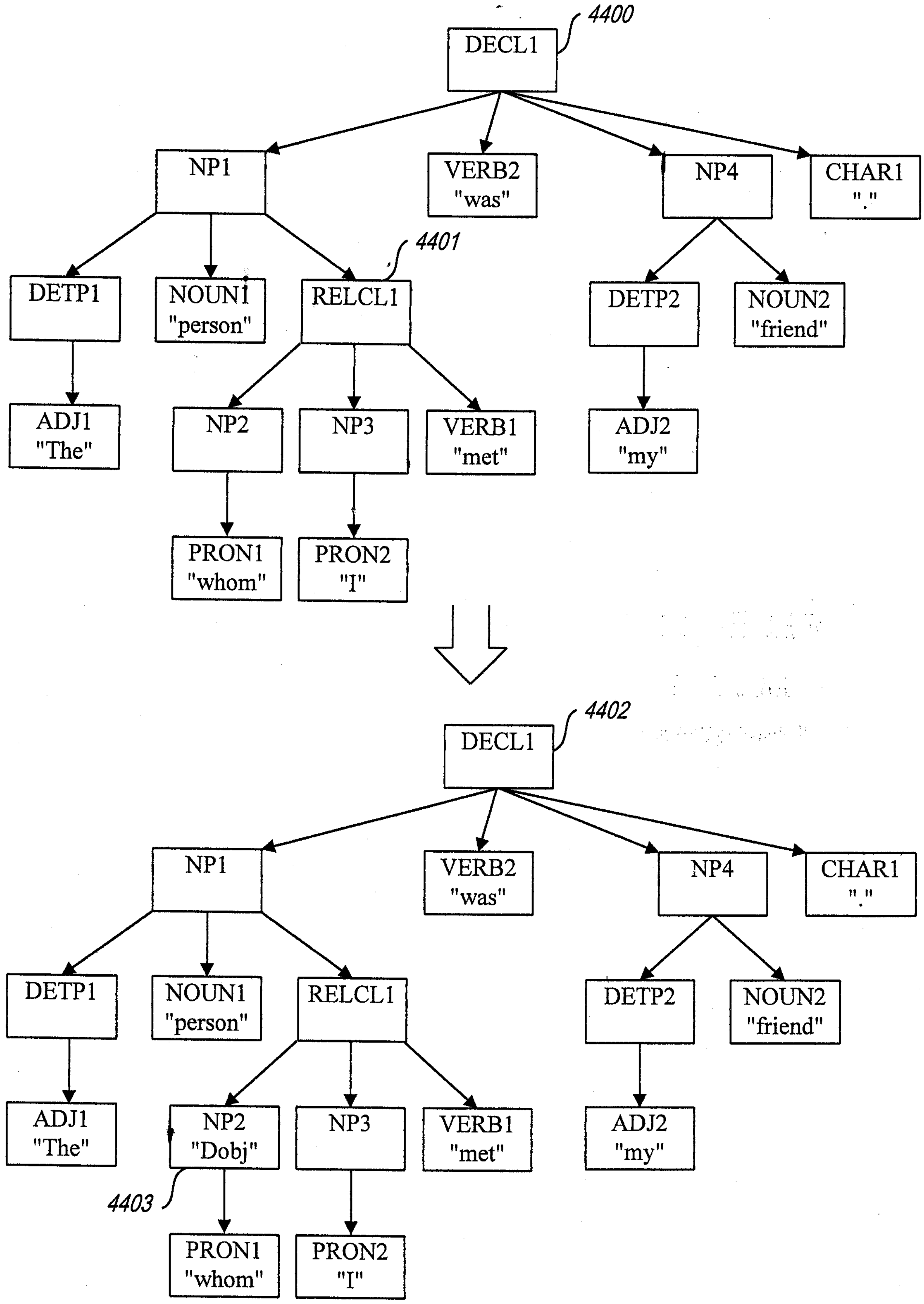
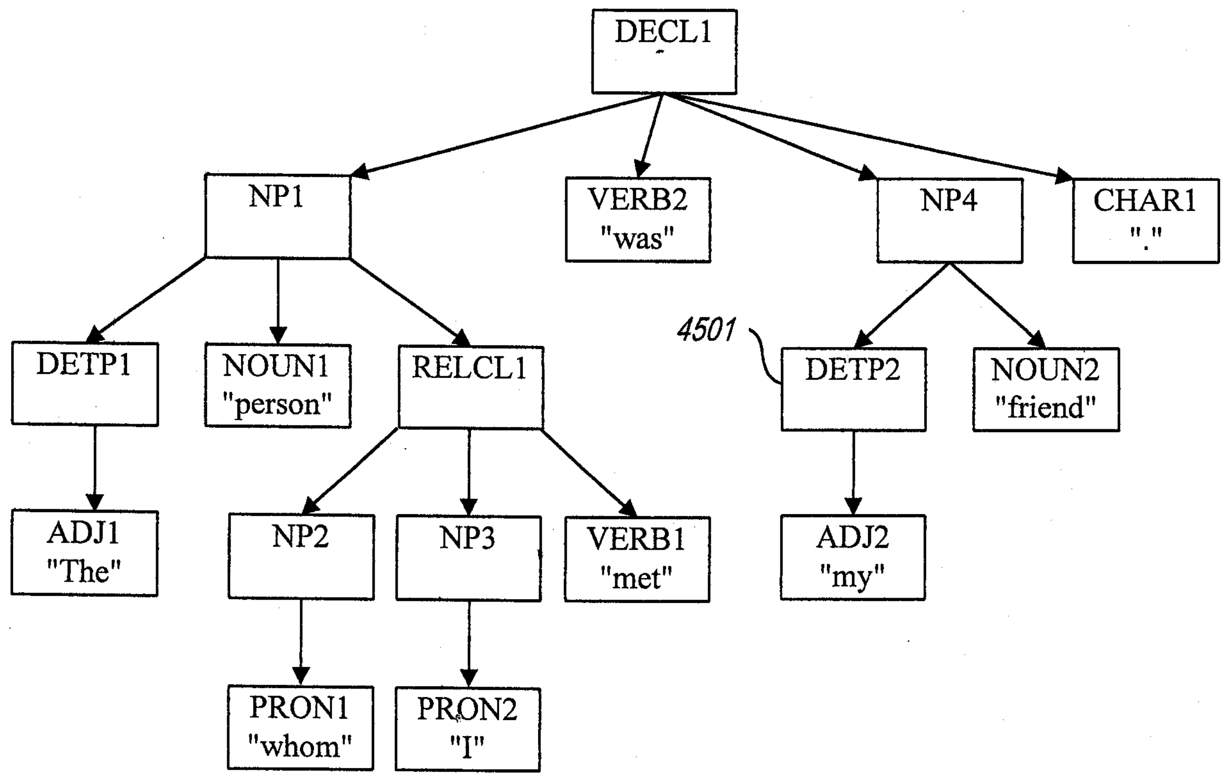


Fig. 44



Rule: SynToSem1 produces logical form graph node from DETP2 ("my")

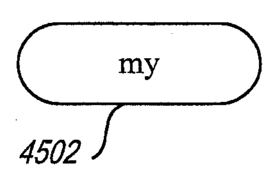
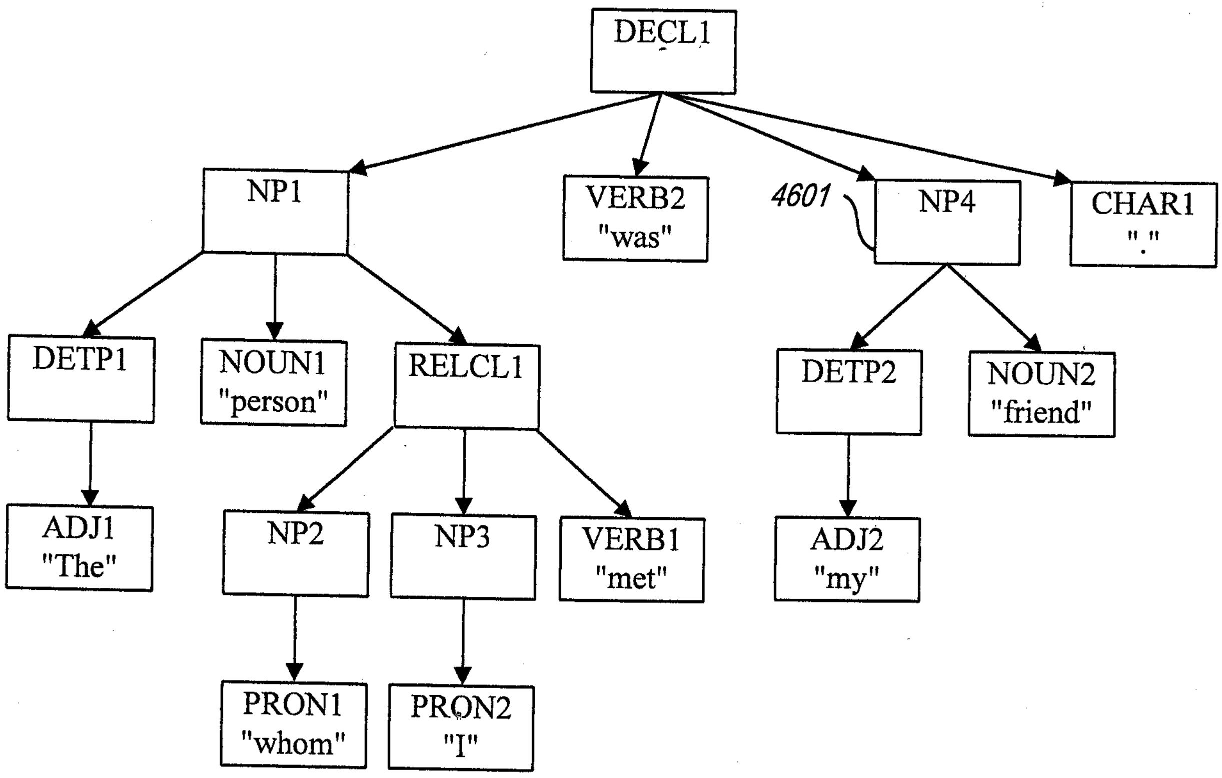


Fig. 45



Rule: SynToSem1 produces logical form graph node "friend" from NP4 ("my friend")

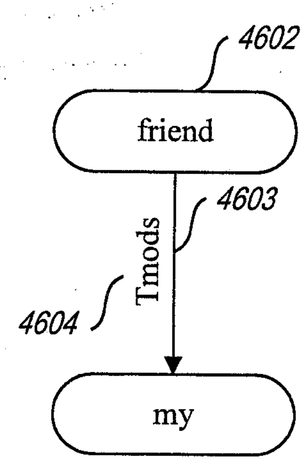
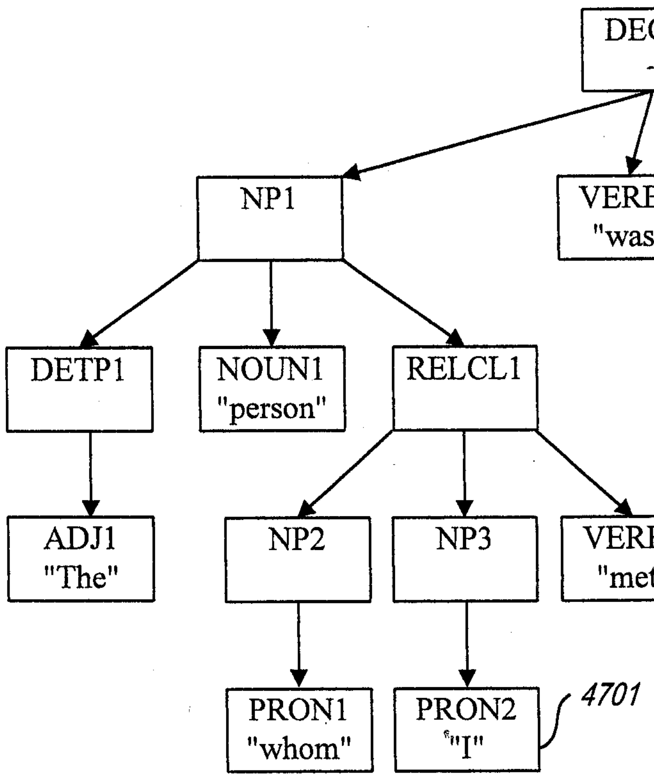


Fig. 46



Rule: SynToSem1 produces logical form graph node "I" from NP3 ("I")

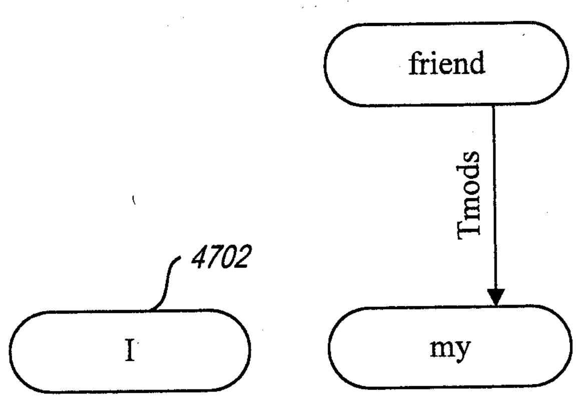
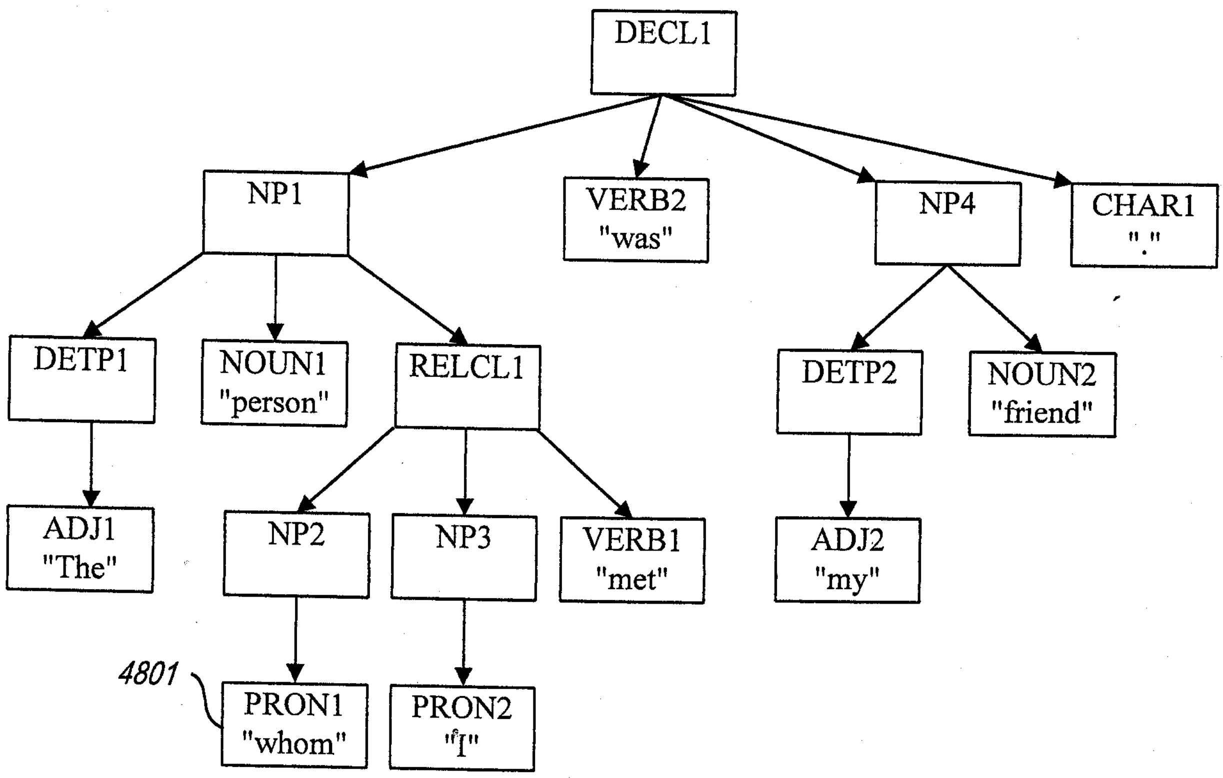


Fig. 47



Rule: SynToSem1 produces logical form graph node "whom" from NP2 ("whom")

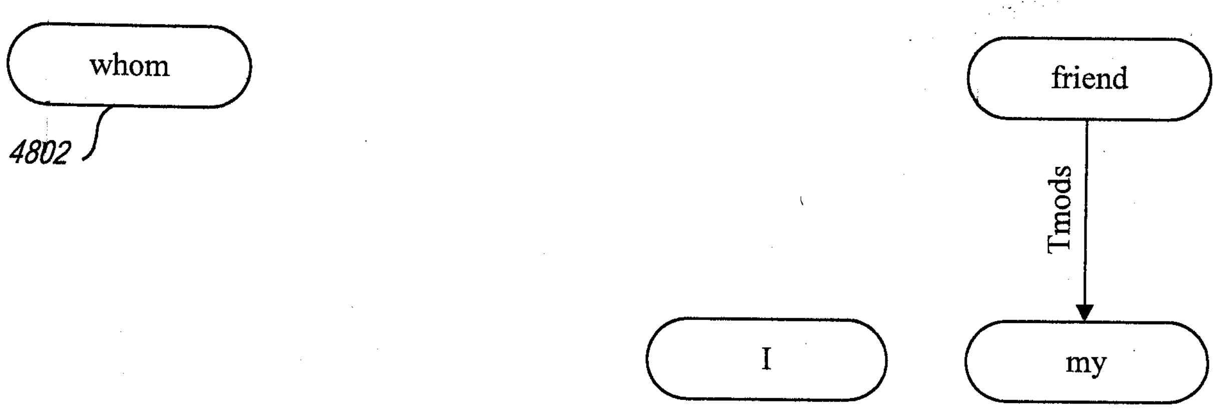
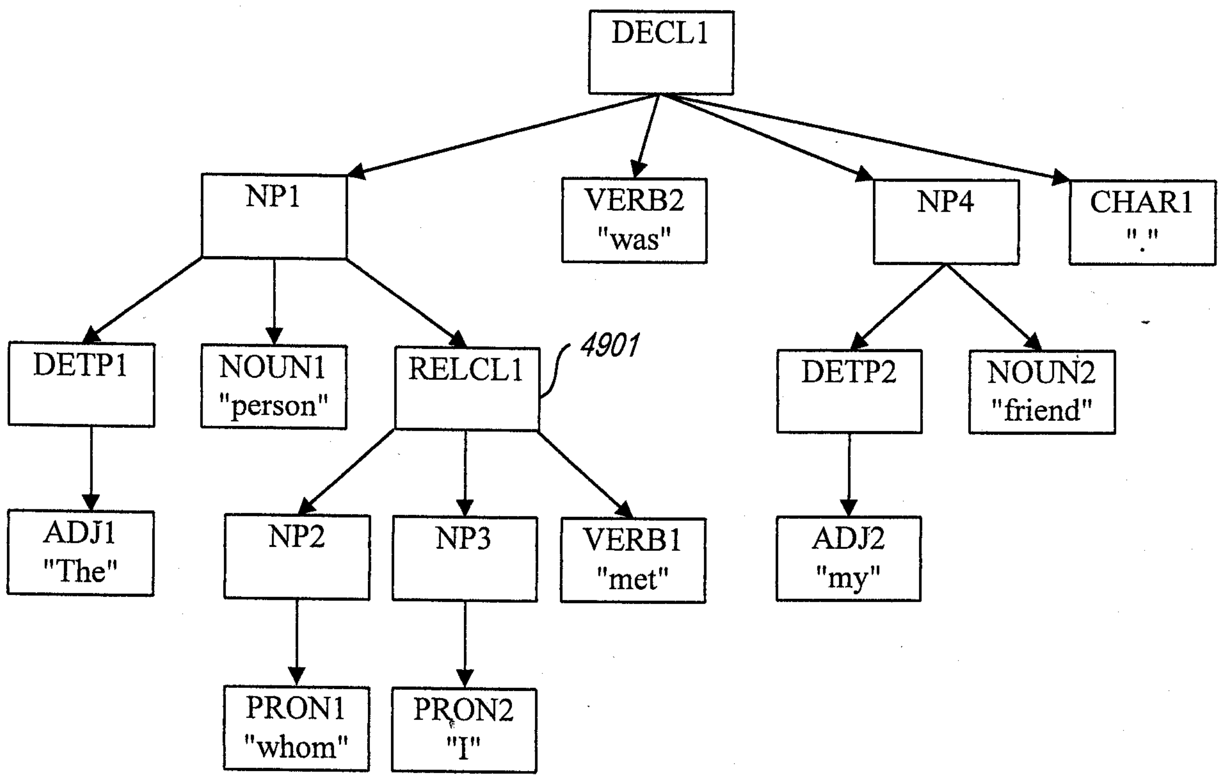


Fig. 48



Rule: SynToSem1 produces logical form graph node "meet" from RELCL1 ("whom I met")

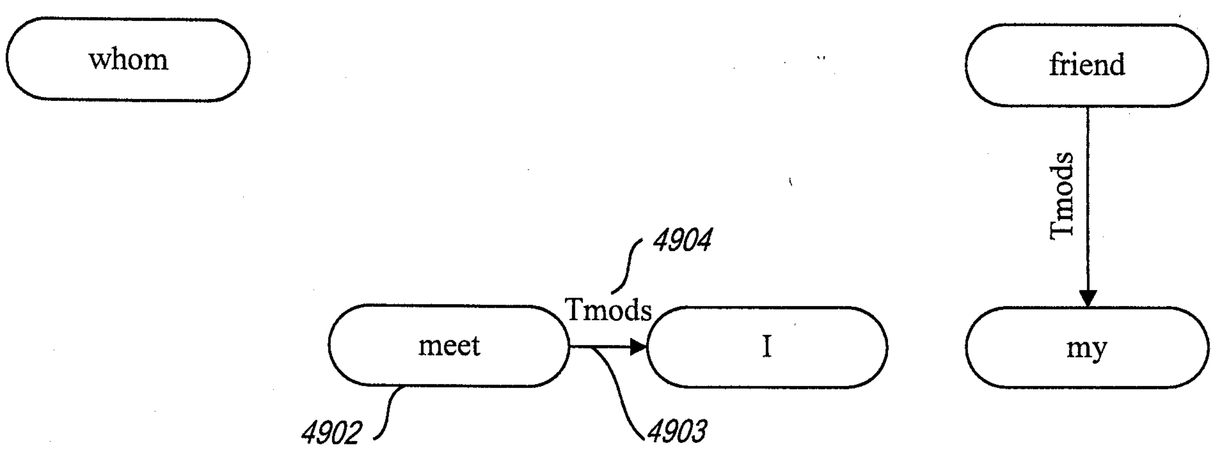
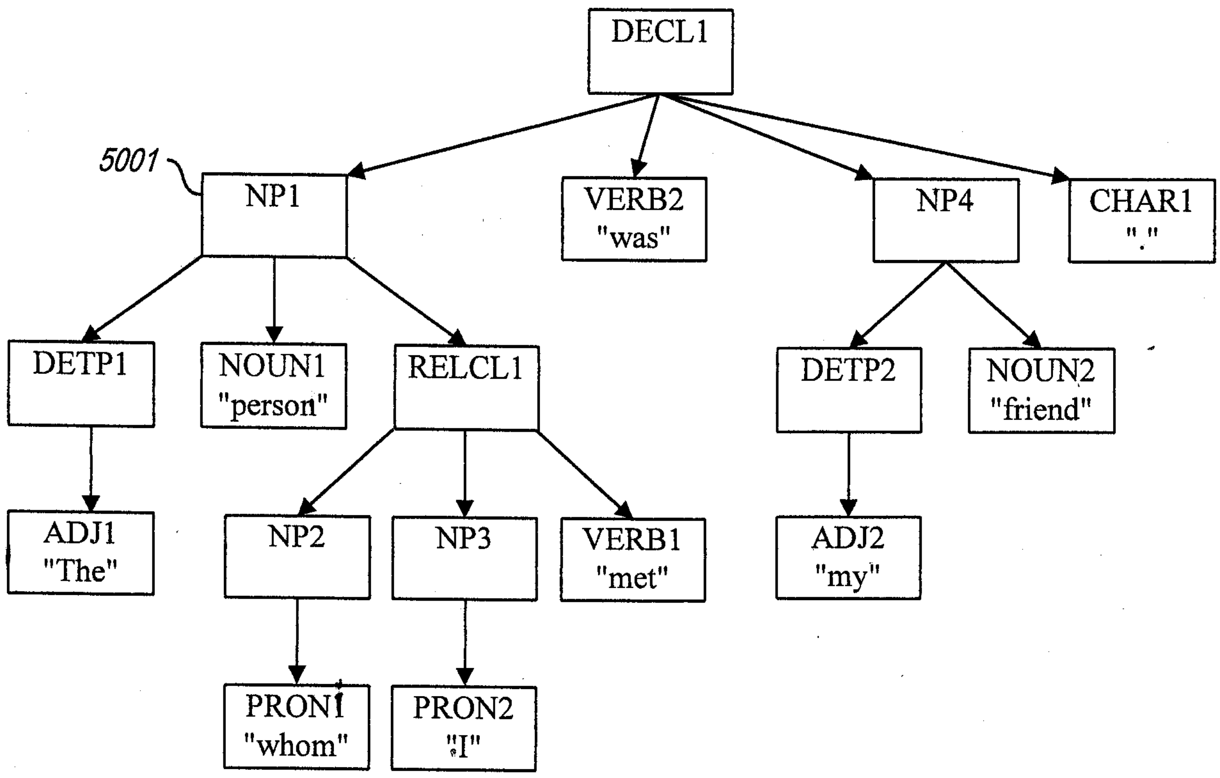


Fig. 49



Rule: SynToSem1 produces logical form graph node "person" from NP1 ("The . . . met")

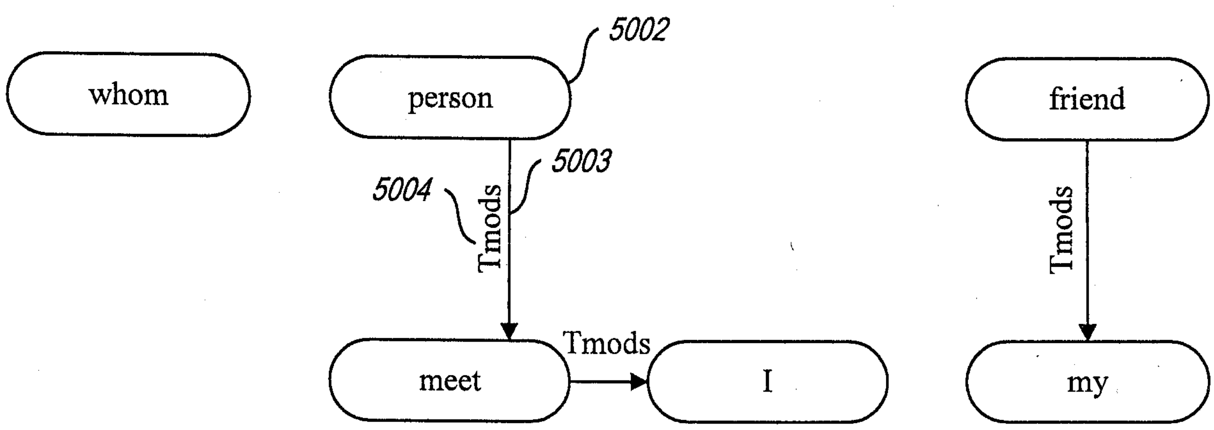
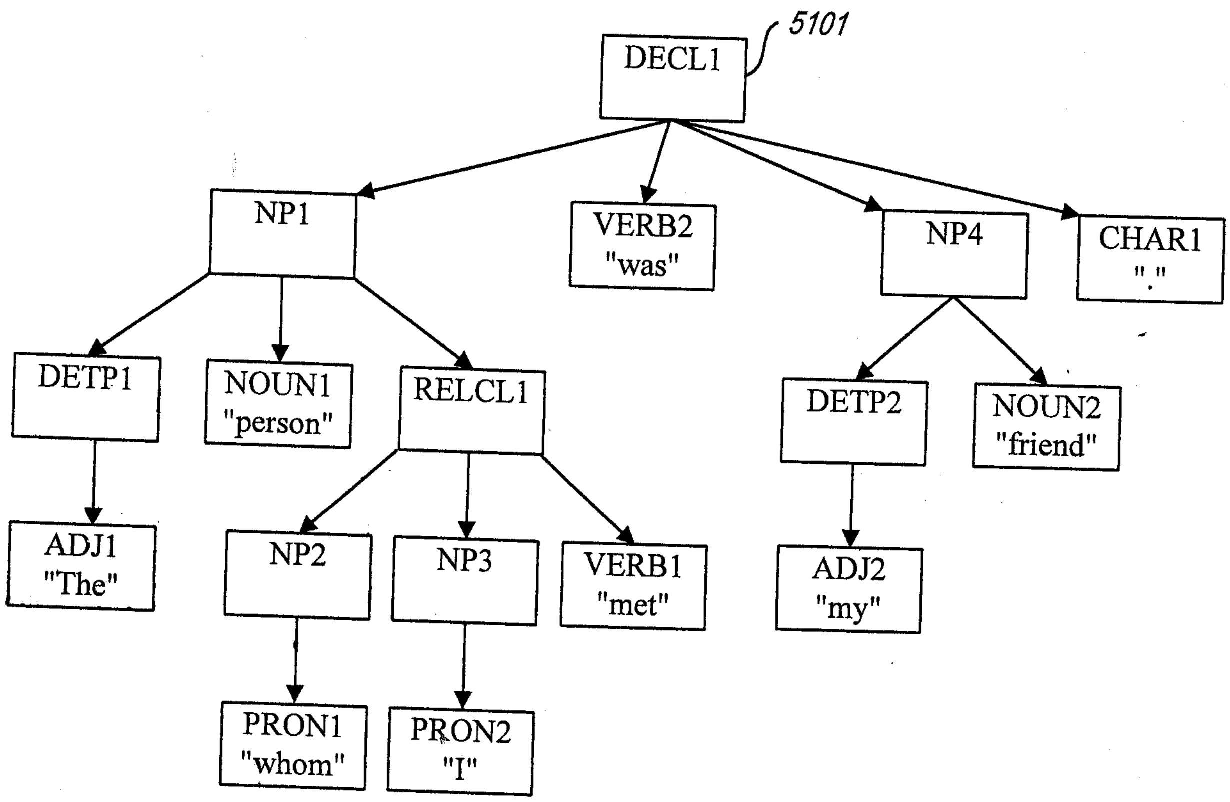


Fig. 50



Rule: SynToSem1 produces logical form graph node "be" from DECL1

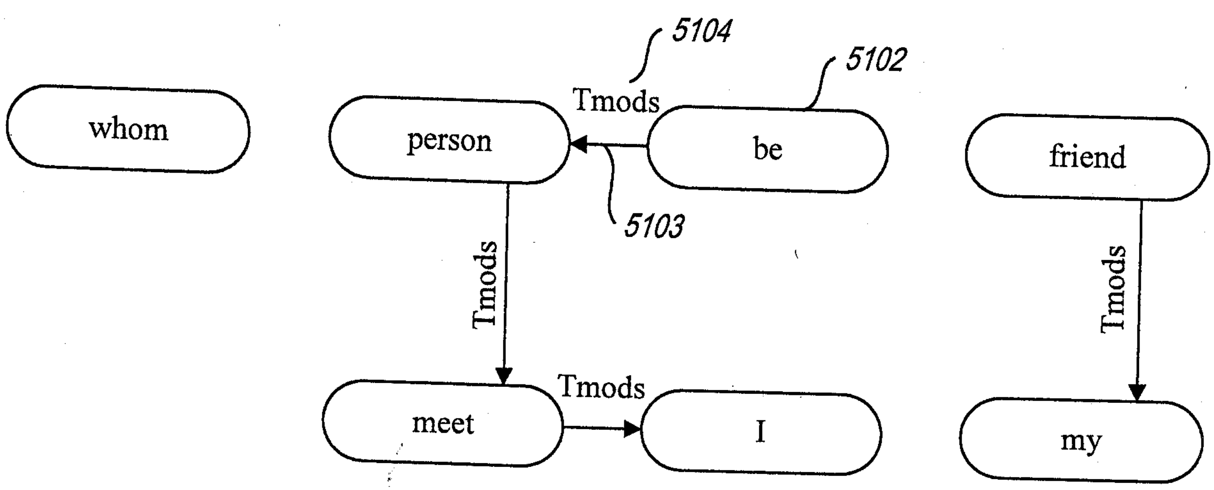
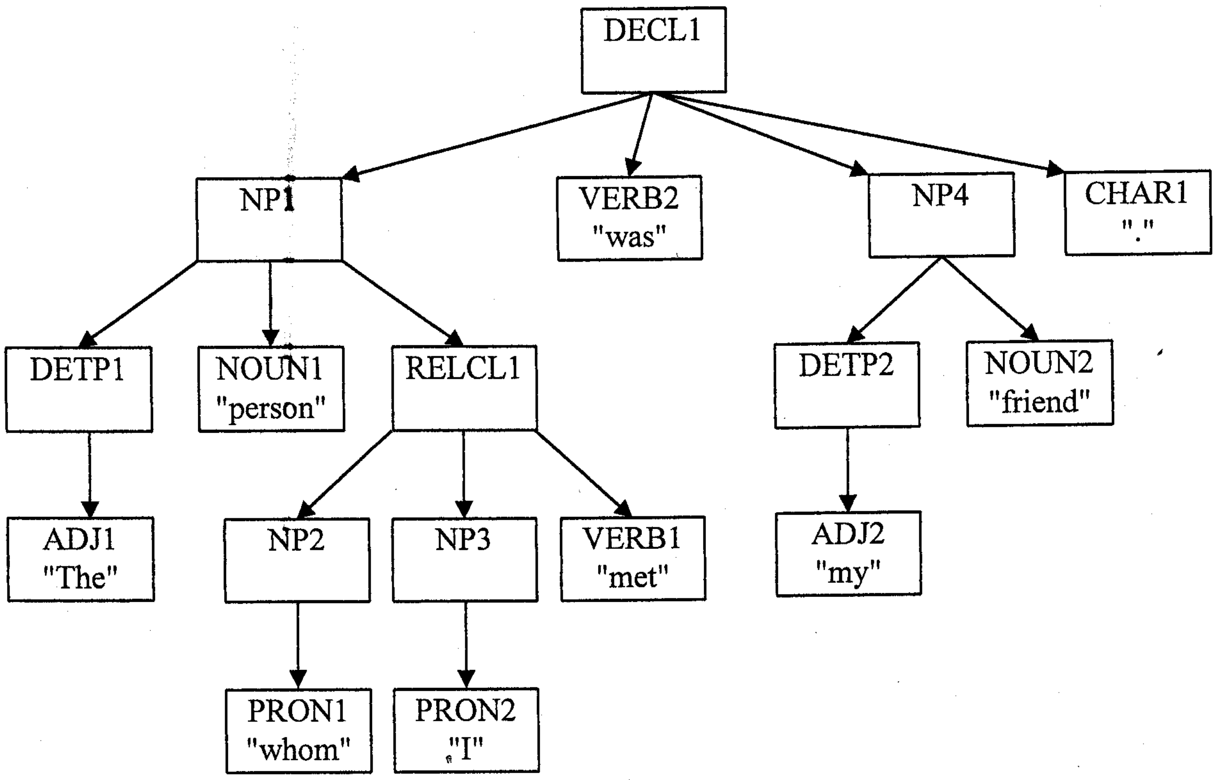


Fig. 51



Rule: LF_Dsub1 with node "be" labels link and creates another link

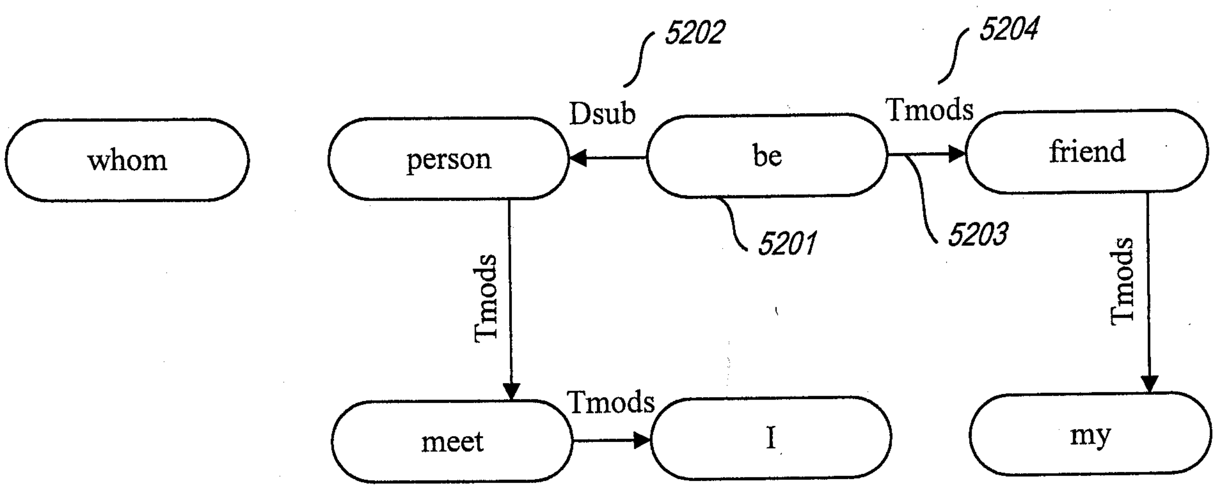
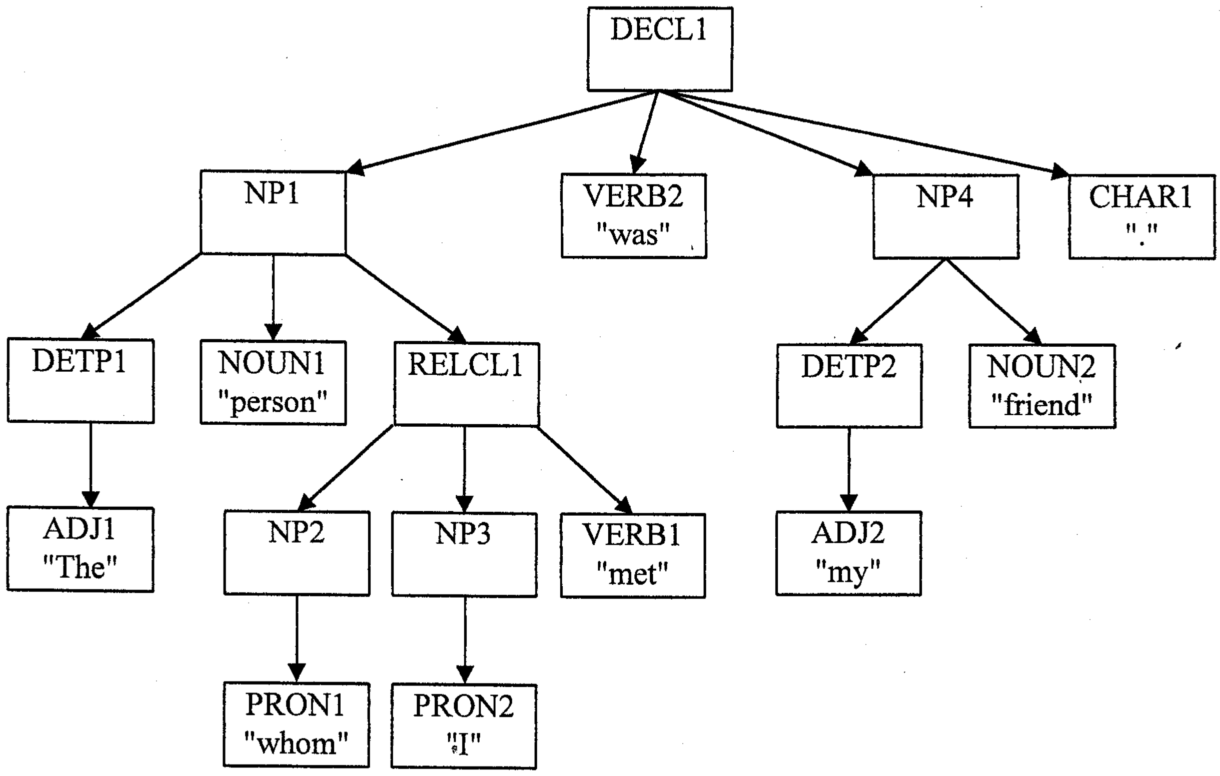


Fig. 52



Rule: LF_Dnom with node "be" labels link

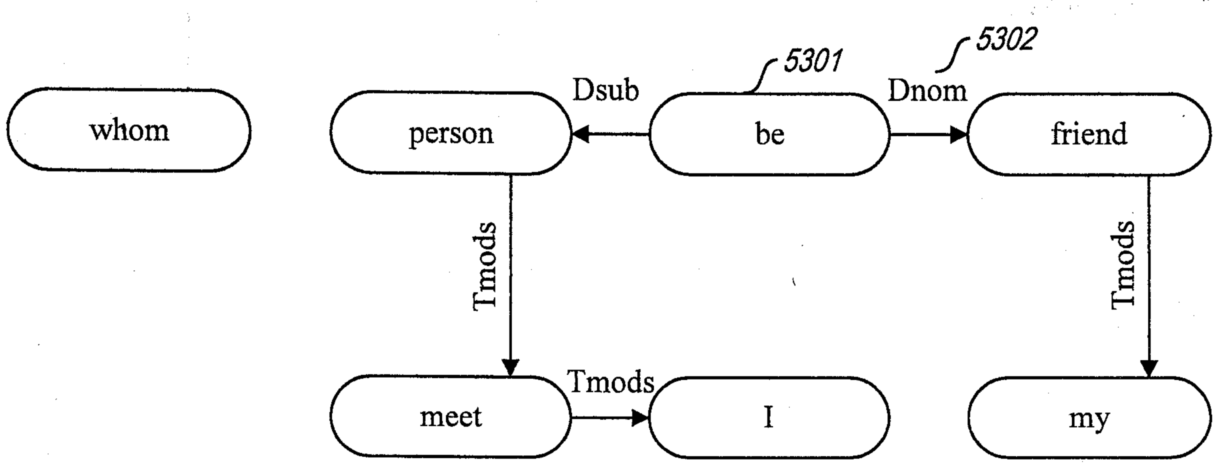
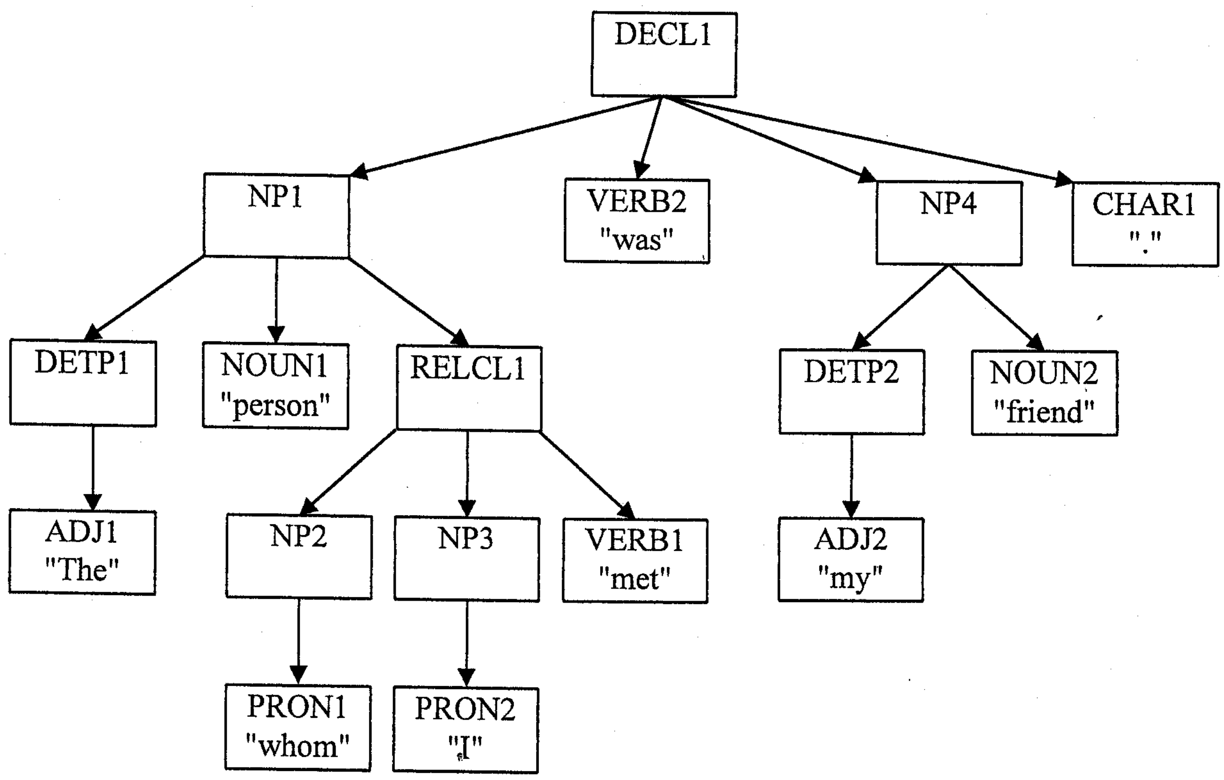


Fig. 53



Rule: LF_Props with node "person" labels link

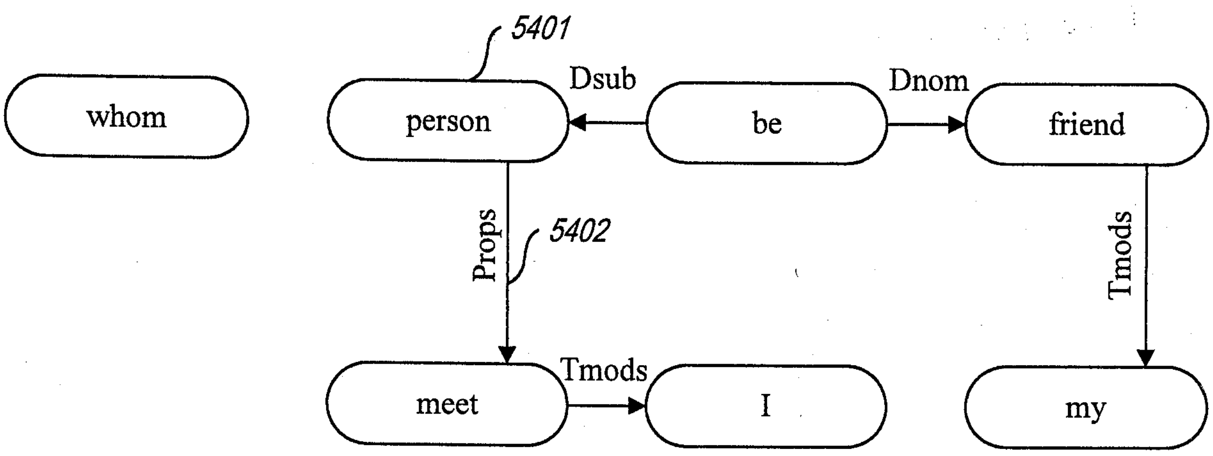
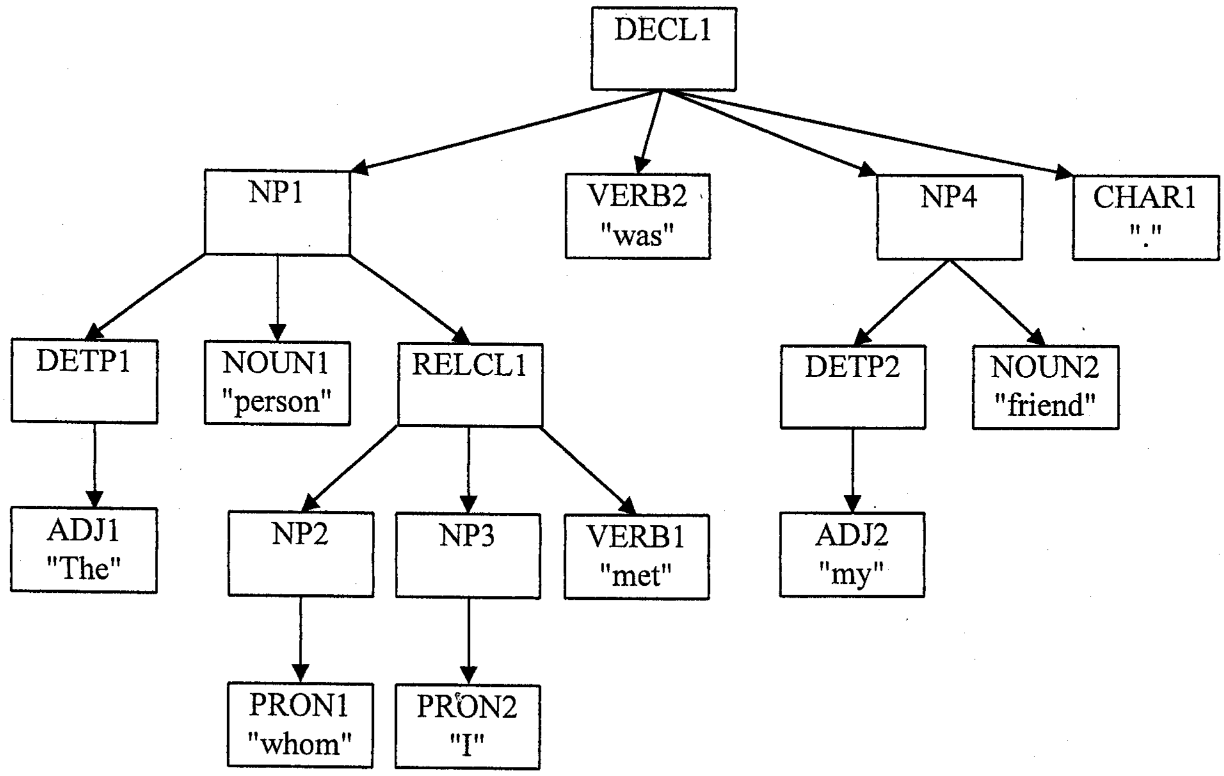


Fig. 54



Rule: LF_Dsub1 with node "meet" labels link

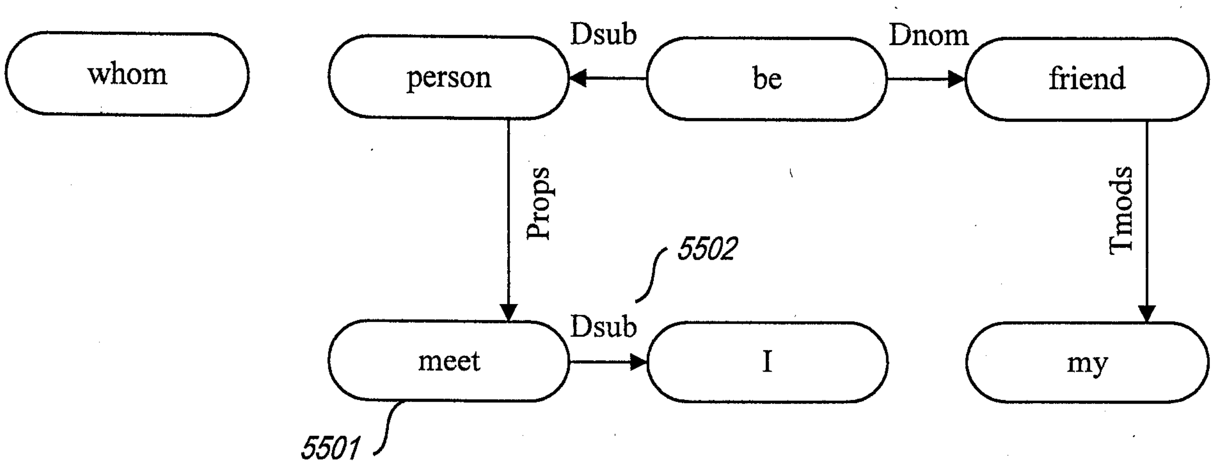
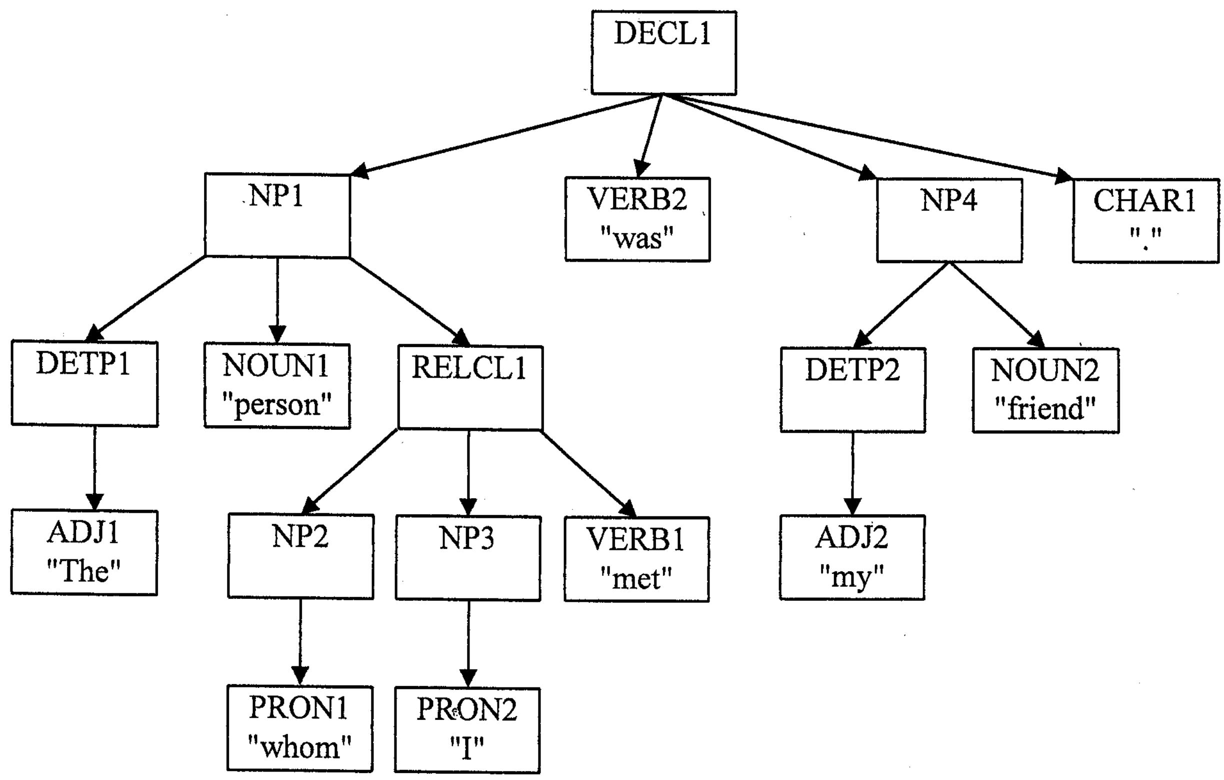


Fig. 55



Rule: LF_Dobj1 with node "meet" adds link and labels it

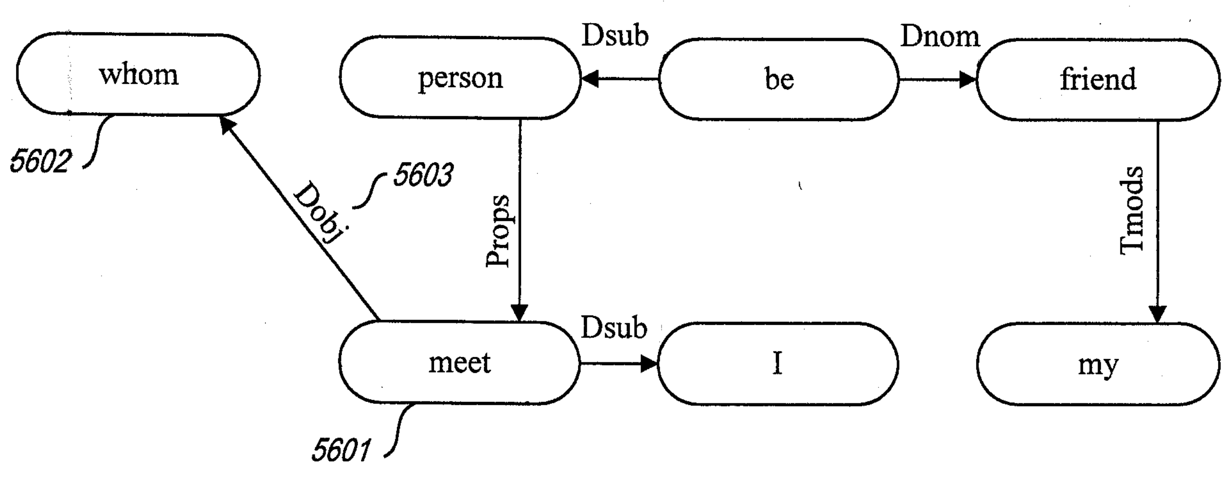
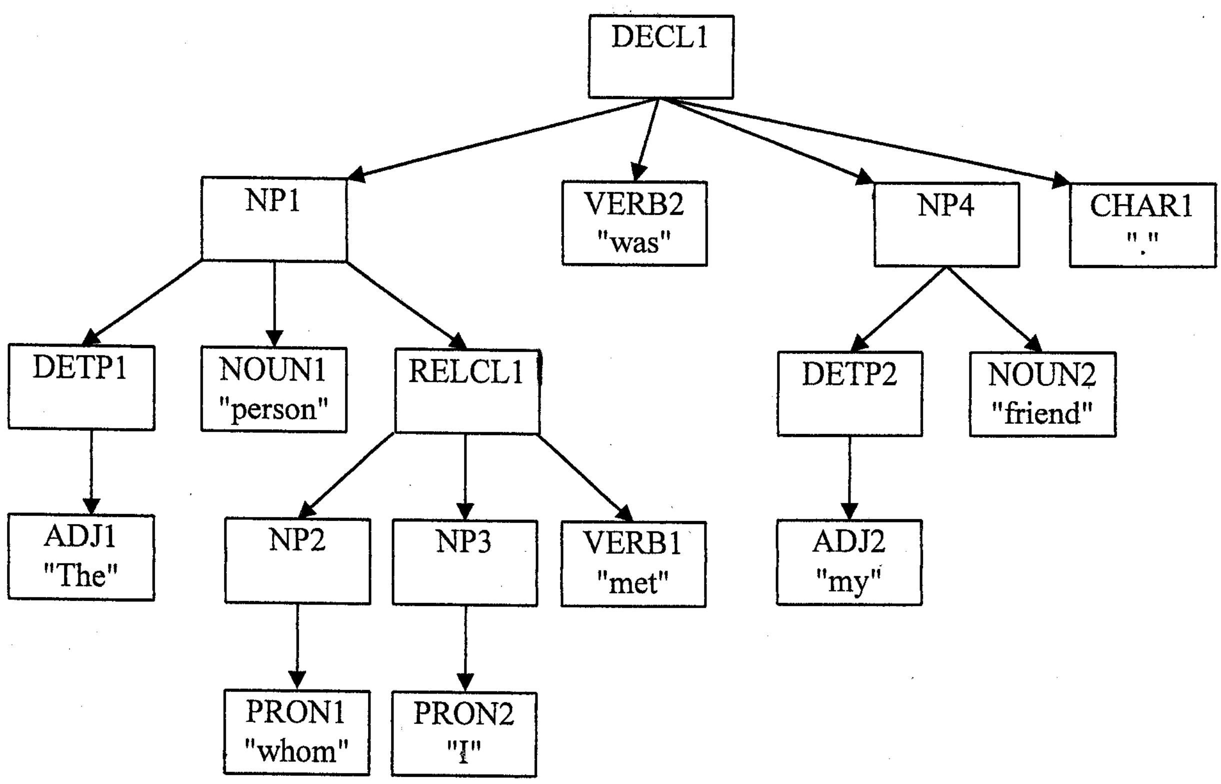


Fig. 56



Rule: LF_Ops with node "friend" labels link

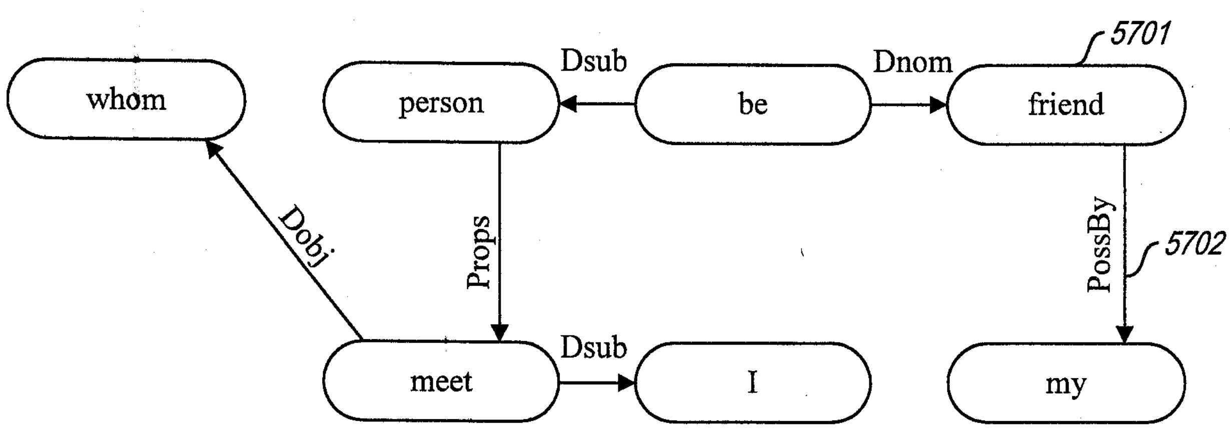
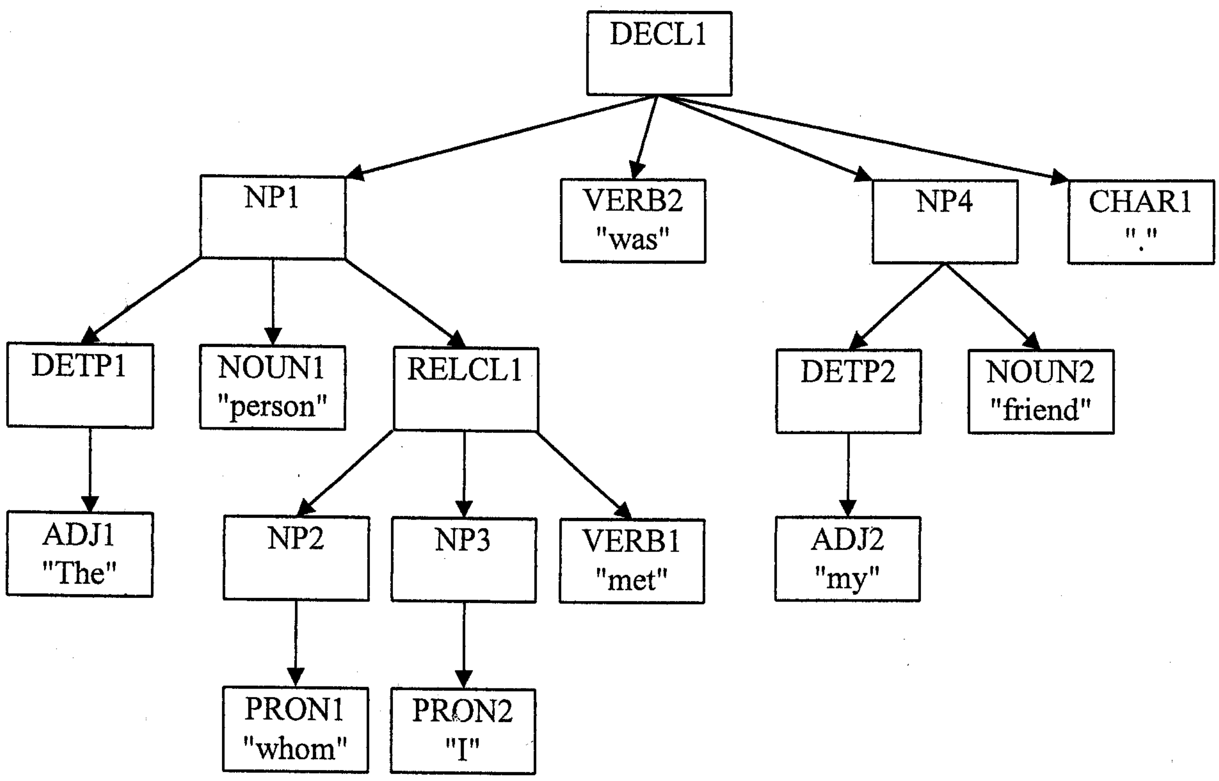


Fig. 57



Rule: PsLF_RelPro with node "whom" removes node and adds link

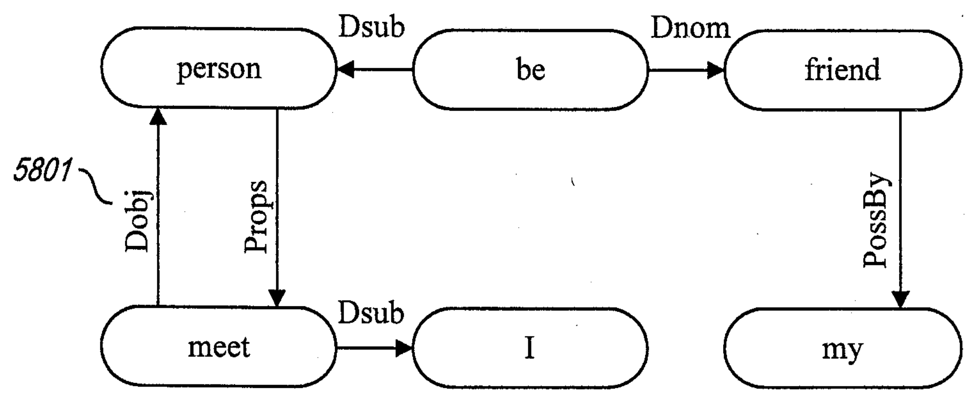
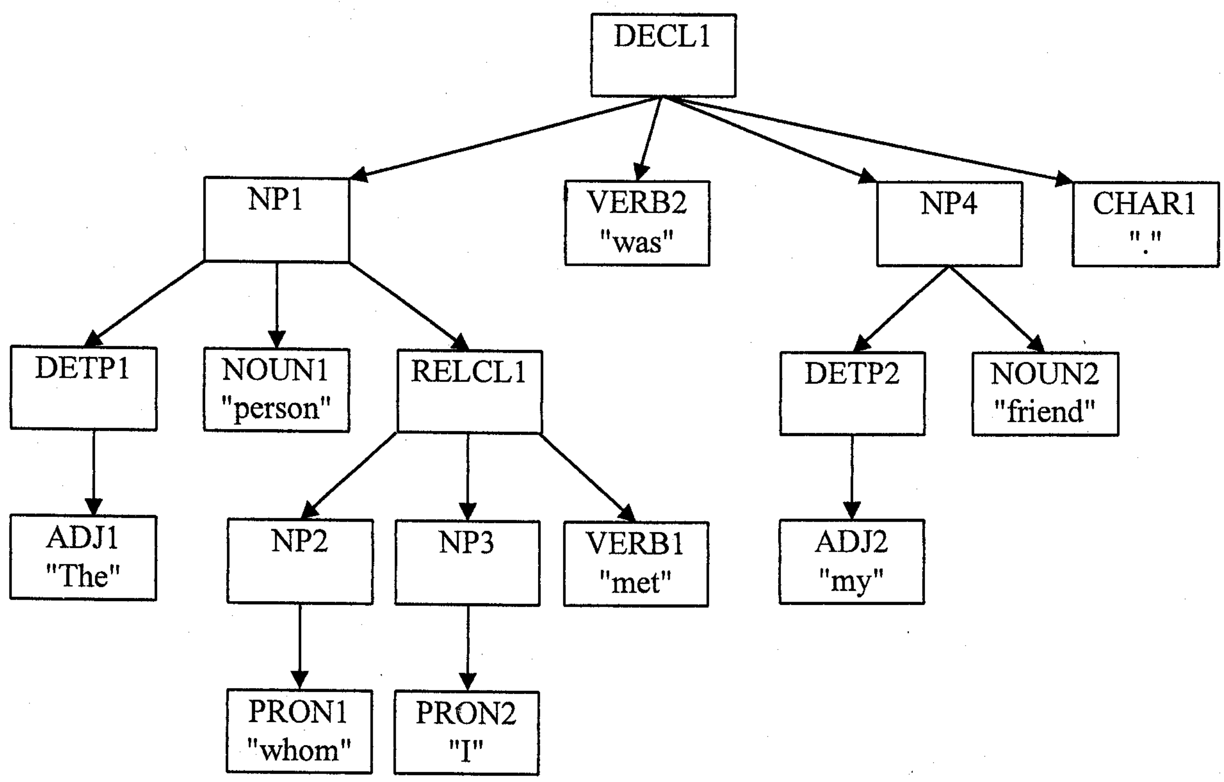


Fig. 58



Rule: PsLF_UnifyProns consolidates nodes "I" and "my" into a single node

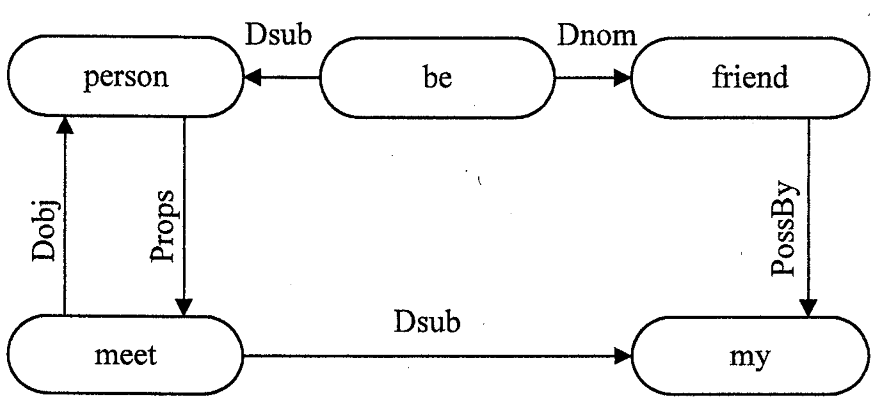


Fig. 59

PART B—ISSUE FEE TRANSMITTAL

Complete and mail this form, together with applicable fees, to: **Box ISSUE FEE**
Assistant Commissioner for Patents
Washington, D.C. 20231

L-S

B8

MAILING INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE. Blocks 1 through 4 should be completed where appropriate. All further correspondence including the Issue Fee Receipt, the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Legibly mark-up with any corrections or use Block 1)

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

OIPE
KM41/0427
JUL 23 1999
PATENT & TRADEMARK

Note: The certificate of mailing below can only be used for domestic mailings of the Issue Fee Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing.

Certificate of Mailing

I hereby certify that this Issue Fee Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Box Issue Fee address above on the date indicated below.

Bonita R. Gladden (Depositor's name)

Bonita R. Gladden (Signature)

July 20, 1999 (Date)

APPLICATION NO.	FILING DATE	TOTAL CLAIMS	EXAMINER AND GROUP ART UNIT		DATE MAILED
08/674,610	06/28/96	009	THOMAS, J	2747	04/27/99

First Named Applicant: HEIDORN, 35 USC 154(b) term ext. = 0 Days.

TITLE OF INVENTION: METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

ATTY'S DOCKET NO.	CLASS-SUBCLASS	BATCH NO.	APPLN. TYPE	SMALL ENTITY	FEE DUE	DATE DUE
2 661005.447	704-009.000	S42	UTILITY	NO	$1210.00	07/27/99

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363). Use of PTO form(s) and Customer Number are recommended, but not required.

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47) attached.

2. For printing on the patent front page, list (1) the names of up to 3 registered patent attorneys or agents OR, alternatively, (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Seed and Berry LLP
2 ____
3 ____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)
PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. Inclusion of assignee data is only appropiate when an assignment has been previously submitted to the PTO or is being submitted under separate cover. Completion of this form is NOT a subsititue for filing an assignment.

(A) NAME OF ASSIGNEE
MICROSOFT CORPORATION
(B) RESIDENCE: (CITY & STATE OR COUNTRY)
Redmond, Washington

Please check the appropriate assignee category indicated below (will not be printed on the patent)
☐ individual ☒ corporation or other private group entity ☐ government

4a. The following fees are enclosed (make check payable to Commissioner of Patents and Trademarks):
☒ Issue Fee
☒ Advance Order - # of Copies 14

4b. The following fees or deficiency in these fees should be charged to:
DEPOSIT ACCOUNT NUMBER 19-1090
(ENCLOSE AN EXTRA COPY OF THIS FORM)
☒ Issue Fee
☒ Advance Order - # of Copies ____

The COMMISSIONER OF PATENTS AND TRADEMARKS IS requested to apply the Issue Fee to the application identified above.

(Authorized Signature) [signature] (Date) 7/20/99

NOTE; The Issue Fee will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the Patent and Trademark Office.

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending on the needs of the individual case. Any comments on the amount of time required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND FEES AND THIS FORM TO: Box Issue Fee, Assistant Commissioner for Patents, Washington D.C. 20231

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

RECEIVED
JUL 28 1999
Publishing Division

07/26/1999 SSALEEK1 00000043 08674610
01 FC:142 1210.00 OP
02 FC:561 42.00 OP

TRANSMIT THIS FORM WITH FEE

PTOL-85B (REV.10-96) Approved for use through 06/30/99. OMB 0651-0033

Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

PART B—ISSUE FEE TRANSMITTA

Complete and mail this form, together with a :able fees, to: **Box ISSUE FEE**
Assistant Commissioner for Pat..its
Washington, D.C. 20231

MAILING INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE. Blocks 1 through 4 should be completed where appropriate. All further correspondence including the Issue Fee Receipt, the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Legibly mark-up with any corrections or use Block 1)

SEED AND BERRY
6300 COLUMBIA CENTER
SEATTLE WA 98104-7092

OIPE
QM41/0427 JC33
JUL 23 1999
PATENT & TRADEMARK OFFICE

Note: The certificate of mailing below can only be used for domestic mailings of the Issue Fee Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing.

Certificate of Mailing

I hereby certify that this Issue Fee Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Box Issue Fee address above on the date indicated below.

Bonita R. Gladden (Depositor's name)

Bonita R. Gladden (Signature)

July 20, 1999 (Date)

APPLICATION NO.	FILING DATE	TOTAL CLAIMS	EXAMINER AND GROUP ART UNIT		DATE MAILED
08/674,610	06/28/96	009	THOMAS, J	2747	04/27/9

First Named Applicant: HEIDORN, 35 USC 154(b) term ext. = 0 Days.

TITLE OF INVENTION: METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

ATTY'S DOCKET NO.	CLASS-SUBCLASS	BATCH NO.	APPLN. TYPE	SMALL ENTITY	FEE DUE	DATE DUE
2 661005.447	704-009.000	S42	UTILITY	NO	$1210.00	07/27/99

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363). Use of PTO form(s) and Customer Number are recommended, but not required.

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47) attached.

2. For printing on the patent front page, list (1) the names of up to 3 registered patent attorneys or agents OR, alternatively, (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Seed and Berry LLP
2 ____
3 ____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)
PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. Inclusion of assignee data is only appropiate when an assignment has been previously submitted to the PTO or is being submitted under separate cover. Completion of this form is NOT a subsitiute for filing an assignment.

(A) NAME OF ASSIGNEE
MICROSOFT CORPORATION
(B) RESIDENCE: (CITY & STATE OR COUNTRY)
Redmond, Washington

Please check the appropriate assignee category indicated below (will not be printed on the patent)
☐ individual ☒ corporation or other private group entity ☐ government

4a. The following fees are enclosed (make check payable to Commissioner of Patents and Trademarks):
☒ Issue Fee
☒ Advance Order - # of Copies 14

4b: The following fees or deficiency in these fees should be charged to:
DEPOSIT ACCOUNT NUMBER 19-1090
(ENCLOSE AN EXTRA COPY OF THIS FORM)
☒ Issue Fee
☒ Advance Order - # of Copies ____

The COMMISSIONER OF PATENTS AND TRADEMARKS IS requested to apply the Issue Fee to the application identified above.

(Authorized Signature) (Date) 7/20/99

NOTE; The Issue Fee will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the Patent and Trademark Office.

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending on the needs of the individual case. Any comments on the amount of time required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND FEES AND THIS FORM TO: Box Issue Fee, Assistant Commissioner for Patents, Washington D.C. 20231

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMIT THIS FORM WITH FEE

5966686 GP 2741

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant : George Heidorn et al.		
Serial No.: 08/674,610		
Filed : June 28, 1996	Group Art Unit: 2741	
For : METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES	Examiner: Harold A. Zintel	
Docket No.: M61.12-0199		

REVOCATION OF PRIOR POWERS OF ATTORNEY
AND POWER OF ATTORNEY

I HEREBY CERTIFY THAT THIS PAPER IS BEING SENT BY U.S. MAIL, FIRST CLASS, TO THE ASSISTANT COMMISSIONER FOR PATENTS, WASHINGTON, D.C. 20231, THIS 4th DAY OF February, ~~19~~ 2000

Joseph R. Kelly
PATENT ATTORNEY

Assistant Commissioner for Patents
Washington, D.C. 20231

The undersigned, authorized to act on behalf of Microsoft Corporation, the owner by assignment of the entire right, title and interest in and to the above-identified application, hereby revokes all previous powers of attorney and appoints the following attorneys and/or agents to prosecute this application and to transact all business in the U.S. Patent and Trademark Office connected therewith: Nickolas E. Westman, Reg. No. 20,147; Judson K. Champlin, Reg. No. 34,797; Joseph R. Kelly, Reg. No. 34,847; Steven M. Koehler, Reg. No. 36,188; David D. Brush, Reg. No. 34,557; John D. Veldhuis-Kroeze, Reg. No. 38,354; Deirdre Megley Kvale, Reg. No. 35,612; Theodore M. Magee, Reg. No. 39,758; Peter S. Dardi, Reg. No. 39,650; Christopher R. Christenson, Reg. No. 42,413; and John A. Wiberg, Reg. No. P-44,401.

Address all telephone calls to Joseph R. Kelly at telephone number (612) 334-3222.

Address all correspondence to Joseph R. Kelly, Westman, Champlin & Kelly, P.A., Suite 1600 - International Centre, 900 Second Avenue South, Minneapolis, Minnesota 55402-3319.

Respectfully submitted,

Date: 1/10/2000

By: Kate Cole

Title: Assistant Corporate Secretary

OIPE JC46
APR 1 0 2000
PATENT & TRADEMARK OFFICE

CofC
#17

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant	: George Heidorn et al.	
Applic No	: 08/674,610	
Filed	: June 28, 1996	Group Art Unit: 2741
For	: METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES	Examiner: Harold A. Zintel
Patent No.	: 5,966,686	Batch No: S42
Issued	: October 12, 1999	
Docket No.	: M61.12-0199	

REQUEST FOR CERTIFICATE OF CORRECTION

CERTIFICATE
APR 1 9 2000
OF CORRECTION

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

In conformity with the notice appearing in the May 6, 1969 Official Gazette, applicant hereby requests a Certificate of Correction in connection with the above-identified patent.

Form PTO-1050 entitled CERTIFICATE OF CORRECTION setting out the printer's errors has been completed and is enclosed. It is respectfully requested that the enclosed Certificate be approved and signed by an Attesting Officer, and that a copy be returned to applicant's attorney for attachment to the original Certificate of Letters Patent.

Respectfully submitted,

WESTMAN, CHAMPLIN & KELLY, P.A.

LW
Approval-In Part
OCT 2 5 2000

By: Joseph R. Kelly

Joseph R. Kelly, Reg. No. 34,847
Suite 1600 - International Centre
900 Second Avenue South
Minneapolis, Minnesota 55402-3319
Telephone: (612) 334-3222
Facsimile: (612) 334-3312

JRK:slg

Staple Here Only !

PRINTER'S TRIM LINE

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,966,686

DATED : October 12, 1999

INVENTOR(S) : George Heidorn et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Page 1 of 2

~~Specifications:~~

Column 1, line 42, a new paragraph should have been started with "Syntactic".

Column 9, line 64, "if" should be --it--.

No Column 11, line 1, delete "of".

Column 14, line 32, "DEPT2" should be --DETP2--.

Column 14, line 62, "LF__Dusbl" should be --LF__Dsub1--.

Column 15, line 6, "LF__Dusbl" should be --LF__Dsub1--.

Note 409 Column 15, line 23, "PsLF_UnifyPron" should be --PshF_UnifyProns--.

MAILING ADDRESS OF SENDER: Joseph R. Kelly
WESTMAN, CHAMPLIN & KELLY, P.A.
Suite 1600 - International Centre
900 Second Avenue South
Minneapolis, MN 55402-3319

PATENT NO. 5,966,686

No. of add'l copies @ 50¢ per page

FORM PTO 1050 (Rev. 2-93)

Staple Here Only !

PRINTER'S TRIM LINE

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO. : 5,966,686
DATED : October 12, 1999
INVENTOR(S) : George Heidorn et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

~~Page 2 of 2~~

~~Claims:~~

Claim 1, Column 15, line 41, "all" should be --an--.

Claim 4, Column 16, line 11, "The" should be --the--.

MAILING ADDRESS OF SENDER: WESTMAN, CHAMPLIN & KELLY, P.A.
Suite 1600 - International Centre
900 Second Avenue South
Minneapolis, MN 55402-3319

PATENT NO. 5,966,686

No. of add'l copies @ 50¢ per page

FORM PTO 1050 (Rev. 2-93)

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
ASSISTANT SECRETARY OF COMMERCE AND
COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

#18

MAILING DATE 10-25-00	
PATENT NO. 5,966,686	PATENT DATE 10/12/99
INVENTORS: George Heidorn, et al.	
ATTORNEY DOCKET NO: 661005447	

Joseph R. Kelly
Westman, Champlin & Kelly, P.A.
Suite 1600 - International Centre
900 Second Avenue South
Minneapolis, MN 55402-3319

NOTIFICATION REGARDING REQUEST FOR CERTIFICATE OF CORRECTION

The Certificate of Correction requested in the patent identified above has been APPROVED with the exception indicated below. The remaining errors will be corrected as requested. The Certificate, so modified, will be issued on 01-02-2001.

[ ] **A. THE CHANGES BELOW CANNOT BE INCLUDED IN THE CERTIFICATE SINCE THE REQUEST WAS FILED UNDER RULE 322:**

[X] 1. Column 11, line 1, is printed in accordance with the record.

[ ] (a) The change referred to was initialed and dated by applicant before execution of the application papers.

[ ] 2. In column _, line , the errors resulted from applicant's failure to comply with Rule 121(a), in that the precise point of entry of the amendment was omitted.

[ ] 3. In column ________, line ____________, the alleged error is due to applicant's failure to comply with Rule 121(b), wherein provision is made for use of brackets, instead of parentheses, to cancel subject matter and for the use of interlineations to indicate new subject matter.

[ ] 4. Omission of the priority data from the patent resulted from applicant's failure to fully comply with 35 U.S.C. 119, in that:

[ ] (a) The priority data was omitted from the oath, or declaration

[ ] (b) The claim for priority was not included in the application papers.

[ ] (c) The certified copy of the foreign application was not filed.

[ ] 5. Since, the inventor name(s) is/are printed in accordance with the type written signature, no correction is in order here, unless a petition is granted (See Petition filing information below).

[ ] 6. The assignment data is printed in the patent in accordance with PTO-85b, submitted by applicant at time of payment of the base issue fee, no correction is in order here, unless a petition is granted (See Petition filing information below).

Any petition should be directed to the attention of the Assistant Commissioner for Patents, using the following mailing address or FAX number.

By Mail: Commissioner of Patents and Trademarks
Box DAC
Washington, D.C. 20231

OR

By FAX: (703) 308-6916
Attn.: Office of Petitions

[ ] 7. In column ___, line ___, the error arose because Rule 1.52(a) or 1.52(b) was not complied with. Consequently, words on top of certain pages were obliterated or not legible causing the Office to provide what appeared to be the proper words.

[ ] **B. THE REQUEST HAS BEEN CHANGED AS SHOWN BELOW TO COMPLY WITH THE RECORD:**

[ ] 1. The error complained of in claim , column , line , occurred in claim column , line , where the change will be made.

[X] 2. The change requested in column 15, line 23, has been modified by changing the correction to read:

-- PsLF_UnifyProns --

Form PTOL-404 (rev. 5/94)

☐ THE FOLLOWING CORRECTION(S) CANNOT BE INCLUDED IN THE CERTIFICATE FOR THE REASONS GIVEN BELOW:

1. The words , purported to be , cannot be found in the printed patent.

☐ 2. The alleged error on the , is an editing change made in accordance with the style of the Invention Patent Manual.

☐ 3. In column , line , alleged error is in fact a change made by the examiner and considered to be in accordance with the permissible amendments enumerated in M.P.E.P. 1302.04.

☐ 4. In the title, it is the practice to exclude words such as "Improvements in", "New", "A", "Novel", etc., from the printed patent.

☐ 5. Comparison of the patent in columns, lines , with the corresponding location in the application file reveals that there is no discrepancy.

☐ 6. The numbering of the claims and their dependency in the printed patent is in accordance with the renumbering of dependent claims by the examiner as described in M.P.E.P.608.01(n).

☐ 7. The alleged error in column__________ , line___________ , is a change made in an Examiner's Amendment at time of allowance. Since no error is involved and since applicant filed no objection prior to payment of the base issue fee, the requested change will not be included in the Certificate.

☐ 8. The error complained of on the title page item , cannot be corrected since: the initial citation is not in conformance with MPEP 609.

D. ADDITIONAL CORRECTION:
E. OTHER (Fee not enclosed):

FOR ADDITIONAL INFORMATION REGARDING THIS NOTIFICATION PLEASE CONTACT:

Valerie Jackson
Certificates of Correction
(703) 305-8347

WITHIN 4 WEEKS FROM MAILING DATE OF THIS NOTIFICATION

L. Watson
Supervisor, Certificates of Correction Branch-

This decision is rendered pursuant to authority delegated by the Solicitor under authority delegated to him by the Commissioner of Patents and Trademarks.

```
=> d his full

     (FILE 'USPAT' ENTERED AT 11:53:40 ON 19 APR 1999)
                ACT HAL/L
               ---------
L1   (       3688)SEA SYNTAX?
L2   (       1937)SEA SYNTACT?
L3   (       9532)SEA PARS?
L4   (      12711)SEA TOKEN?
L5   (      38168)SEA TREE#
L6   (         99)SEA TRIE
L7   (      12602)SEA TRIES
L8   (     180778)SEA ANALYZ?
L9   (     306620)SEA ANALYS?
L10  (       1022)SEA (SYNTAX? OR SYNTACT?)(10A)(PARS? OR TOKEN? OR TREE# OR
 TR
                IE OR TRIES OR ANALYZ? OR ANALYS?)
L11  (       2811)SEA SEMANTIC?
L12  (        509)SEA (SYNTAX? OR SYNTACT?)(10A)(PARS? OR TOKEN? OR TREE# OR
 TR
                IE OR TRIES OR ANALYZ? OR ANALYS?) AND SEMANTIC?
L13  (      48520)SEA (DATA OR INFORMATION? OR DATUM)(5A)(STRUCTUR? OR SCHEM
?)

L14  (          0)SEA LOGIC? FORM GRAPH#
L15  (     171012)SEA GRAPH#
L16  (     119956)SEA GRAPHIC?
L17  (     998702)SEA REPRESENT?
L18  (     158684)SEA CHART#
L19  (     735913)SEA DIAGRAM?
L20  (       1198)SEA VERB#
L21  (       3052)SEA TENSE#
L22  (       5565)SEA SENTENCE#
L23  (     345370)SEA WORD#
L24  (     138791)SEA MEANING#
L25  (      90564)SEA (GRAPH#  OR GRAPHIC? OR REPRESENT? OR CHART# OR DIAGRA
M?)
                (P)(VERB# OR TENSE# OR SENTENCE# OR WORD# OR MEANING#)
L26  (        222)SEA L12 AND L13 AND L25
               ---------
L27           346 SEA L25(P)(L11)
L28           363 SEA L11(10A)((L8 OR L9))
L29           106 SEA L26 AND L27
L30           117 SEA L26 AND L28
L31        311366 SEA ((L15 OR L16 OR L17 OR L18 OR L19))(15A)(LOGIC? OR FOR
M#
                OR FORMAT# OR FORMATION? OR CAST# OR CONFIGUR? OR PATTERN#
 OR
                 SHAPE#)
L32           127 SEA ((L29 OR L30)) AND L31
L33           132 SEA L10 AND L13 AND L31 AND L28
L34           118 SEA L33 NOT FD>=19960628
L35  (       4856)SEA 704*?/CCLS
L36  (          0)SEA 707*?/CCL
L37  (      17185)SEA 434*?/CCLS
L38  (      38553)SEA 364*?/CCLS
L39  (       3164)SEA 706*?/CCLS
L40  (      14770)SEA 395*?/CCLS
```

```
L41 (     17185)SEA 434*?/CCLS
L42        69534 SEA ·    *?/CCLS OR 707*?/CCL OR 434*?  . S OR 364*?/CCLS O
R 7
               06*?/CCLS OR 395*?/CCLS OR 434*?/CCLS)
L43          105 SEA L34 AND L42
L44 (       9532)SEA PARS?
L45 (      12711)SEA TOKEN?
L46 (      38168)SEA TREE#
L47 (         99)SEA TRIE
L48 (      12602)SEA TRIES
L49 (      65441)SEA NODE#
L50 (       4050)SEA NODAL?
L51 (      70625)SEA LEAF
L52 (     149214)SEA LEAVES
L53 (     228459)SEA BRANCH?
L54 (      64815)SEA ROOT#
L55         1143 SEA (PARS? OR TOKEN?)(5A)(TREE# OR TRIE OR TRIES OR NODE#
OR
               NODAL? OR LEAF OR LEAVES OR BRANCH? OR ROOT#)
L56           42 SEA L43 AND L55
L57         5854 SEA (SEPARATE? OR DISTINCT? OR INDEPENDEN?)(P)L13
L58           12 SEA L56 AND L57

FILE USPAT
  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
  *               W E L C O M E   T O   T H E                   *
  *          U. S.   P A T E N T   T E X T   F I L E            *
  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
      (FILE 'USPAT' ENTERED AT 11:53:40 ON 19 APR 1999)
                     ACT HAL/L
                    ---------
L1   (       3688)SEA SYNTAX?
L2   (       1937)SEA SYNTACT?
L3   (       9532)SEA PARS?
L4   (      12711)SEA TOKEN?
L5   (      38168)SEA TREE#
L6   (         99)SEA TRIE
L7   (      12602)SEA TRIES
L8   (     180778)SEA ANALYZ?
L9   (     306620)SEA ANALYS?
L10  (       1022)SEA (SYNTAX? OR SYNTACT?)(10A)(PARS? OR TOKEN? OR TREE# OR
 TR
                     IE OR TRIES OR ANALYZ? OR ANALYS?)
L11  (       2811)SEA SEMANTIC?
L12  (        509)SEA (SYNTAX? OR SYNTACT?)(10A)(PARS? OR TOKEN? OR TREE# OR
 TR
                     IE OR TRIES OR ANALYZ? OR ANALYS?) AND SEMANTIC?
L13  (      48520)SEA (DATA OR INFORMATION? OR DATUM)(5A)(STRUCTUR? OR SCHEM
?)

L14  (          0)SEA LOGIC? FORM GRAPH#
L15  (     171012)SEA GRAPH#
L16  (     119956)SEA GRAPHIC?
L17  (     998702)SEA REPRESENT?
L18  (     158684)SEA CHART#
L19  (     735913)SEA DIAGRAM?
L20  (       1198)SEA VERB#
L21  (       3052)SEA TENSE#
L22  (       5565)SEA SENTENCE#
L23  (     345370)SEA WORD#
L24  (     138791)SEA MEANING#
L25  (      90564)SEA (GRAPH#  OR GRAPHIC? OR REPRESENT? OR CHART# OR DIAGRA
M?)
                     (P)(VERB# OR TENSE# OR SENTENCE# OR WORD# OR MEANING#)
L26  (        222)SEA L12 AND L13 AND L25
                    ---------
L27           346 SEA L25(P)(L11)
L28           363 SEA L11(10A)((L8 OR L9))
L29           106 SEA L26 AND L27
L30           117 SEA L26 AND L28
L31        311366 SEA ((L15 OR L16 OR L17 OR L18 OR L19))(15A)(LOGIC? OR FOR
M#
                    OR FORMAT# OR FORMATION? OR CAST# OR CONFIGUR? OR PATTERN#
 OR
                     SHAPE#)
L32           127 SEA ((L29 OR L30)) AND L31
L33           132 SEA L10 AND L13 AND L31 AND L28
L34           118 SEA L33 NOT FD>=19960628
L35  (       4856)SEA 704*?/CCLS
L36  (          0)SEA 707*?/CCL
L37  (      17185)SEA 434*?/CCLS
L38  (      38553)SEA 364*?/CCLS
L39  (       3164)SEA 706*?/CCLS
L40  (      14770)SEA 395*?/CCLS
L41  (      17185)SEA 434*?/CCLS
```

```
L42        69534 SEA '^^4*?/CCLS OR 707*?/CCL OR 434*?/^^LS OR 364*?/CCLS O
R 7
                 06*?/CCLS OR 395*?/CCLS OR 434*?/CCLS)
L43          105 SEA L34 AND L42
L44 (       9532)SEA PARS?
L45 (      12711)SEA TOKEN?
L46 (      38168)SEA TREE#
L47 (         99)SEA TRIE
L48 (      12602)SEA TRIES
L49 (      65441)SEA NODE#
L50 (       4050)SEA NODAL?
L51 (      70625)SEA LEAF
L52 (     149214)SEA LEAVES
L53 (     228459)SEA BRANCH?
L54 (      64815)SEA ROOT#
L55         1143 SEA (PARS? OR TOKEN?)(5A)(TREE# OR TRIE OR TRIES OR NODE#
OR
                 NODAL? OR LEAF OR LEAVES OR BRANCH? OR ROOT#)
L56           42 SEA L43 AND L55
L57         5854 SEA (SEPARATE? OR DISTINCT? OR INDEPENDEN?)(P)L13
L58           12 SEA L56 AND L57
                 SAV HAL/L L1-L58
                 E DAMERAU?/IN
                 E DAMERA?/IN
L59            7 SEA ("DAMERAU, FREDERICK J"/IN OR "DAMERAU, HERBERT R"/IN
OR
                 "DAMERAU, WERNER"/IN)
                 E DISAMBIG?/TI
L60            8 SEA DISAMBIG?/TI
                 E 5258909/UREF
L61           10 SEA 5258909/UREF
                 E THOMAS, JO?/XA
                 E THOMAS, JO?/XP
L62            0 SEA E52-E53-E64
L63          117 SEA "THOMAS, JOE"/XA OR "THOMAS, JOSEPH"/XA OR "THOMAS, JO
SPE
                 H"/XA OR "THOMAS, JOSEPH"/XP
L64            1 SEA L61 AND L63
L65        25886 SEA SKELETON? OR SKELETAL?
L66            1 SEA L58 AND L65


     FILE USPAT
       * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
       *                W E L C O M E   T O   T H E                   *
       *          U. S.   P A T E N T   T E X T   F I L E             *
       * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
=> d his full

     (FILE 'USPAT' ENTERED AT 18:32:24 ON 15 APR 1999)
                DEL HIS
L1  (      3688)SEA SYNTAX?
L2  (      1937)SEA SYNTACT?
L3  (      9532)SEA PARS?
L4  (     12711)SEA TOKEN?
L5  (     38168)SEA TREE#
L6  (        99)SEA TRIE
L7  (     12602)SEA TRIES
L8  (    180778)SEA ANALYZ?
L9  (    306620)SEA ANALYS?
L10 (      1022)SEA (SYNTAX? OR SYNTACT?)(10A)(PARS? OR TOKEN? OR TREE# OR
 TR
                IE OR TRIES OR ANALYZ? OR ANALYS?)
L11 (      2811)SEA SEMANTIC?
L12         509 SEA (SYNTAX? OR SYNTACT?)(10A)(PARS? OR TOKEN? OR TREE# OR
 TR
                IE OR TRIES OR ANALYZ? OR ANALYS?) AND SEMANTIC?
L13       48520 SEA (DATA OR INFORMATION? OR DATUM)(5A)(STRUCTUR? OR SCHEM
?)

L14           0 SEA LOGIC? FORM GRAPH#
L15 (    171012)SEA GRAPH#
L16 (    119956)SEA GRAPHIC?
L17 (    998702)SEA REPRESENT?
L18 (    158684)SEA CHART#
L19 (    735913)SEA DIAGRAM?
L20 (      1198)SEA VERB#
L21 (      3052)SEA TENSE#
L22 (      5565)SEA SENTENCE#
L23 (    345370)SEA WORD#
L24 (    138791)SEA MEANING#
L25       90564 SEA (GRAPH#  OR GRAPHIC? OR REPRESENT? OR CHART# OR DIAGRA
M?)
                (P)(VERB# OR TENSE# OR SENTENCE# OR WORD# OR MEANING#)
L26         222 SEA L12 AND L13 AND L25
                DEL CORNEA?/L
                SAV HAL/L L1-L26
```

STAPLE AREA

PATENT NUMBER	ORIGINAL CLASSIFICATION	
	CLASS 704	SUBCLASS 9

APPLICATION SERIAL NUMBER
08/674,610

APPLICANT'S NAME (PLEASE PRINT)
HEIDORN, et al.

IF REISSUE, ORIGINAL PATENT NUMBER

CROSS REFERENCE(S)

CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)			
707	104			

INTERNATIONAL CLASSIFICATION

G	0	6	F		17 / 27
					/
					/
					/

GROUP ART UNIT 2747

~~ASSISTANT EXAMINER (PLEASE STAMP OR PRINT FULL NAME)~~

PRIMARY EXAMINER (PLEASE STAMP OR PRINT FULL NAME)
JOSEPH THOMAS
PRIMARY EXAMINER

PTO 270
(REV. 5-91)

ISSUE CLASSIFICATION SLIP

U.S. DEPARTMENT OF COMMERCE
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION FEE DETERMINATION RECORD Effective October 1, 1995	Application or Docket Number 674610

CLAIMS AS FILED - PART I

FOR	NUMBER FILED (Column 1)		NUMBER EXTRA (Column 2)	SMALL ENTITY RATE	SMALL ENTITY FEE	OR	OTHER THAN SMALL ENTITY RATE	OTHER THAN SMALL ENTITY FEE
BASIC FEE					375.00	OR		750.00
TOTAL CLAIMS	36	minus 20 =	* 16	x$11=		OR	x$22=	352
INDEPENDENT CLAIMS	7	minus 3 =	* 4	x39=		OR	x78=	312
MULTIPLE DEPENDENT CLAIM PRESENT				+125=		OR	+250=	
				TOTAL		OR	TOTAL	1414

* If the difference in column 1 is less than zero, enter "0" in column 2

CLAIMS AS AMENDED - PART II

AMENDMENT A

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	* 64	Minus	** 36	= 28	x$11=		OR	x$22=	616
Independent	* 11	Minus	*** ~~3~~ 7	= 4	x39=		OR	x78=	328
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+125=		OR	+250=	~~54~~ 110.00
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	1,054

CPA

AMENDMENT B

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	* 9	Minus	** 64	= Ø	x$11=		OR	x$22=	
Independent	* 4	Minus	*** 11	= Ø	x39=		OR	x78=	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+125=		OR	+250=	
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT C

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	*	Minus	**	=	x$11=		OR	x$22=	
Independent	*	Minus	***	=	x39=		OR	x78=	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+125=		OR	+250=	
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

FORM PTO-875
(Rev. 10/95)

Patent and Trademark Office, U.S. DEPARTMENT OF COMMERCE

Form PTO 1130
(REV 2/94)

U.S. DEPARTMENT OF COMMERCE
Patent and Trademark Office

PACE DATA ENTRY CODING SHEET

1ST EXAMINER	Nhu Thuy	DATE	8/20
2ND EXAMINER	M. Switty	DATE	11-19-96

APPLICATION NUMBER	TYPE APPL	FILING DATE (MONTH DAY YEAR)	SPECIAL HANDLING	GROUP ART UNIT	CLASS	SHEETS OF DRAWING
08/674610	1	06 28 96	0	2412	395	069

TOTAL CLAIMS	INDEPENDENT CLAIMS	SMALL ENTITY?	FILING FEE	FOREIGN LICENSE	ATTORNEY DOCKET NUMBER
036	007	ø	1544	Y	661005.447

CONTINUITY DATA

CONT CODE	STATUS CODE	PARENT APPLICATION SERIAL NUMBER	PCT APPLICATION SERIAL NUMBER	PARENT PATENT NUMBER	PARENT FILING DATE (MONTH DAY YEAR)
			PCT/ /		
			PCT/ /		
			PCT/ /		
			PCT/ /		
			PCT/ /		

PCT/FOREIGN APPLICATION DATA

FOREIGN PRIORITY CLAIMED	COUNTRY CODE	PCT/FOREIGN APPLICATION SERIAL NUMBER	FOREIGN FILING DATE (MONTH DAY YEAR)

☆U.S.G.P.O.: 1994 - 384-021

Staple Issue Slip Here

POSITION		ID NO.	DATE
CLASSIFIER		7	8-13-96
EXAMINER		[illegible]	5/20
TYPIST		1440 (5)	11/20/96
VERIFIER			
CORPS CORR.			
SPEC. HAND	422		11-19-96
FILE MAINT.			
DRAFTING			

INDEX OF CLAIMS

Claim Final	Claim Original	3/25/98	9/10/98
	1	✓	✓
	2		
	3		
	4		
	5		
	6		
	7		
	8		
	9		
	10		
	11		
	12		
	13		
	14		
	15		
	16		
	17		
	18		
	19		
	20		
	21		
	22		
	23		
	24		
	25		
	26		
	27		
	28		
	29		
	30		
	31		
	32		
	33		
	34		
	35		
	36	✓	
	37		
	38		
	39		
	40		
	41		
	42		
	43		
	44		
	45		
	46		
	47		
	48		
	49		
	50		✓

Claim Final	Claim Original	1/10/98	4/99
	51	✓	
	52		
	53		
	54		
	55		
	56		
	57		
	58		
	59		
	60		
	61		
	62		
	63		
	64	✓	
1	65		=
2	66		=
3	67		=
4	68		=
5	69		=
6	70		=
7	71		=
8	72		=
9	73		=
	74		
	75		
	76		
	77		
	78		
	79		
	80		
	81		
	82		
	83		
	84		
	85		
	86		
	87		
	88		
	89		
	90		
	91		
	92		
	93		
	94		
	95		
	96		
	97		
	98		
	99		
	100		

SYMBOLS

- ✓ .... Rejected
- = ... Allowed
- - (Through numeral) Canceled
- \+ .... Restricted
- N .... Non-elected
- I .... Interference
- A .... Appeal
- O .... Objected

(LEFT INSIDE)

[redacted]

SEARCHED

Class	Sub.	Date	Exmr.
704	9 8	3/20/98	[illegible]
395	12		
updated	above	4/15/99	J.T.
704	1 10	4/19/99	J.T.
707	100 101 102 104	4/19/99	J.T.

INTERFERENCE SEARCHED

Class	Sub.	Date	Exmr.
704	9	4/19/99	J.T.
707	104	4/19/99	J.T.

SEARCH NOTES

	Date	Exmr.
MAYA Search	3/18/98	[illegible]
Talked to Joe Thimas about search	3/18/98	[illegible]
APS Search	~~4~~ 4/15/99 4/19/99	J.T.

(RIGHT OUTSIDE)

US005966686A

United States Patent [19]

Heidorn et al.

[11] **Patent Number: 5,966,686**

[45] **Date of Patent: *Oct. 12, 1999**

[54] **METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES**

[75] Inventors: **George Heidorn**; **Karen Jensen**, both of Bellevue, Wash.

[73] Assignee: **Microsoft Corporation**, Redmond, Wash.

[*] Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

[21] Appl. No.: **08/674,610**

[22] Filed: **Jun. 28, 1996**

[51] **Int. Cl.**[6] **G06F 17/27**

[52] **U.S. Cl.** **704/9**; 707/104

[58] **Field of Search** 704/9, 8, 1, 10; 395/12; 707/100, 101, 102, 104

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,111,398	5/1992	Nunberg et al.	704/9
5,146,406	9/1992	Jensen	704/9
5,386,556	1/1995	Hedin et al.	707/4
5,406,480	4/1995	Kanno	704/10
5,424,947	6/1995	Nagao et al.	704/9

FOREIGN PATENT DOCUMENTS

0413132A2	7/1990	European Pat. Off. .
0413132A3	7/1990	European Pat. Off. .

OTHER PUBLICATIONS

Geetha,T. V. and Subramanian, R.K., "Natural Language Representation—a Connectionist Approach", Computer and Communication, New Delhi, Aug. 28, 1991, vol. 3, pp. 294–298.

Isahara, Hitoshi and Ishizaki, Shun, "Context Analysis System for Japanese Text", 11th International Conference on Computational Linguistics. Proceedings of Coling '86, Bonn, West Germany, Aug. 25–29, 1986, pp. 244–246.

Winograd, Terry,"Computer Software for Working with Language", *Scientific American*, Sep. 1984, New York, U.S.A., vol. 251, No. 3, pp. 90–101.

Jensen, Karen et al., *Natural Language Processing: The PLNLP Approcach,* Kluwer Academic Publishers, Boston, 1993.

Garside, Roger et al., *The Computational Analysis of English: A Corpus–Based Approach,* Longman, pp. 97–109, 1987.

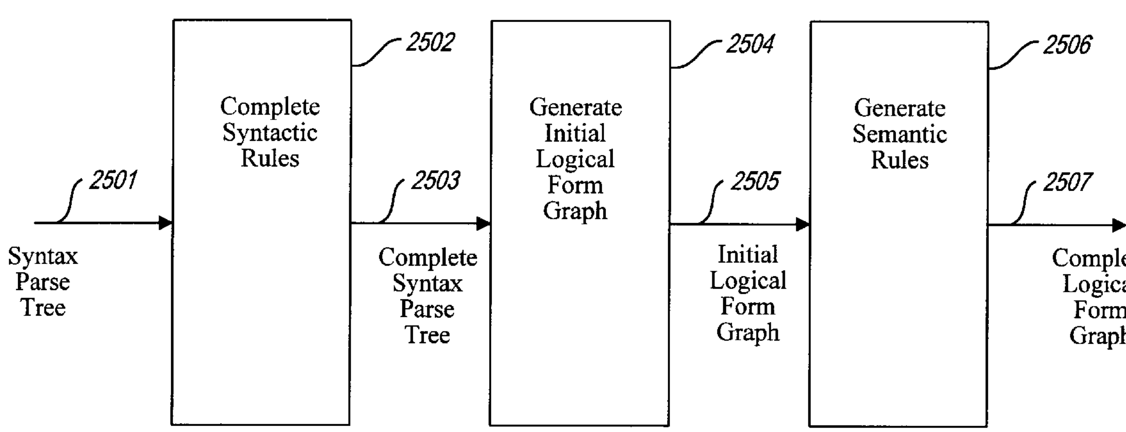
Primary Examiner—Joseph Thomas
Attorney, Agent, or Firm—Seed and Berry LLP

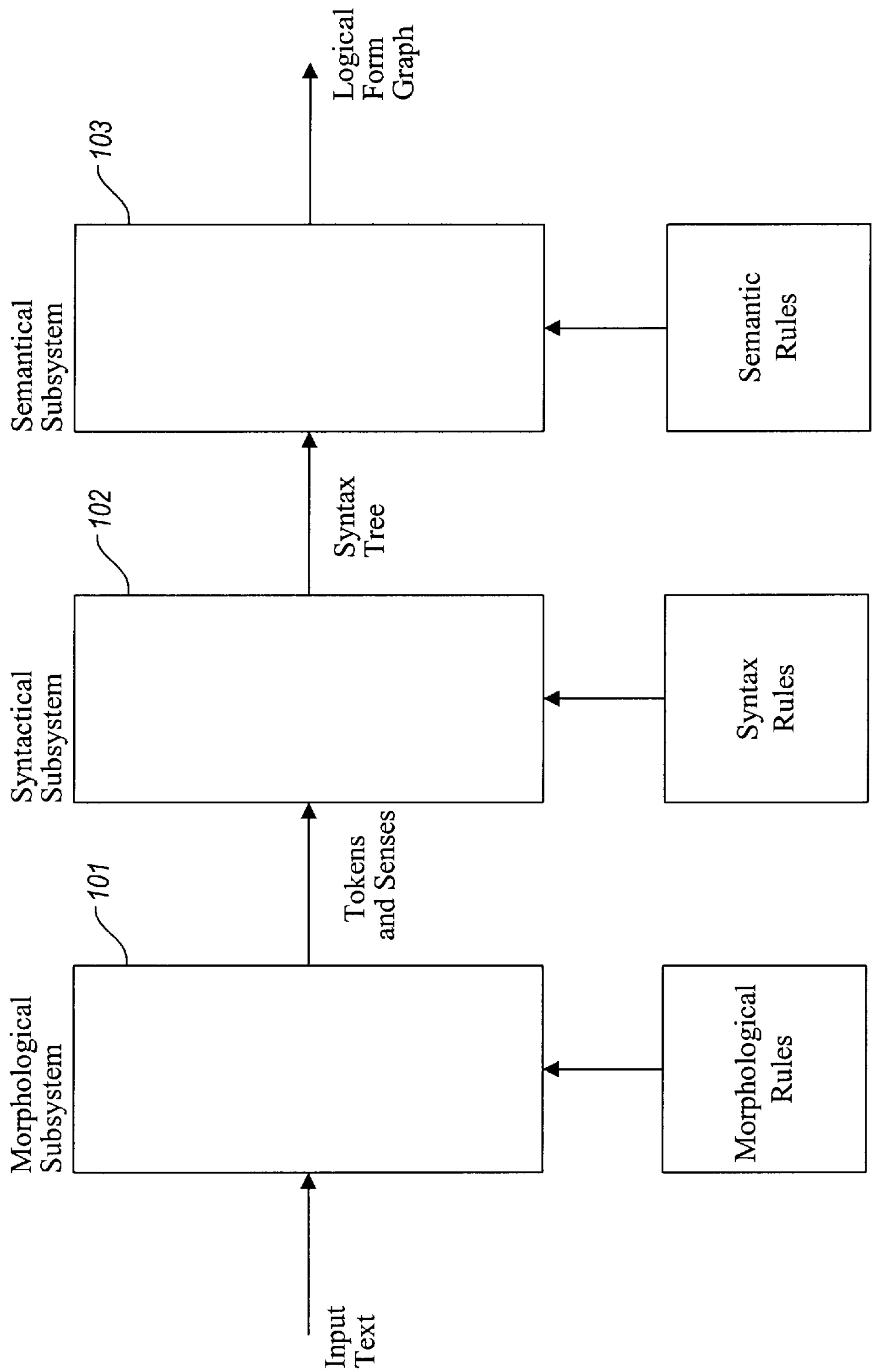
[57] **ABSTRACT**

Methods and computer systems for semantically analyzing natural language sentences. The natural language processing subsystems for morphological and syntactic analysis transform an input sentence into a syntax parse tree. Semantic analysis applies three sets of semantic rules to create a skeletal logical form graph from a syntax parse tree. Semantic analysis then applies two additional sets of semantic rules to provide semantically meaningful labels for the links of the logical form graph, to create additional logical form graph nodes for missing elements, and to unify redundant elements. The final logical form graph represents the complete semantic analysis of an input sentence.

9 Claims, 69 Drawing Sheets

The New Semantic Subsystem





(PRIOR ART)
Fig. 1

```
                                                          201
the     203
{
 Adj    204       205
        {Lemma        "the"
         Bits         Sing Plur Wa6 Det Art
                      B0 Def}     206
 Adv    212
        {Lemma        "the"
         Bits         Wa5}

 Senses  207
        {Bits  208    Sing Plur Wa6 Closed
                      Det Art Def
         Lemma        "the"
         Cat          Adj
         Infl         Adj-nil
         Defin        "used when it is clearly understood who or what is meant"
         Exs          "We have a cat and a dog.  The cat (= our cat) is black and the dog
                      (= our dog) white."
                      "the history of China (= Chinese history)"
                      "The Danes that I know work very hard."
                      "Take these letters to the post office (it is understood that you know
210                   which post office and where it is)"}   209
        {Lemma        "the"
         Cat          Adv
         Defin        "To that extent; by that much"
         Exs          "the sooner the better."}   211

        (more sense records)   213
}
```

```
                                                          202
person
{
 Noun
        {Lemma        "person"
         Bits         Pers3 Sing Humn Mass
                      Anim Count Conc C9
                      Humn_sr
         Infl         Noun-default}

 Senses
        {Lemma        "person"
         Cat          Noun
         Defin        "A living human being."
         Exs          "chairperson"
                      "spokesperson"
                      "salesperson."}

          (more sense records)
}
```

(PRIOR ART) ***Fig. 2***

```
whom
{
 Pron
            {Lemma      "who"
             Bits       Pers3 Sing Plur Rel Wh
                        Humn Obj Anim}
 Senses
            {Lemma      "who"
             Bits       Pers3 Sing Plur Rel Wh
                        Closed Humn Obj Anim
             Cat        Pron
             Defin      "(the object form of who, used esp. in writing and careful speech)"
             Exs        "With whom?"
                        "The man with whom he talked."
                        "You saw whom?"
                        "Whom did they see?"
                        "the man (whom) they saw arriving"
                        "a man (whom) you may know of"}
           (more sense records)
 }
```

```
i
{
 Noun
            {Lemma      "i"
             Bits       Pers3 Sing TakesAn
             Infl       Noun-irreg}
 Pron
            {Lemma      "I"
             Bits       Sing Nom TakesAn Persl
                        Humn Anim LexCap}
 Senses
            {Lemma      "i"
             Cat        Noun
             Infl       Noun-irreg
             Defin      "The ninth letter of the modern English alphabet."}

            {Lemma      "I"
             Cat        Pron
             Defin      "Used to refer to oneself as speaker or writer."}
           (more sense records)
}
```

```
met
{
  Verb
            {Lemma      "meet"
             Bits       Sing Plur Past
                        Pastpart
             Infl       Verb-meet}
  Senses
            {Lemma      "meet"
             Bits       Past Pastpart
             Cat        Verb}
}
```

(PRIOR ART) ***Fig. 3***

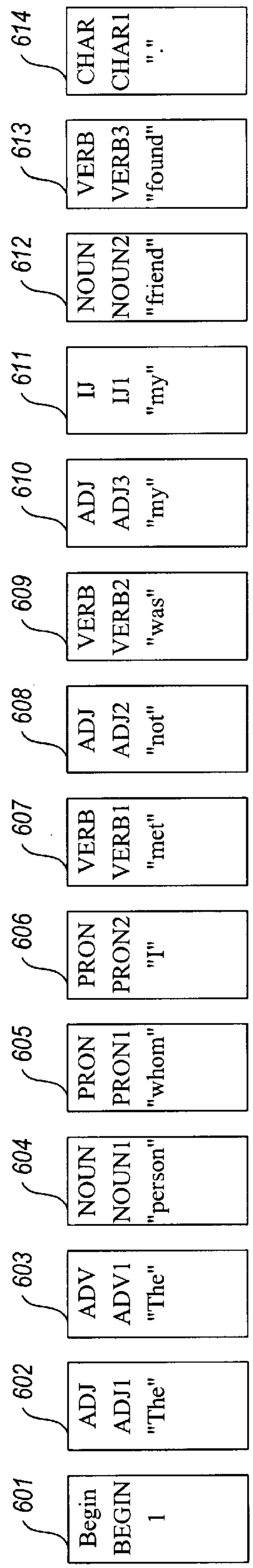
```
was
{
  Verb
           {Lemma      "be"
            Bits       Pers3 Sing Past Pers1
            Infl       Verb-be } }
  Senses
           {Lemma      "be"
            Bits       Past Pastpart
            Cat        Verb}
          (more sense records)
}
```

```
my
{
  Adj
           {Lemma      "I"
            Bits       Wa5 Det Poss Pers1 Def
                       Gen A0
            Infl       Adj-none }
  Ij
           {Lemma      "my } }
  Senses
           {Lemma      "I"
            Bits       Wa5 Closed Det Poss
                       Pers1 Def Gen A0
            Cat        Adj
            Infl       Adj-none
            Defin      "belonging to me"
            Exs        "my car"
                       "my mother"}

           {Cat        Ij
            Defin      "Used as an exclamation of surprise, pleasure, or dismay"
            Exs        "Oh, my!  What a tiring day!"}
          (more sense records)
}
```

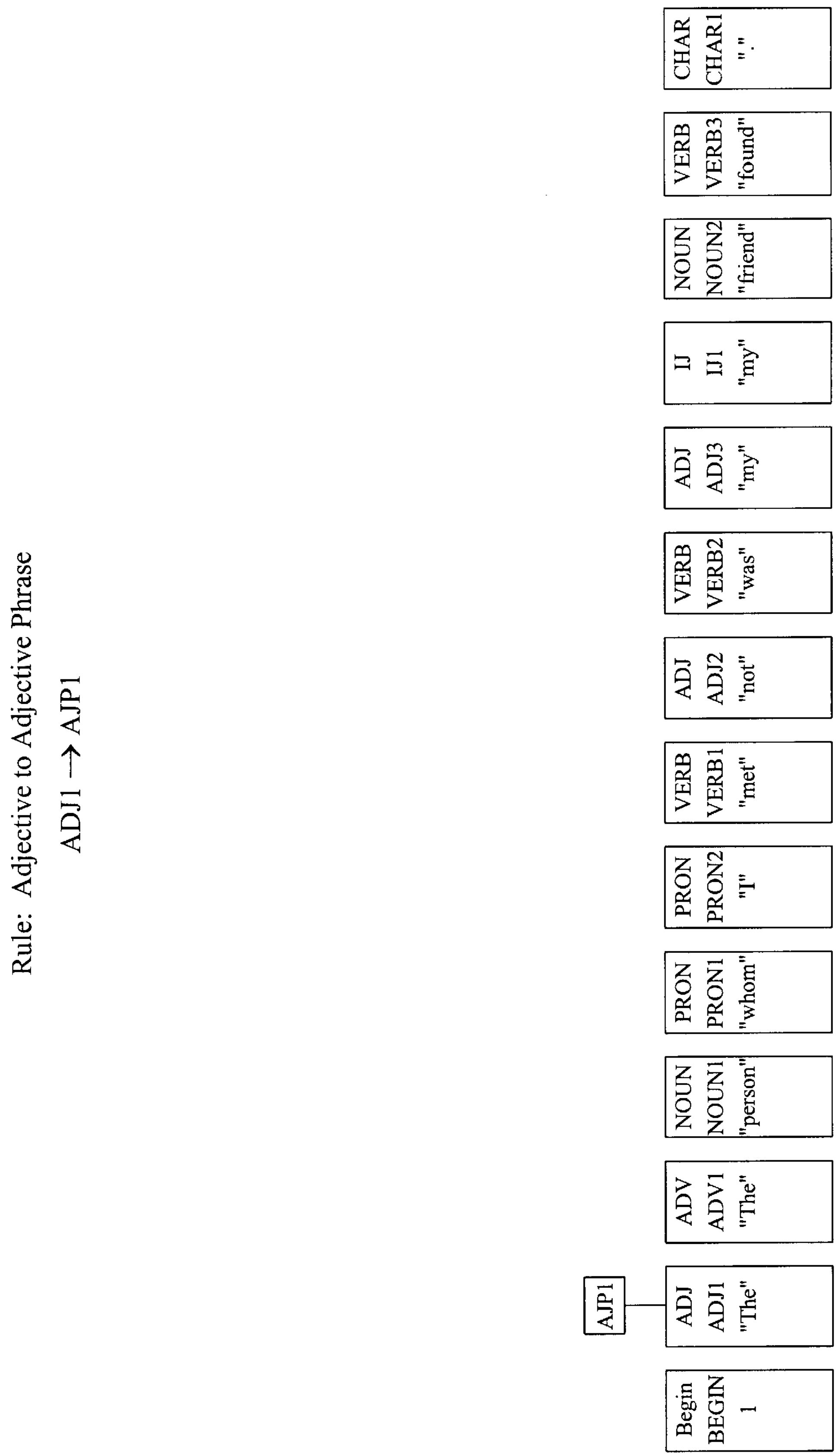
(PRIOR ART)
Fig. 4

```
friend
{
  Noun
          {Lemma      "friend"
           Bits       Pers3 Sing Humn Anim
                      Count Conc Humn_sr N0
                      Wrdy
           Infl       Noun-default
           Vprp       (of to)
           Bitrecs
                {Bits        Humn Count Conc
                 Vprp        (of)  }

                {Bits        Humn Count Conc
                 Vprp        (to)  }  }
  Verb
          {Lemma      "friend"
           Bits       Inf Plur Pres T1
           Infl       Verb-default  }  }

  Senses
          {Lemma      "friend"
           Bits       Humn Conc
           Cat        Noun
           Defin      "A person whom one knows, likes, and trusts."}

          {Bits       T1
           Lemma      "friend"
           Cat        Verb
           Infl       Verb-default
           Defin      "To befriend."}

        (more sense records)
}
```

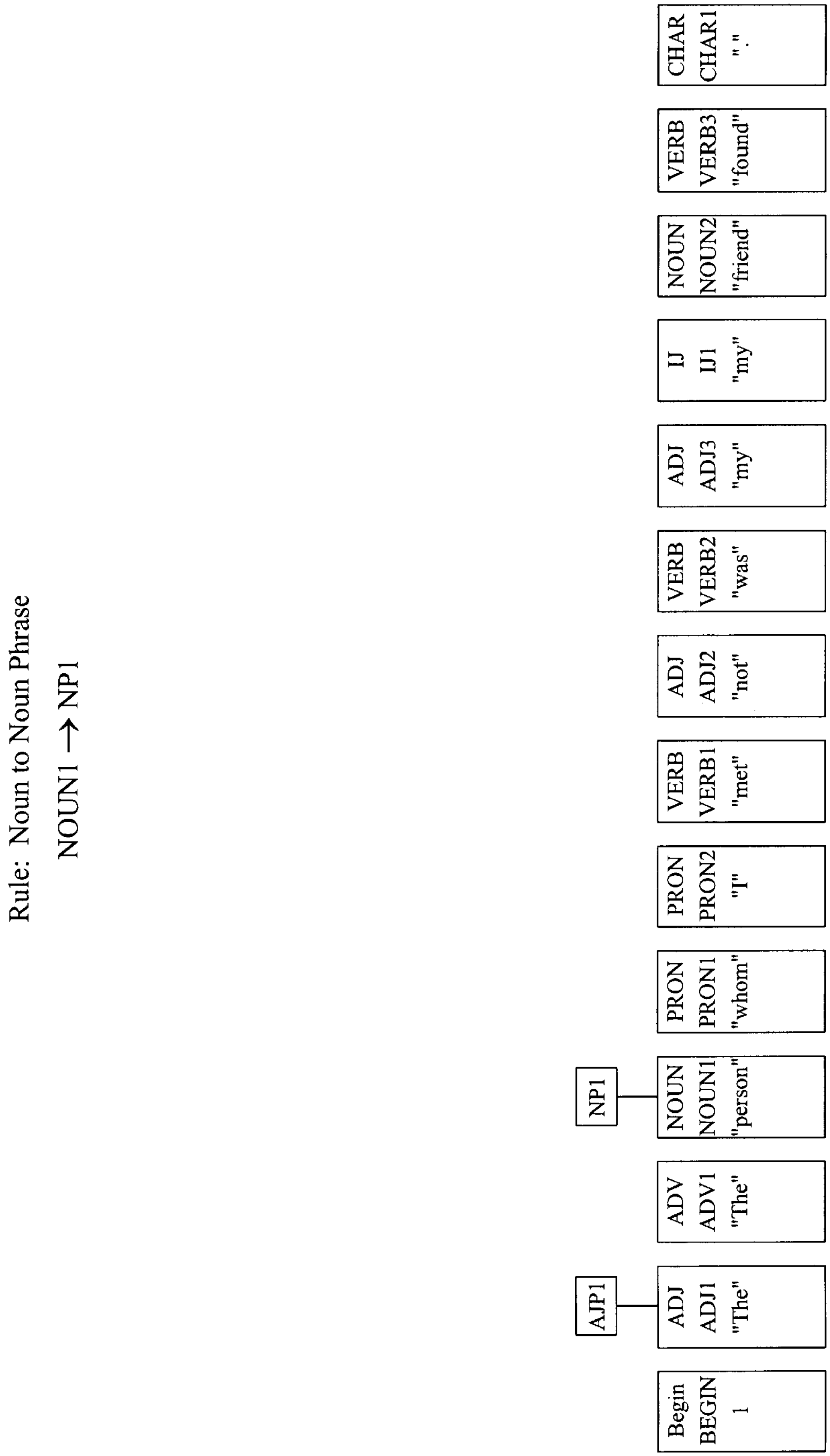
(PRIOR ART)
Fig. 5



(PRIOR ART)
Fig. 6



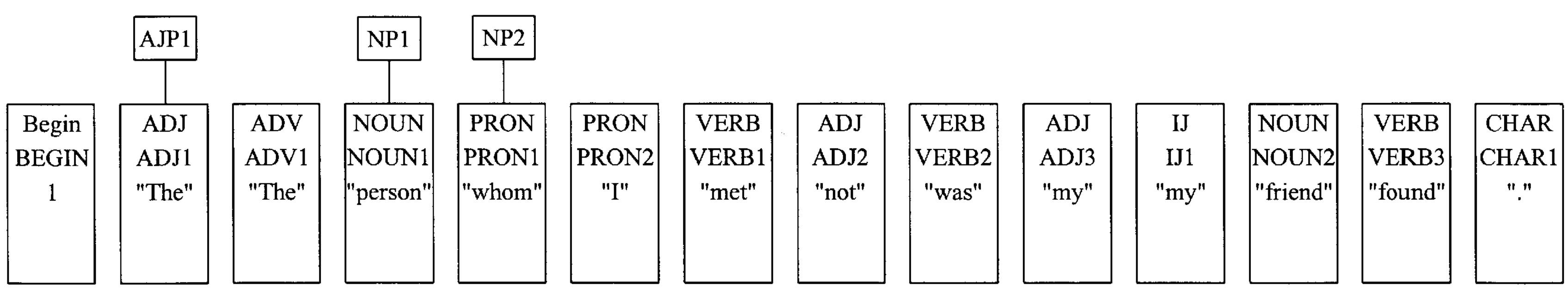
(PRIOR ART)
Fig. 7



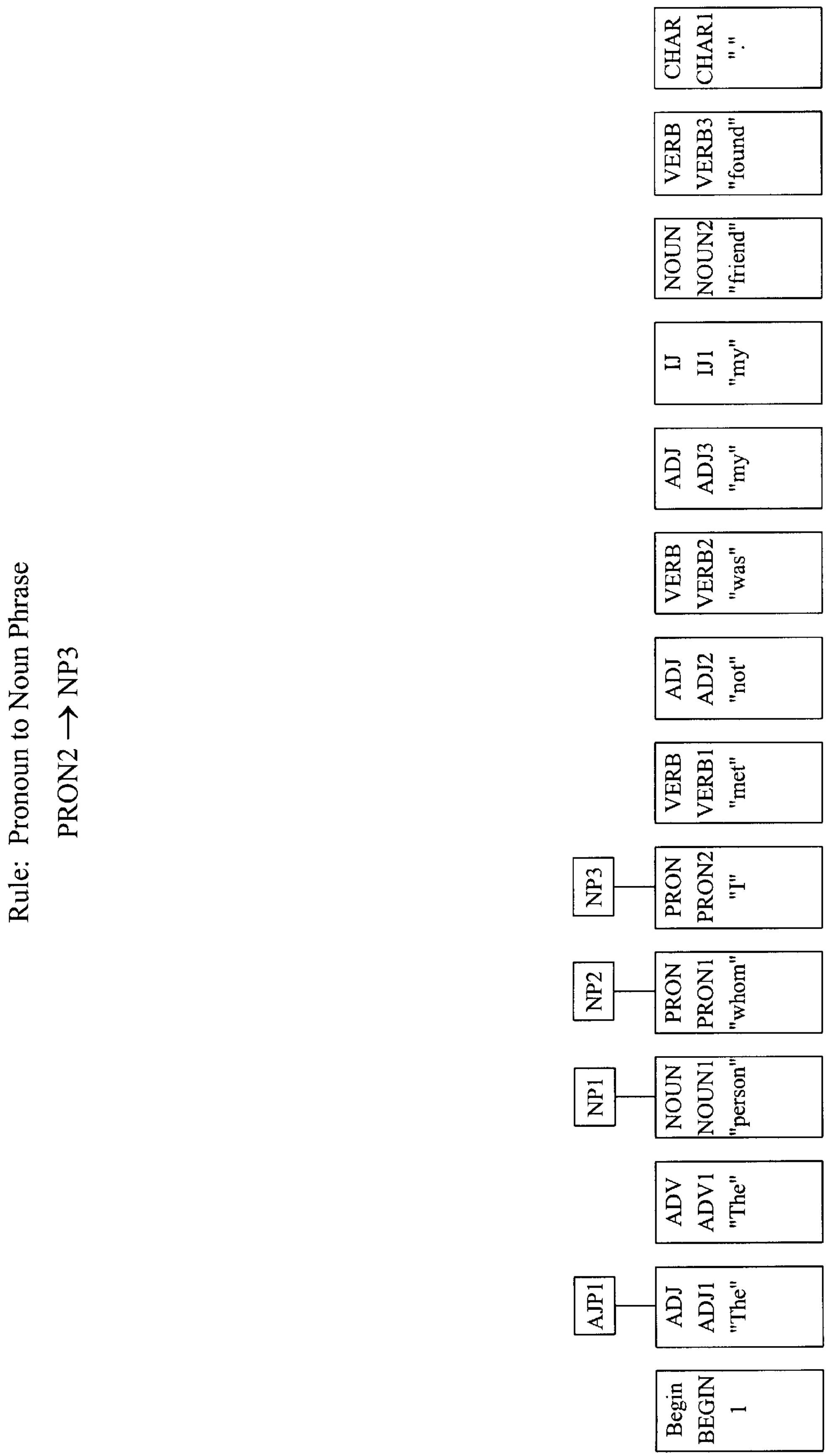
(PRIOR ART)
Fig. 8

Rule: Pronoun to Noun Phrase

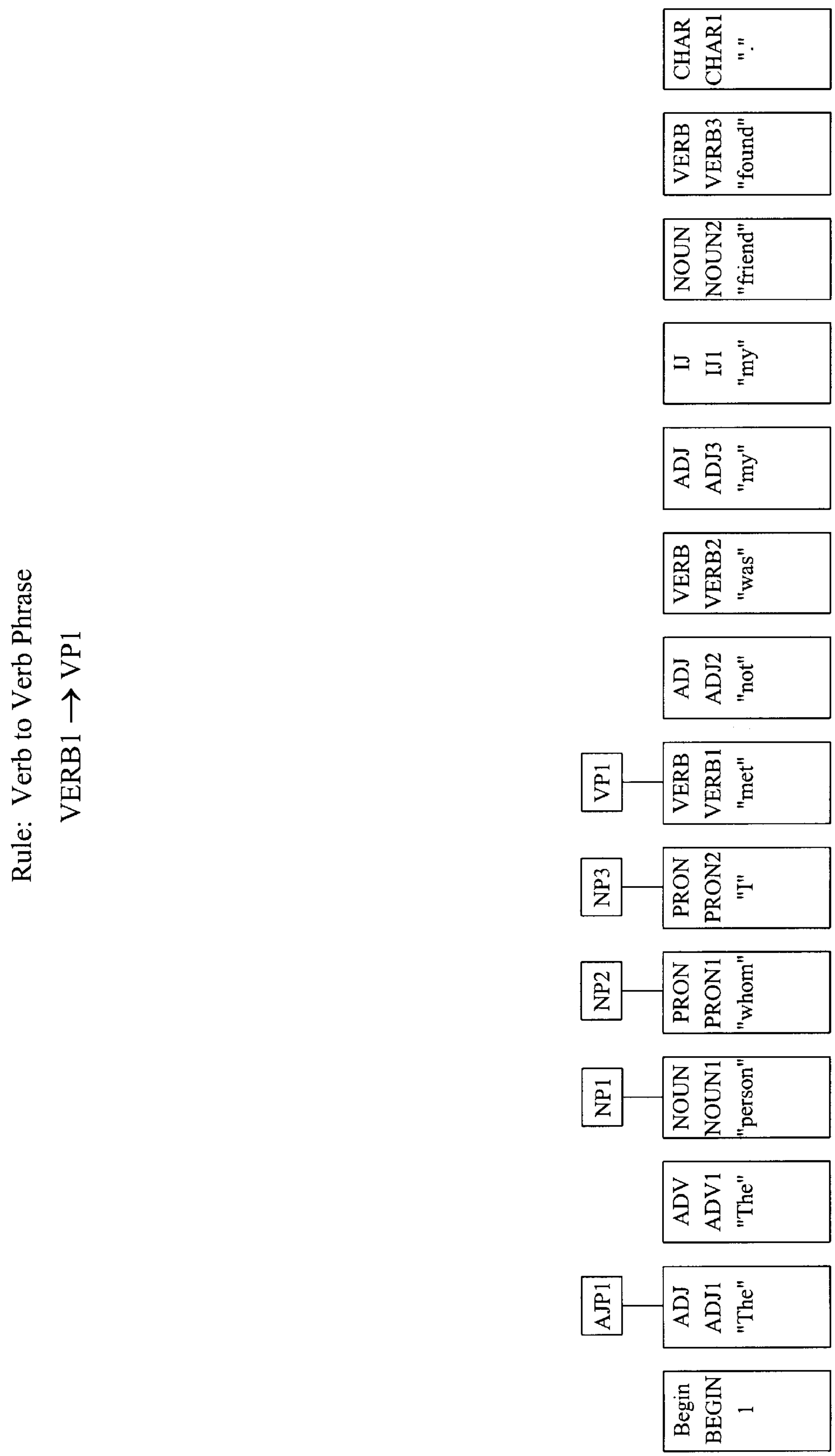
PRON1 → NP2



(PRIOR ART)
Fig. 9



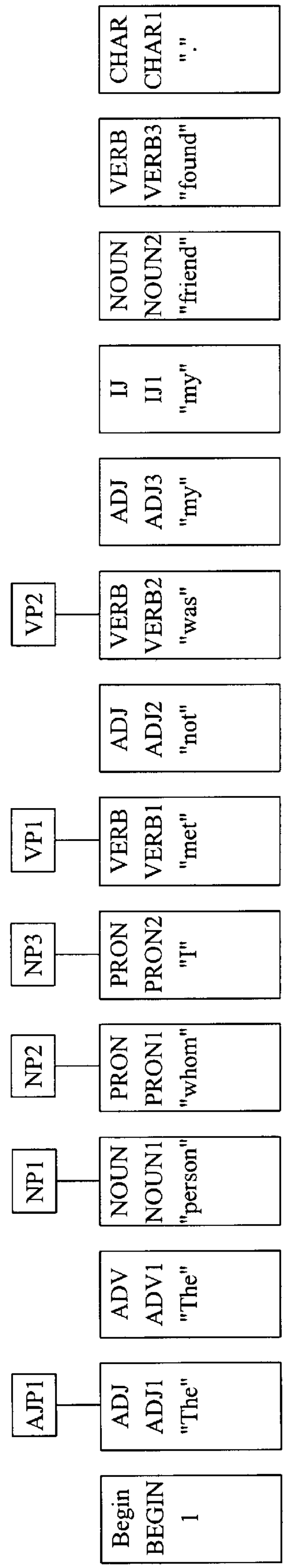
(PRIOR ART)
Fig. 10



(PRIOR ART)
Fig. 11

Rule: Verb to Verb Phrase

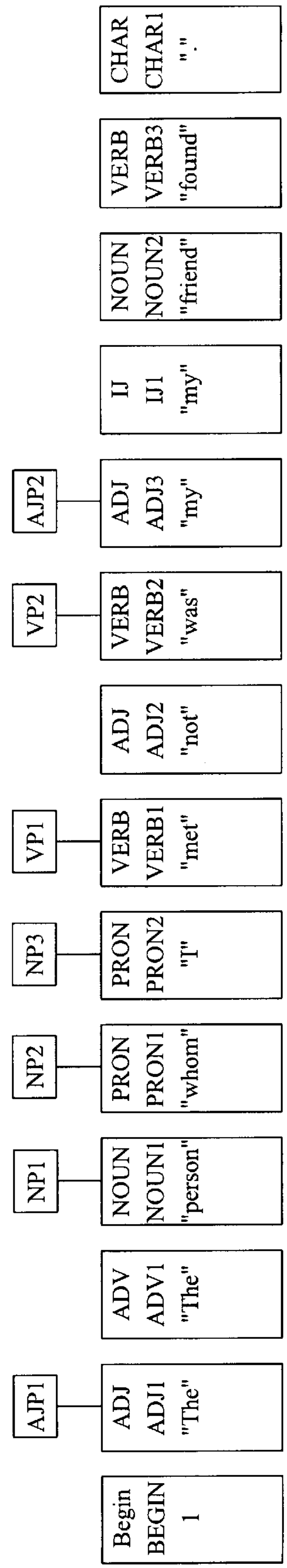
VERB2 → VP2



(PRIOR ART)
Fig. 12

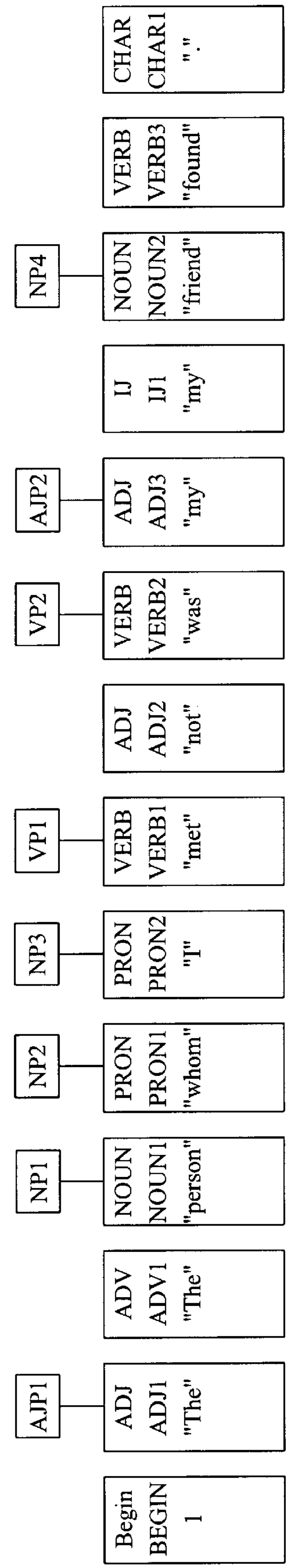
Rule: Adjective to Adjective Phrase

ADJ3 → AJP2



(PRIOR ART)
Fig. 13

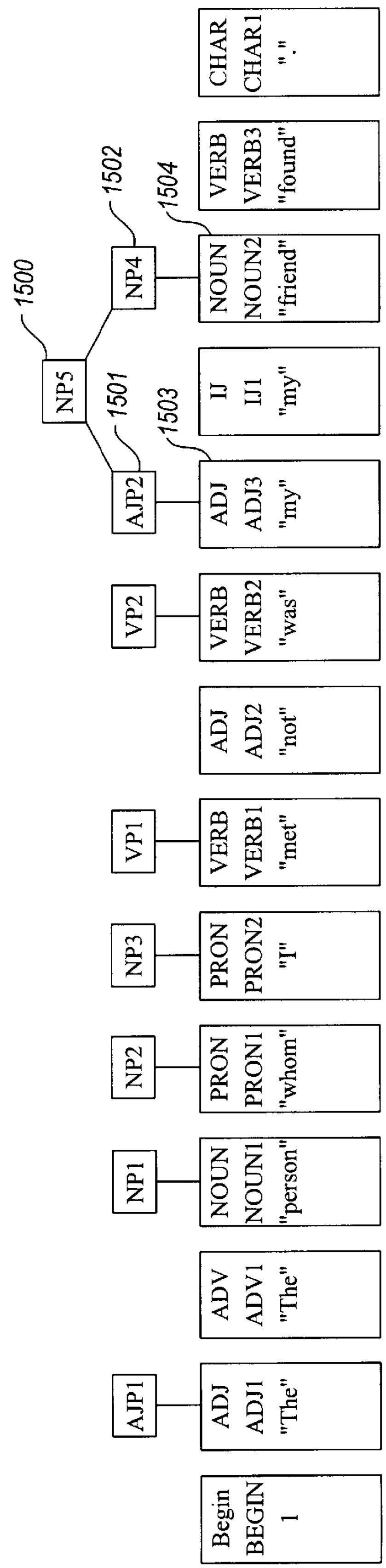
Rule: Noun to Noun Phrase



(PRIOR ART)
Fig. 14

Rule: Noun Phrase with Determer

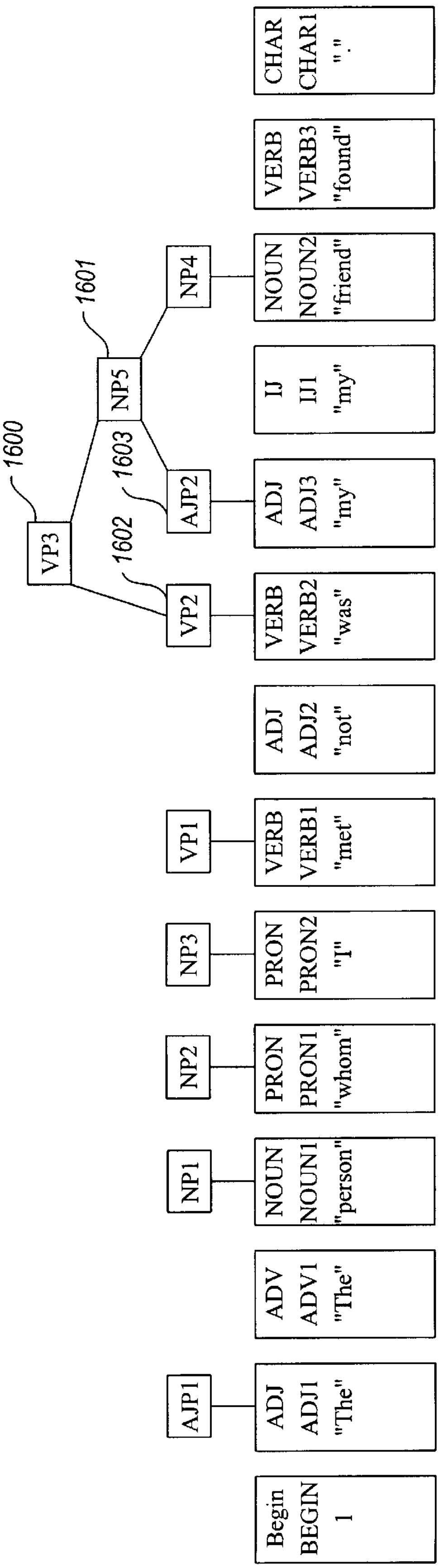
AJP2, NP4 → NP5



(PRIOR ART)
Fig. 15

Rule: Verb Phrase with Noun Phrase as Object of Transitive Verb

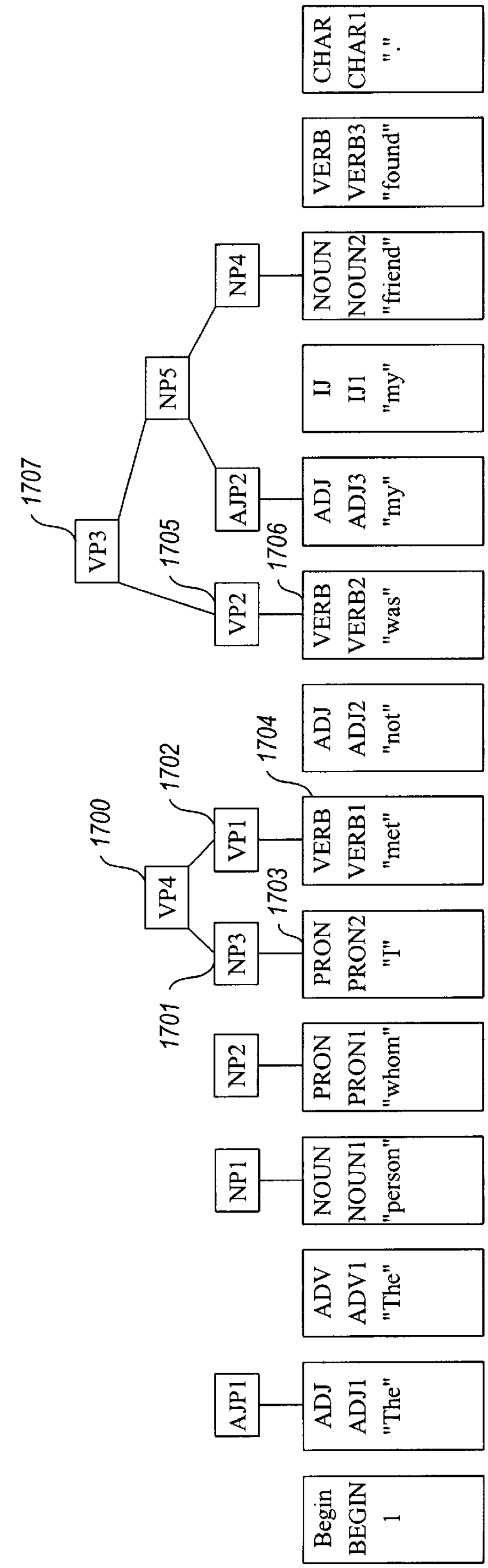
VP2, NP5 → VP3



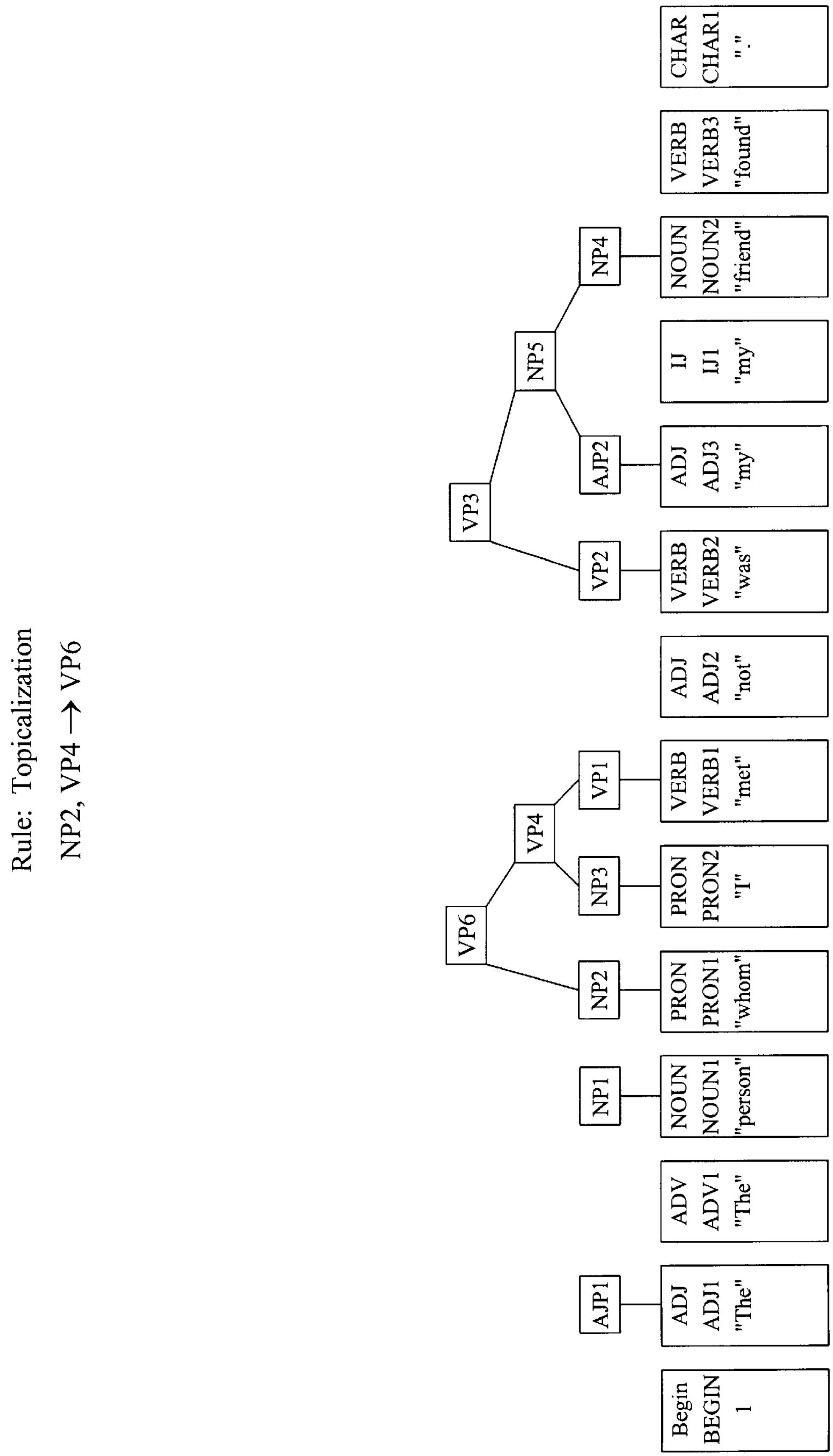
(PRIOR ART)
Fig. 16

Rule: Verb Phrase with Noun Phrase as Subject

NP3, VP1 → VP4



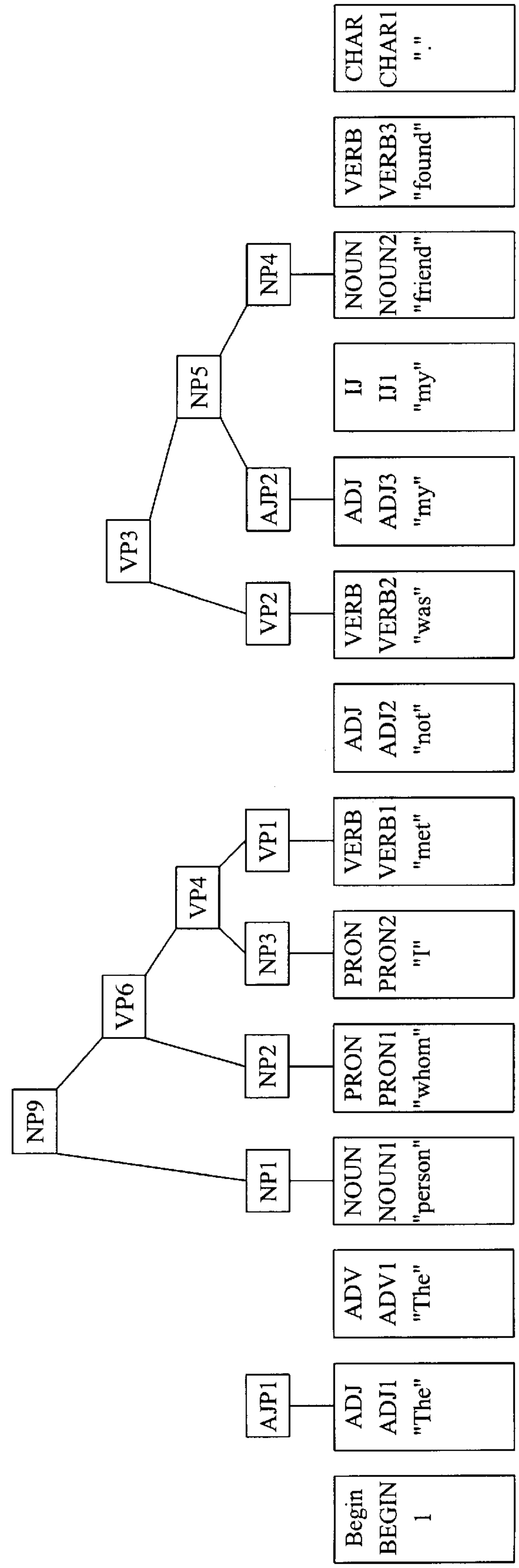
(PRIOR ART)
Fig. 17



(PRIOR ART)
Fig. 18

Rule: Noun Phrase with Relative Clause

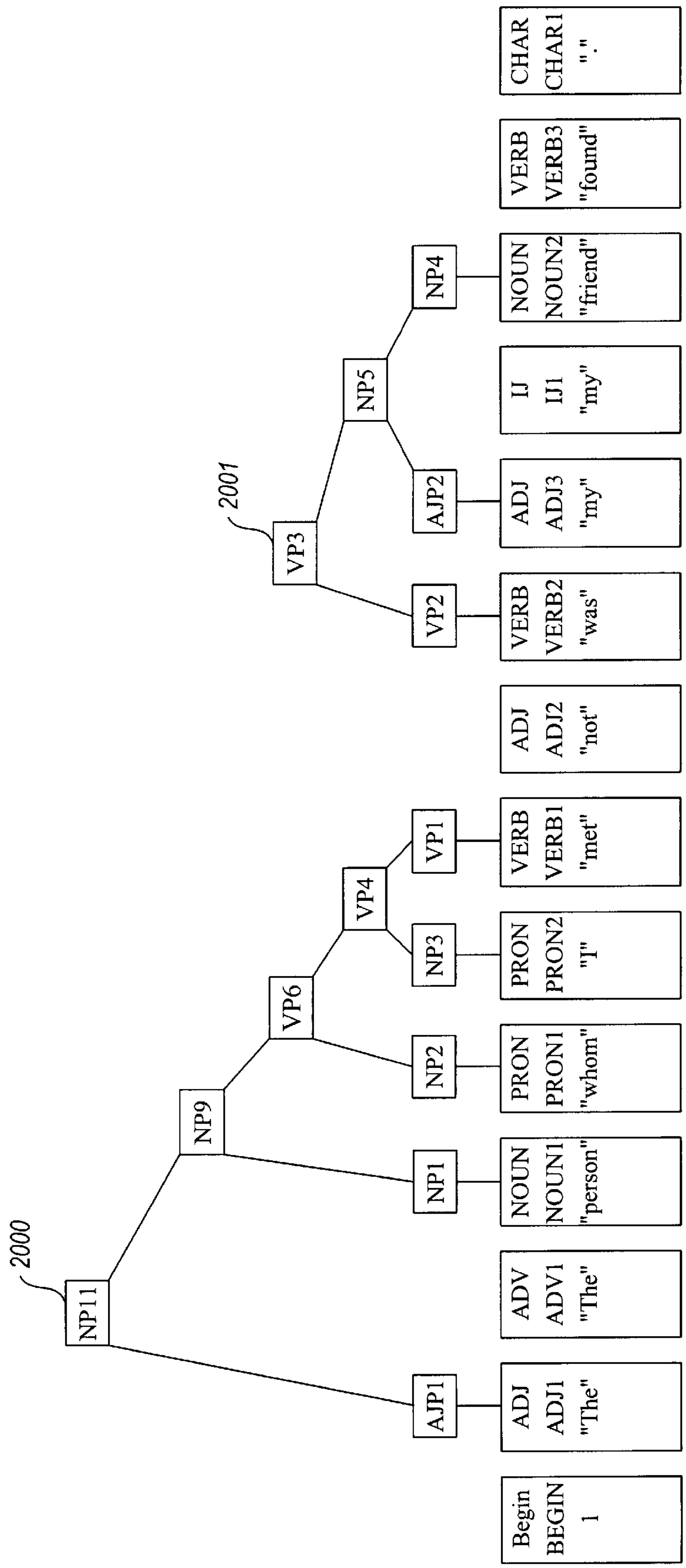
NP1, VP6 → NP9



(PRIOR ART)
Fig. 19

Rule: Noun Phrase with Determinate Quantifier

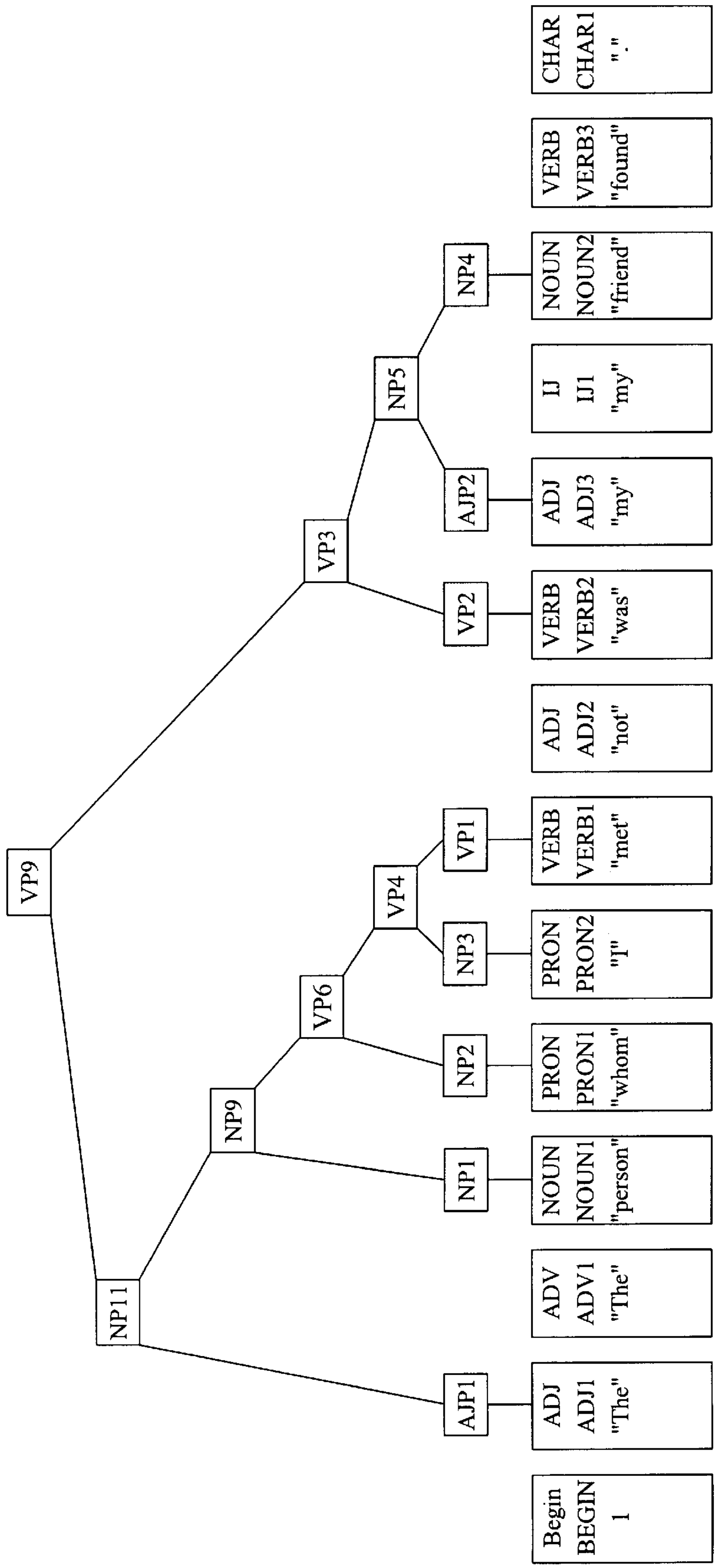
AJP1, NP9 → NP11



(PRIOR ART)
Fig. 20

Rule: Verb Phrase with Noun Phrase Subject

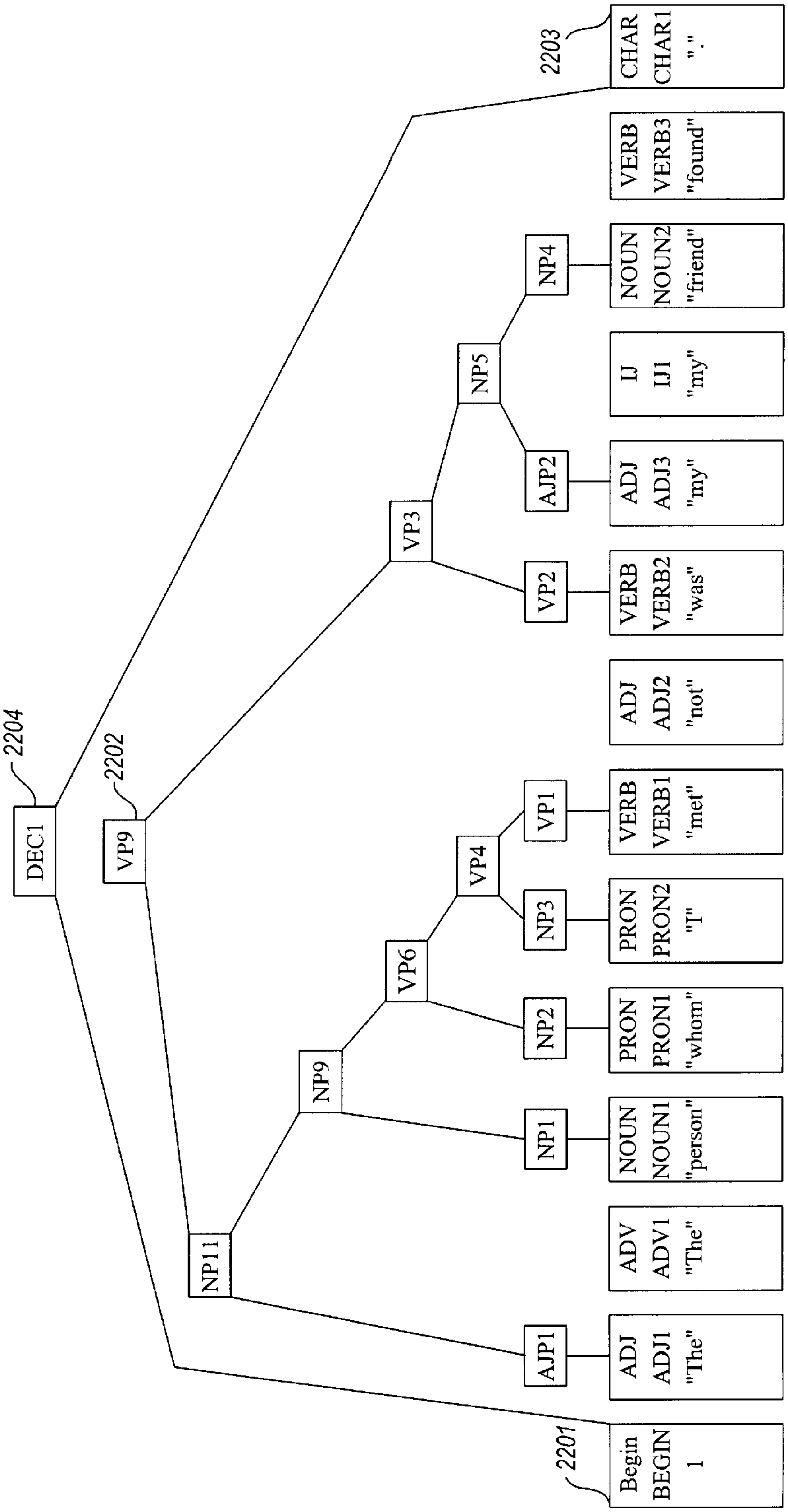
NP11, VP3 $\rightarrow$ VP9



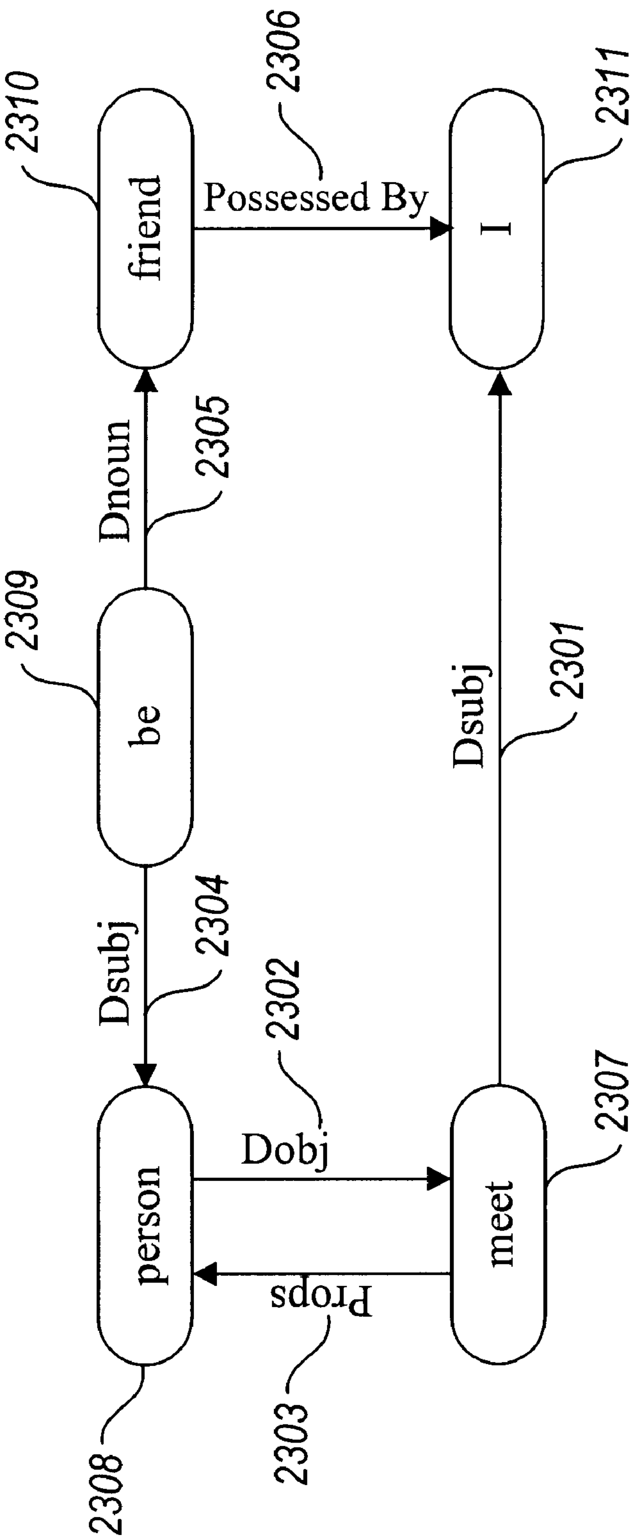
(PRIOR ART)
Fig. 21

Rule: Declarative Sentence from Begin + Verb Phrase + ".".

BEGIN1, VP9, CHAR1 $\rightarrow$ DEC1



(PRIOR ART) Fig. 22



(PRIOR ART)
Fig. 23

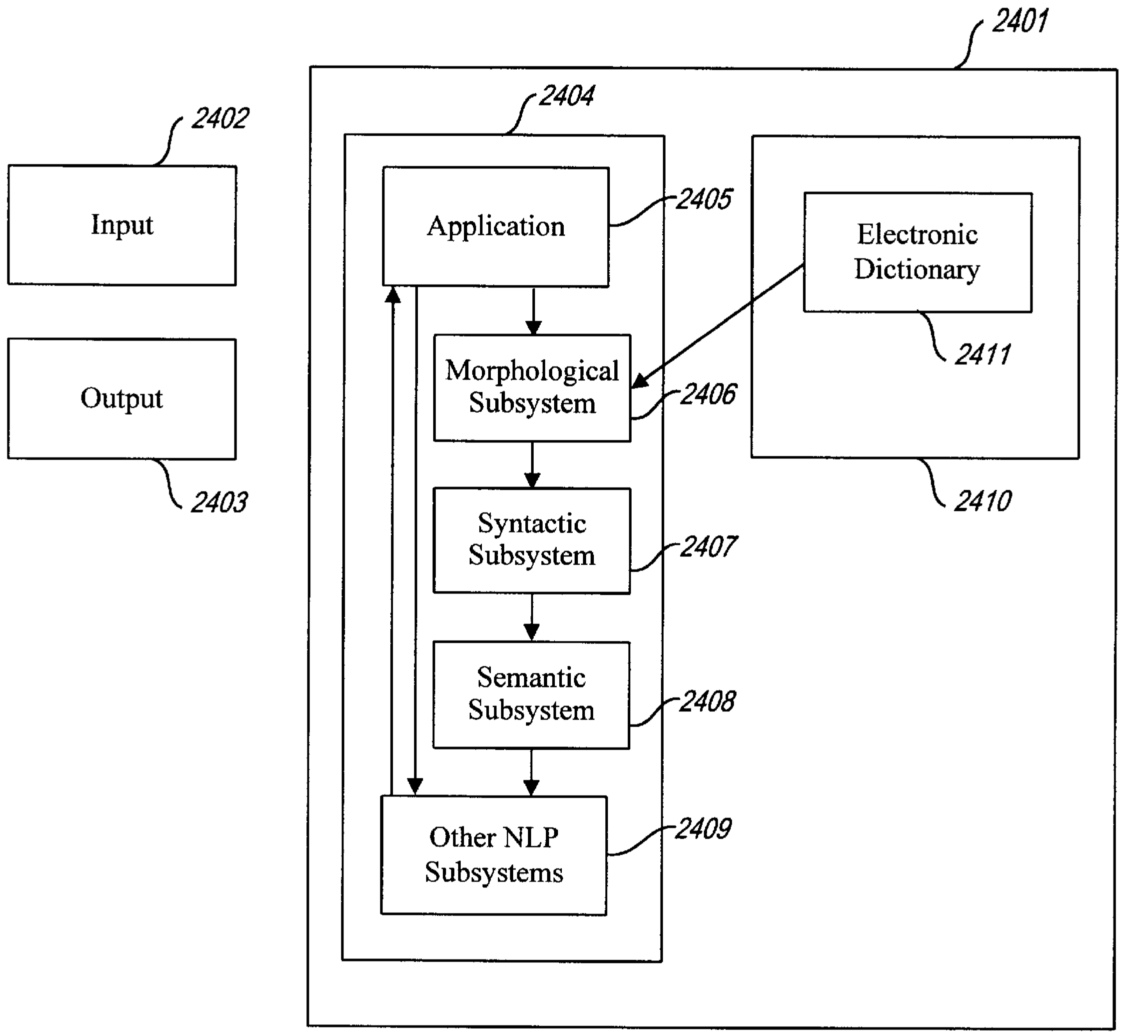

2401
2402
Input
2403
Output
2404
Application
2405
Morphological Subsystem
2406
Syntactic Subsystem
2407
Semantic Subsystem
2408
Other NLP Subsystems
2409
Electronic Dictionary
2411
2410

Fig. 24

The New Semantic Subsystem

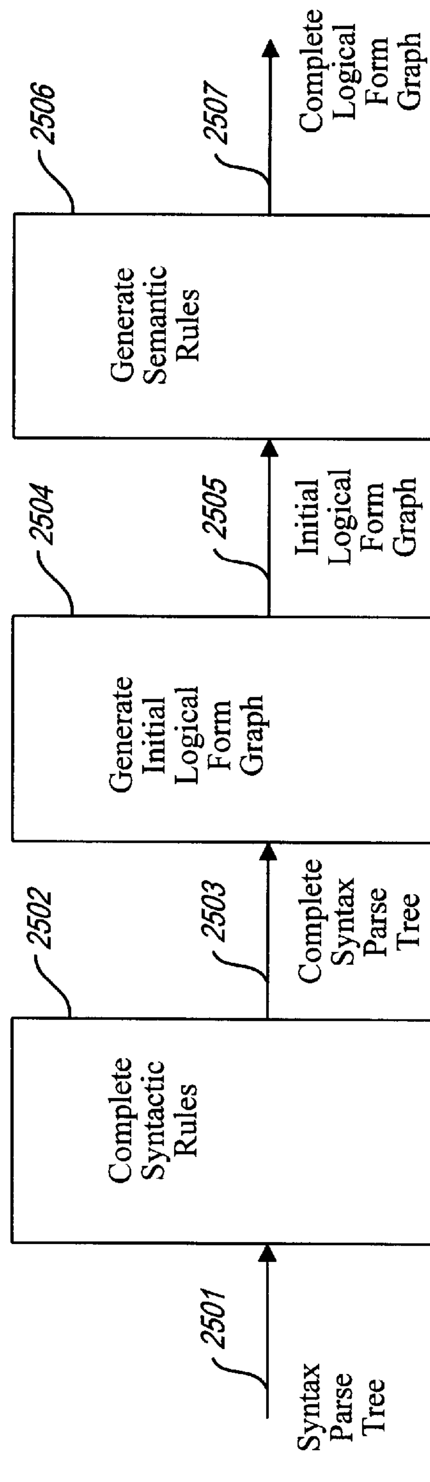


Fig. 25

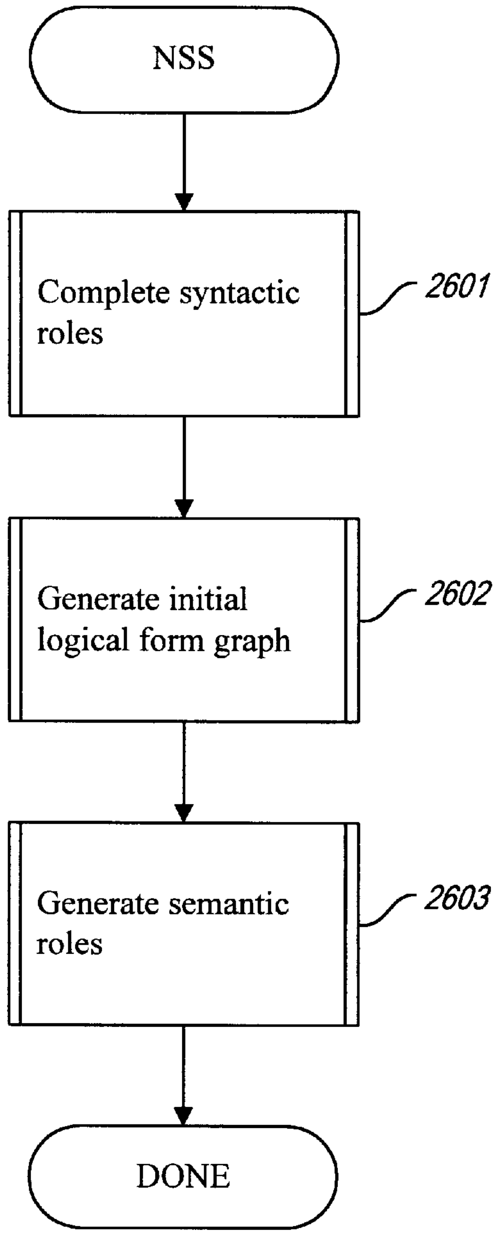

NSS
Complete syntactic roles
2601
Generate initial logical form graph
2602
Generate semantic roles
2603
DONE

Fig. 26

1. PrLF_NPQuantOf: for NPs like "a number of books," makes "books" the head and "a number of" the modifier

2. PrLF_PPQuantOf: same but for PPs, like "with a number of books"

3. PrLF_notAnaphora: prepares to fill VP anaphora like "John thought he would go but Jim though not __________"

4. PrLF_soAnaphora: prepares to fill VP anaphora like "Mary wondered if it was true but Jane knew so __________"

5. PrLF_toAnaphora: prepares to fill VP anaphora like "Chris wanted to go but Pat didn't want to _____"

6. PrLF_You: supplies the understood "you" in commands like "(You) please close the door"

7. PrLF_HowAbout: supplies the understood "you" in constructions like "How about (you) closing the door"

8. PrLF_We: supplies the understood "we/us" in constructions like "Let's (us) go to the movies"

9. PrLF_I: supplies the understood "I" in, for example, "(I) thank you" or "(I) Have not yet received your letter"

10. PrLF_SubjectMods: connects "we" and "all" in, e.g., "We are all reading the book"; connects "he" and "hungry" in, e.g., "He arrived hungry"

11. PrLF_RightShift: connects "the man" and "who was my friend" in, e.g., "The man arrived who was my friend"

12. PrLF_InfclPP: prepares for correct interpretation in constructions like "a person on whom to rely"

13. PrLF_QuantifierEllipsis: having to do with the resolution of pronoun references

14. PrLF_PossessivePronHead: having to do with the resolution of pronoun references

15. PrLF_PossibleCorefsOfProns: having to do with the resolution of pronoun references

16. PrLF_VPAnaphora: identifies and fills missing arguments in all cases of VP anaphora, e.g., "Sarah likes basketball and I do too"

17. PrLF_DistCoords: distributes elements across coordinated structures, like "They washed _____ and dried the dishes"

Fig. 27

PrLF_You

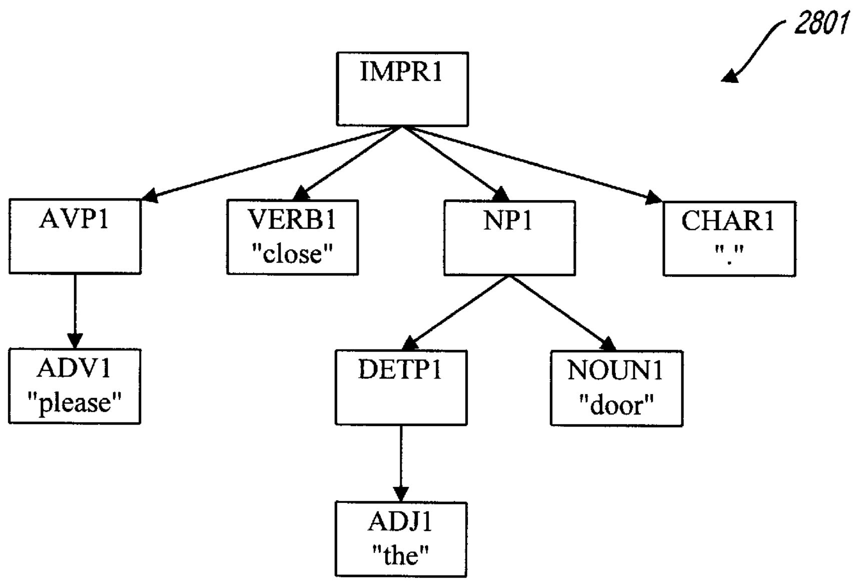
If the Syntax Record
has the attribute "Infinitive"
and does not have the attribute "Subject"
or has the attribute "Verb Phrase Invert" and does not have any of the attributes "Object2," "Yes/No/Question," or "Old Subordinate Clause"
and does not meet the "There Subject Test"
and does not have the "Coordinate Constructions" attribute
and does not have any premodifiers with the node type "Auxiliary Phrase" or the attribute "Modal Verb"
and does not have any premodifiers with the lemma "let" or the node type "Adverbial Phrase,"
and does not have the node type "Abbreviated Clause," "Auxiliary Phrase," "Complement Clause," "Infinitive Clause," "Noun Relative," "Past Participle Clause," or "Relative Clause"
and does not have a parent with the node type "Past Participle Clause"
and if the head of the parent has node type "Conjunction,"
then the parent does not have a "Subject" attribute and does not have the node type "Auxiliary Phrase," "Complement Clause," "Infinitive," "Noun Relative," or "Relative Clause"
and if there is an Auxiliary Attribute on its Head
then for all its Premodifiers their Lemma must not be "neither" nor "so,"
and if it has a Do Modifier,
then it must have an Infinitive attribute and either there must not be a Modal on the First Verb Attribute, or the Lemma of its First Verb must be either "dare" or "need,"
and it if has a Perfective attribute,
then its Lemma must be do,
and if it has a Verb Phrase Invert attribute,
then either there must not be an L9 attribute
or there must not be a Comma attribute and for all of its Premodifiers their node type must not be equal to "Prepositional Phrase" and for all of its Premodifiers their node type must either not be "Adverbial Phrase" or there must be a Comma attribute or the node type of their Head must be an Interjection,
and has neither "ect" nor "ect." as its Lemma,
and if its Lemma is "suffice,"
then the Lemma of its Object1 cannot be "it,"
and if its Lemma is "thank,"
then the Lemma of its Object1 cannot be "you,"

Then
create a pronoun record for the lemma "you";
make the Subject attribute of the syntax record be a copy of the pronoun record and set the Segtype to be "NP," set the node type to be Segtype, and set the head attribute to be the pronoun record;
and set the premodifiers of the syntax record to be the value of the subject attribute plus all of the original premodifiers and set the Undersubject attribute flag.

Fig. 28A

Sentence represented by parse tree: "Please close the door."

Syntax parse tree generated by syntactic subsystem:



Rule PrLF_You

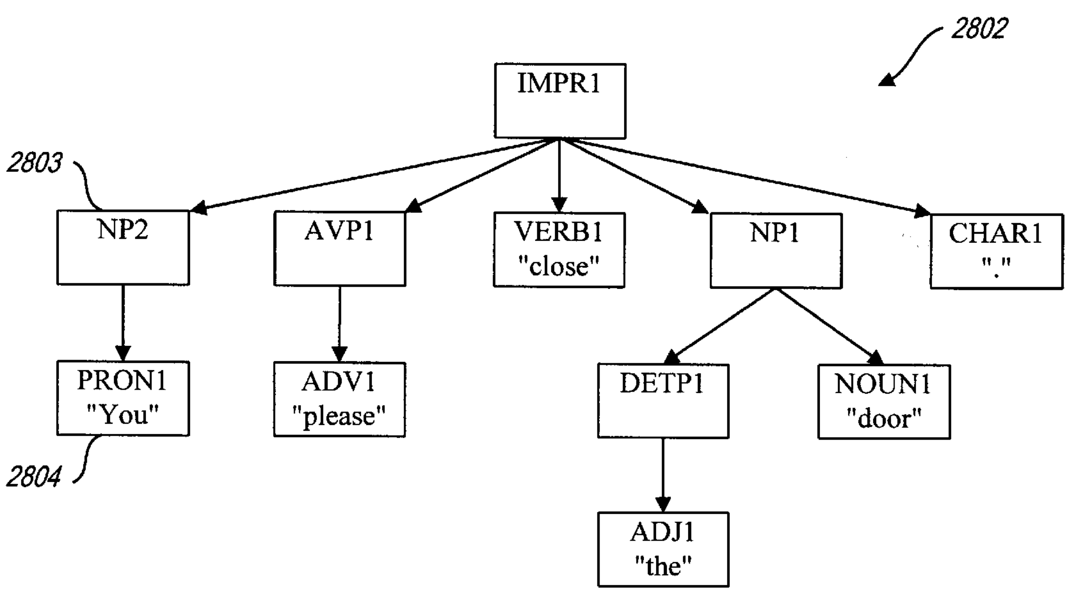


Fig. 28B

1. TrLF_LongDist1: locates NPs that are removed from their semantic heads and reattaches them, e.g., "Who did John say that Mary likes_(who)_?"

2. TrLF_LongDist2: performs the same kind of long-distance attachment for AJPs, INFCLs, PPs, PRPRTCLs, PTPRTCLs, SUBCLs

3. TrLF_PhrasalVerb: defines semantic objects of certain verbs when they appear hidden inside PPs: "his hat" is really the semantic object of "took off" in "He took off his hat"

4. TrLF_ControlwNP: e.g., in "Chris told Pat what to eat," "Pat" is really the subject of "eat" and "what" is its object

5. TrLF_ControlwAJP: e.g., in "I find this difficult to believe," "this" is really the object of "believe"

6. TrLF_ForInfcl: used in "for-to" constructions, e.g., in "For Mary to talk to John is easy," "Mary" is really the subject of "talk"

7. TrLF_ForInfclCoords: used in "for-to" constructions that have coordinated PPs

8. TrLF_MoveProp: given our strategy for attachment, it is sometimes necessary to move clauses from a lower to a higher level so that the proper argument structure can be assigned

9. TrLF_ControlatVP: e.g., in "Farmers grow food by using salt water," "farmers" is really the subject of "use salt water"

10. TrLF_PropsAsArgs: some clauses (propositions) can be arguments, e.g., in "Has he to answer the letter?" the object of "has" is "to answer the letter"

11. TrLF_Extraposition: e.g., in "It makes me happy to meet you," the real subject of "makes" is "to meet you" -- "it" is an empty word and must drop out

12. TrLF_FillCoords: fills in missing arguments in coordinated structures

13. TrLF_RedefineSubject: e.g., in "What is John's address?" we interpret "John's address" as the logical subject even though it is not in canonical subject position

Fig. 29

TrLF MoveProp

If the syntax Record
has either a node type of Abbreviated Clause, Infinitive Clause, Present Participle Clause, Past Participle Clause
or if it has a Gerund attribute and an Object of a Prepositional Phrase and
if it has Premodifiers,
then the node type of all Premodifiers must be either Auxiliary Phrase, Adverbial Phrase, or Prepositional Phrase,
and the node type of the Head attribute of the Parent is not "verb"
and this syntax record is the last of the post modifiers of its parent
and this syntax record is not in the coordinates attribute of its parent
and among the ancestors of the parent there is a record whose node type of the Head is "Verb" but none of those ancestors can have a Coordinates attribute (this record will later be referred to as "same ancestor")
and there should be no For To Prepositional Phrase attribute on the parent,
and if the node type equals Infinitive Clause,
then there must be either no WH attribute on PP obj of the parent or the syntax record is not equal to the Nominal Relative of the parent,
and if the node type is either Present Participle or Past Participle,
then its Parent does not have an Object of a Prepositional Phrase,
and if the node type is a Present Participle Clause,
then there must be an 'ING' Complement on the same ancestor
and if the node type is a Past Participle Clause,
then there must be a V8 (code from Longman's dictionary) attribute on the same ancestor and if there is an X1 attribute on the syntax record then there must not be an Object 1
and there is no B3 attribute on its parent,
and this syntax record must follow the head of the same ancestor or there is a passive attribute on the same ancestor
and if the Lemma of the Parent is 'certain'
then the node type of the parent must not be an Adjective Phrase
and if the Lemma of the Preposition is either "as" or "of,"
then there must be a To Noun attribute of its Parent
and if the Lemma of the same ancestor is either "be" or "become"
then either the node type of the Parent must be an Adjective Phrase
or there must be a WH attribute on the Parent
or there must be both a To Noun attribute on parent and no There Subject Test on the same ancestor
or the Lemma of the Parent must be one of the following: "delight," "horror," "joy," "pleasure," "riot," "shame," "surprise," "terror,"

Fig. 30A

TrLF MoveProp

Then

the syntax record whose attributes will be changed is the same ancestor syntax record (see above);

if the Parent of the syntax record has the Subject attribute and the Parent of syntax record also has the Object attribute,

then delete the object attribute from the ancestor;

if the Parent of the syntax record has the Subject attribute and the Parent of the Syntax Record does not also have the Object 1 attribute,

then set the Subject attribute of same ancestor to be the syntax record;

if the same ancestor has

the DI (Longman code) attribute and there is an Object Complement attribute and no Indirect Object attribute and there is a To Infinitive on the syntax record and the Parent of syntax record is the Object

and there is no WH attribute on the Parent of Syntax Record

and either there is an Animate attribute on Parent of syntax record

or there is a Case attribute on Parent of Syntax Record and the Lemma of the Parent of the syntax record is not "it"

or there is a Human attribute on the Parent of Syntax Record

or there is a Proper Name attribute on Parent of syntax record,

then make the Indirect Object Attribute on same ancestor equal to that of the Parent of syntax record;

if there is a To Infinitive attribute on the syntax record and no Passive attribute on same ancestor,

then make the Predicate Complement attribute equal to the syntax record;

if the Parent of syntax record is in the Propositions attribute of same ancestor,

then take that Propositions list and replace the Parent of the syntax record with the syntax record itself in the propositions list;

delete the Infinitive attribute of the Parent of the syntax record;

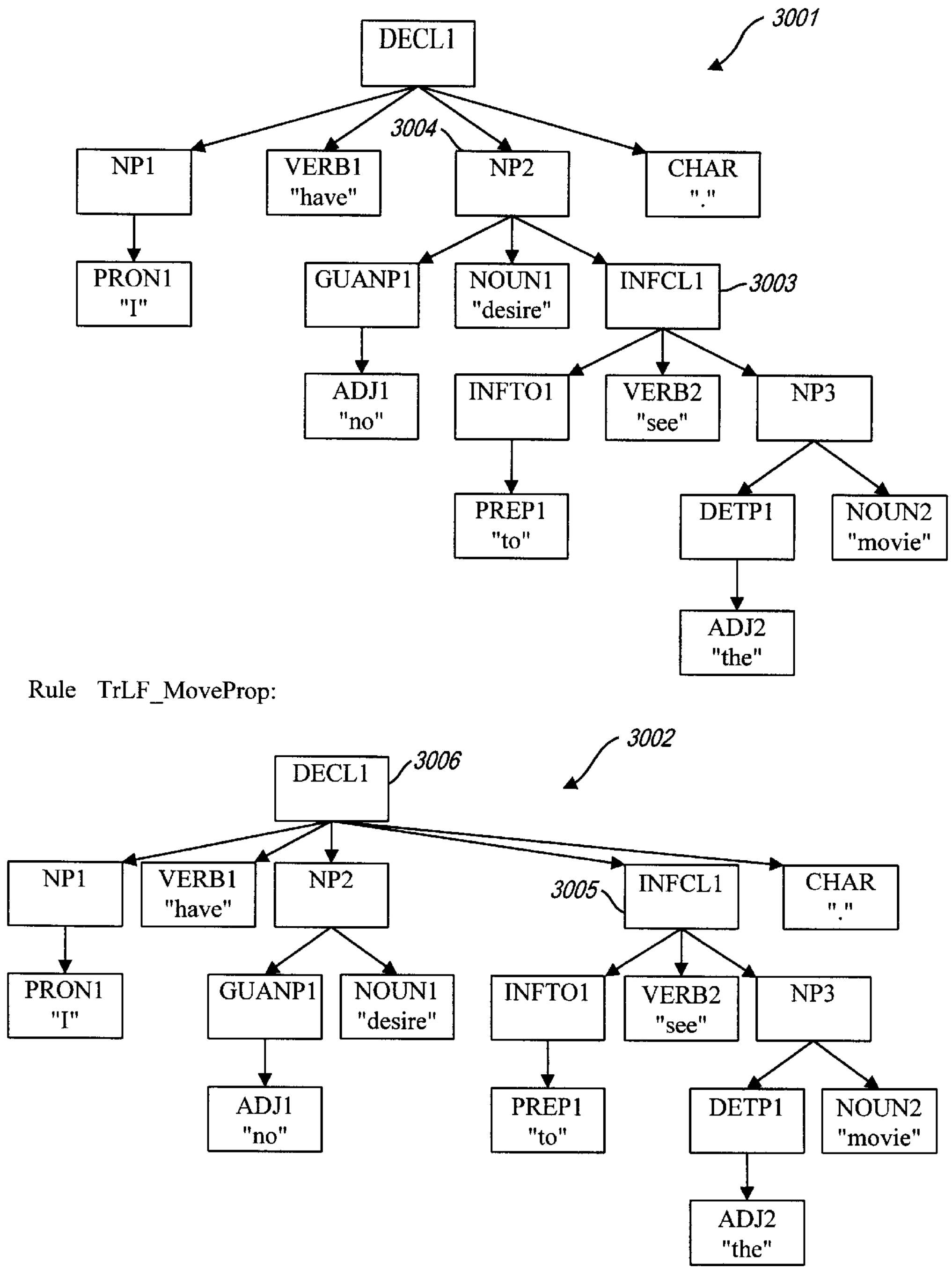
delete the Alternatives attribute on the syntax record;

reattach the syntax record to the same ancestor.

Fig. 30B

Sentence represented by parse tree: "I have no desire to see the movie."

Syntax parse tree prior to applying rule TvLF_MoveProp:



Rule TrLF_MoveProp:

Fig. 30C

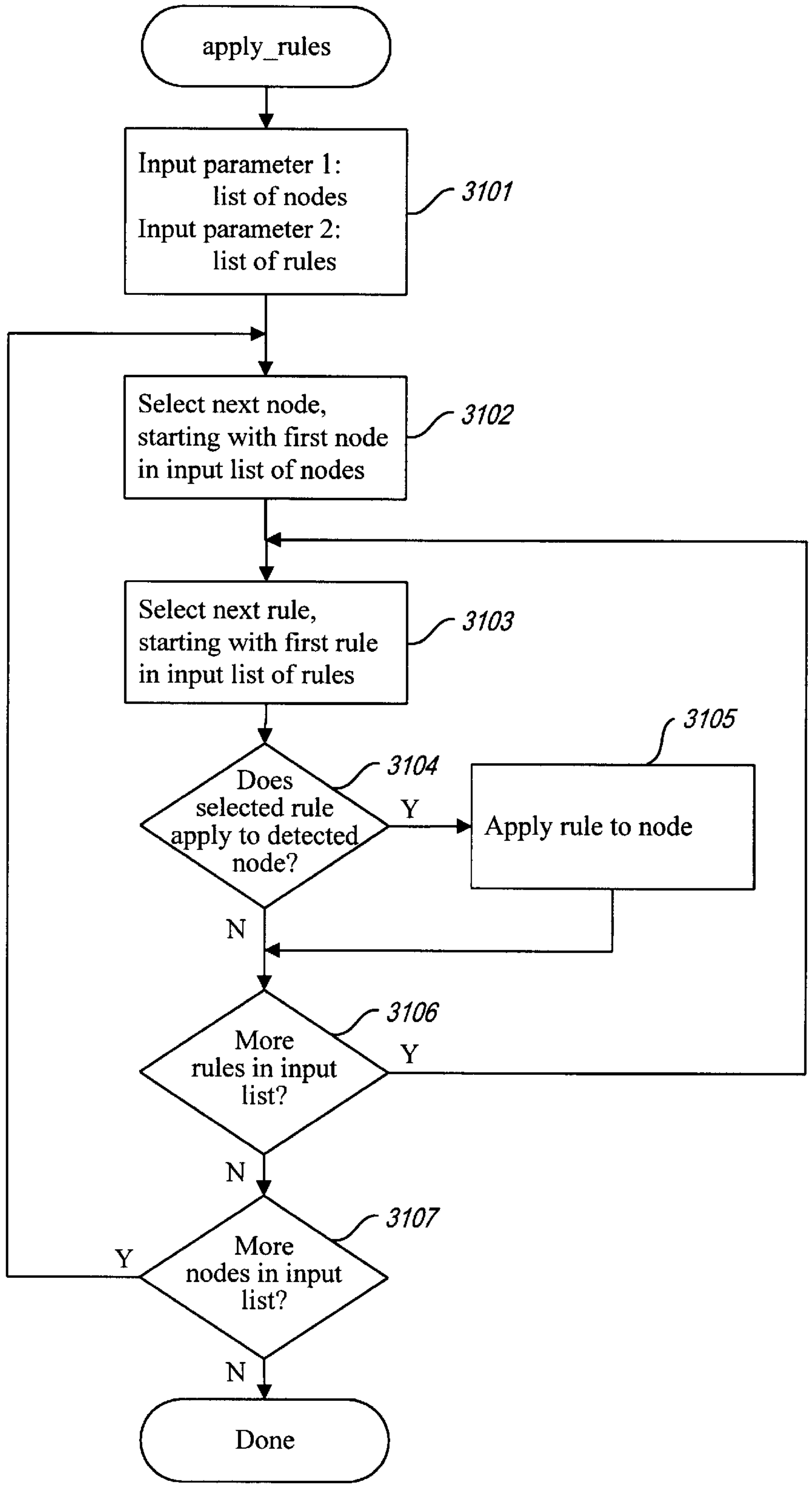

apply_rules
Input parameter 1: list of nodes
Input parameter 2: list of rules
3101
Select next node, starting with first node in input list of nodes
3102
Select next rule, starting with first rule in input list of rules
3103
Does selected rule apply to detected node?
3104
Y
Apply rule to node
3105
N
More rules in input list?
3106
Y
N
More nodes in input list?
3107
Y
N
Done

Fig. 31

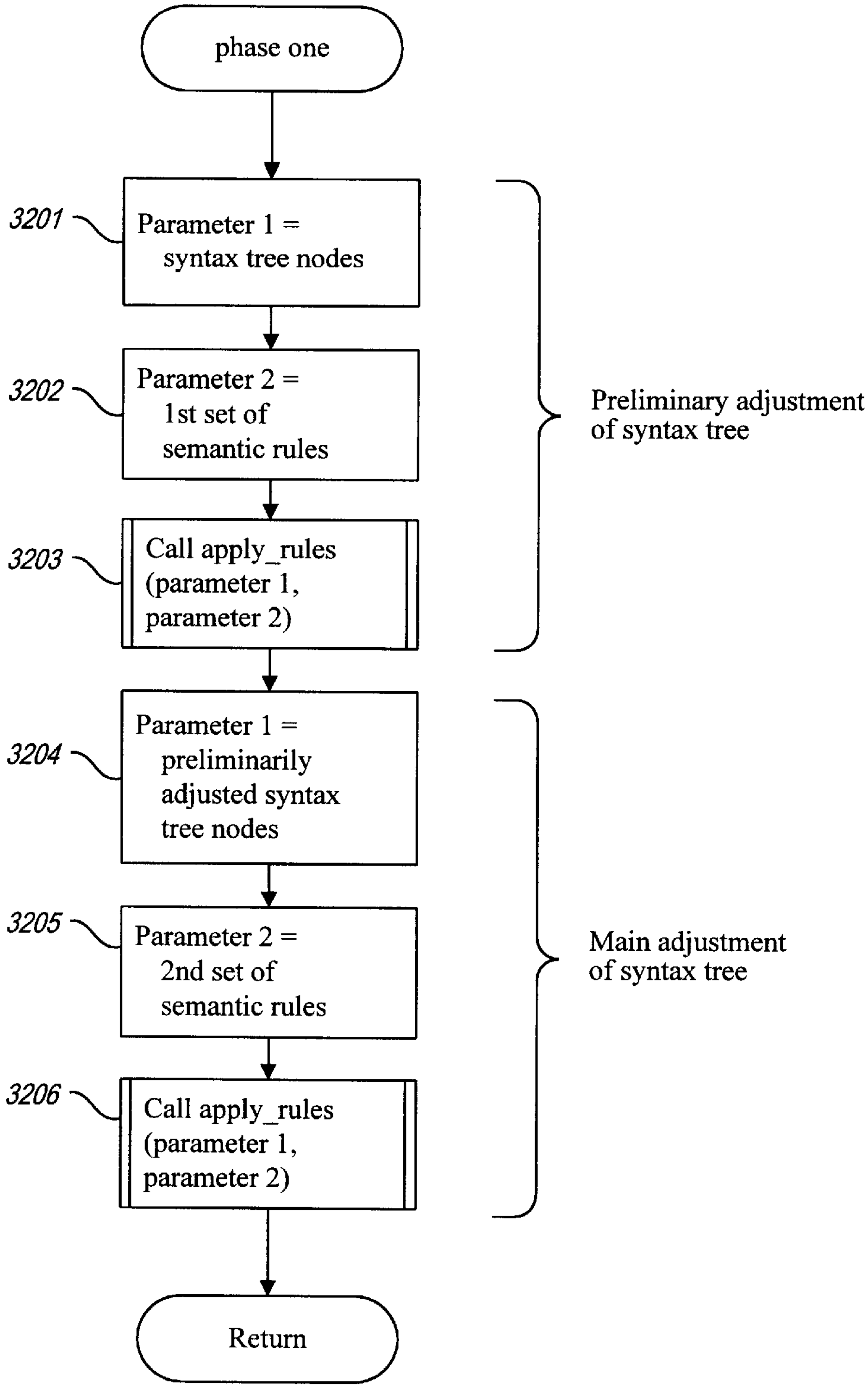

phase one
3201
Parameter 1 =
syntax tree nodes
3202
Parameter 2 =
1st set of
semantic rules
3203
Call apply_rules
(parameter 1,
parameter 2)
Preliminary adjustment
of syntax tree
3204
Parameter 1 =
preliminarily
adjusted syntax
tree nodes
3205
Parameter 2 =
2nd set of
semantic rules
3206
Call apply_rules
(parameter 1,
parameter 2)
Main adjustment
of syntax tree
Return

Fig. 32

1. SynToSem1: creates semantic nodes and a basic semantic graph in es
2. SynToSem2: creates the top-level semantic node and graph for fitted parses
3. SynToSem3: creates semantic nodes for a special subclass of elements in fitted parses

Fig. 33

Rule SynToSem1

If

the Syntax Record
has a Head and
there is no Subordinate Conjunction and
there is no Correlative and
there is no "It subject" and
there is no "There subject" and
there is no Ancestor of the Head for which it is true that that node
is the Emphatic of its Parent and is not a fraction and the head node
is not a verb and
if the segment is the Relative Pronoun of its Parent,
then there must not be a Nominal Relative on the Object of its Parent
and for all of its Parents last records there must not be a VPDone attribute and
if the lemma equals 'that'
then there must not be an Extra Position on the Parent of the Parent and
the node type is not "Auxiliary Phrase," "To Infinitive," "Determiner Phrase,"
or "Tag" or
there is a Possessive attribute or
there is an EVR attribute or
the Lemma equals "other" or
there are Coordinates and for all of those coordinates there is either a
Possessive attribute or an EVR attribute or the lemma is "other" and
if the node type is "Adverb Phrase"
then if the node type of Parent equals Prepositional Phrase
then the segment must not be the first of the Premodifiers of its Parent
and
either the Lemma must not be equal to 'well' or there must not be any Degree
attribute or there must not be any Weak Obligation on the Parent and
If the node type of the Head is a Conjunction or a Preposition,
then the segment node must not be a Conjunction of the Parent and the
segment node must not be a Preposition of the Parent and
If the node type is a Conjunctive Phrase
then there must not be any Coordinates of the Parent or there must not be a
Coordinate Conjunction attribute and
If the node type is a Quantifier Phrase,
then the Lemma of the Head must not be "no" and
If the word could have been an Interjection
then the node type must not be an Adverb Phrase or
there must be Premodifiers or
there must be no comma or
the segment must be the Post Adverbial of the Parent or
the number of Post Modifiers must be greater than one and
If there is an Intensifier attribute
then either the node type of Head of Parent is a "verb" or
the node type of Parent equals "fitted" or
there is an Adverbial Phrase attribute or
there is a WH marker and a Nominal Relative on the Parent and
If there is a Preposition attribute,
then there must be an Object of the Prepositional Phrase or
there is a Particle attribute on the Parent or
the word also could have been an Adverb and

Fig. 34A

Rule SynToSem1

If the Lemma is "also," "so," or "too,"
then there must not be a VPDone attribute on the Parent and
If the Lemma is "as" or "than"
then there must not be a Comparative on the Parent and
If the Lemma equals "for"
then there must not be a "for to" Preposition on the Parent and
If the Lemma equals "it"
then if there is a Topic Clause on the Parent
then the segment must be equal to the Subject of the Parent or
the segment must be equal to the Object of the Parent and
If the Lemma equals "it"
then the segment must not be in the Premodifiers of the Parent or
If there is an Extra Position on the Predicate Adjective of the Parent
then there must not be a Right Shift attribute on the Parent and
if there is a WH Question attribute on the Parent
then there is no "To Infinitive" attribute on the
Predicate Compliment of the Parent and it's
not the case that for any of the Post Modifiers of the
Parent that there is a "For to" prepositional phrase
on the first of the Premodifiers and
If the Lemma equals "let"
then the node type is not equal to "Adverb Phrase" and
If the Lemma equals "not"
then there must be a Coordinate Conjunction on the Parent and
If the Lemma equals "there"
then there must not be any Skipover attribute and
either there must not be any "Yes No" question on the parent or
there must not be a Copulative on the Parent or
there must be a T1 attribute on the Parent or
the first token integer must be greater than the first token integer of
the Subject of the Parent and
If the Lemma is "whether" or "whether or not"
then the node type of the Nominative Relative must not be an
Infinitive Clause" and
the Lemma must note be "etc," "etc.," "the," "hm," "mm," "uh," or "um"

Then

(If syntax node was kept, then create a corresponding semantic node.)
If the node type of the syntax node is a Noun Phrase and
there are Bases on the syntax node and
there is a Subject or an Object on the syntax node,
then make the Predicate equal to the Lemma of the first Basis of the syntax node
Else if there is a Proper Noun attribute on the syntax node and
if there is a dictionary entry for that word,
then make the Predicate equal to that dictionary entry
Else set the Predicate equal to the Lemma of the syntax node
If the word could have been a Verb and has a Present Participle attribute and
if for any of the Premodifiers of the syntax node there is a Possessive or
if the Lemma of the Preposition of the first of the Postmodifiers of the syntax node is
"by," "for," "of," or "to"

Fig. 34B

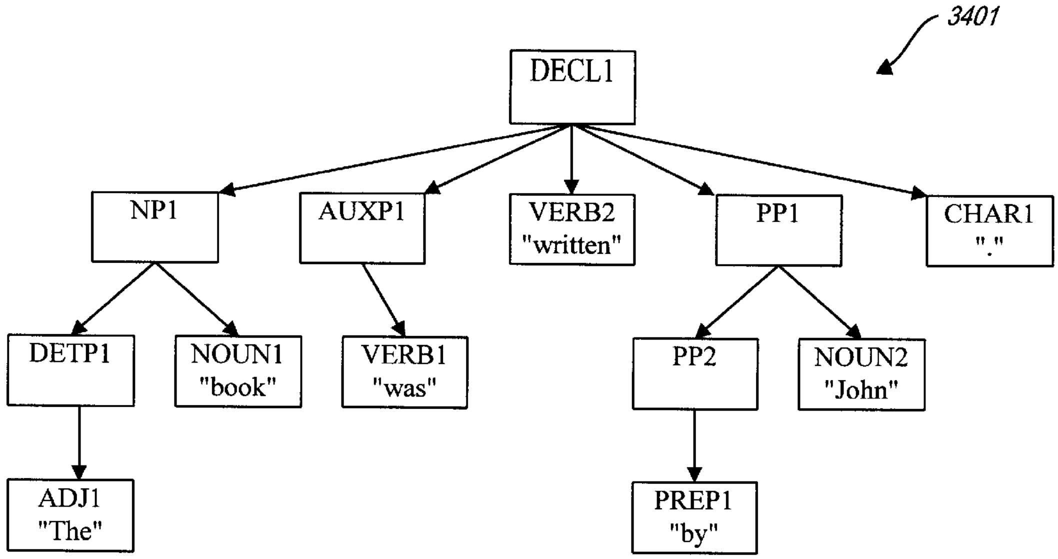
Rule SynToSem1

then make the Predicate equal to the Lemma of the Verb entry of the Part of Speech Record
Copy the appropriate fields from syntax node to the semantic node.
Go through each of the Premodifiers of syntax record and examine each Premodifier
    For each record of Premodifiers of the syntax record
        if there is a semantic node on the record and
        if the semantic node of the record is not in the temporary modifiers attributes of this semantic record and there is no Skipover attribute on the record and the record is not equal to the Preposition of the Parent of the record and the record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record, or Coordinate Subordinate Clauses
        then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record
    For each record of the Postmodifiers of the syntax record
        if there is a semantic node on record and
        if the semantic node of record is not in the Temporary Modifiers attributes of this semantic record and there is no Skipover attribute on record and record is either not in the Coordinates of syntax record or there is a Coordinate of the Prepositional Phrase on syntax record or Coordinate Subordinate Clauses
        then add the Semantic node of the record to the Temporary Modifiers attribute on this semantic record
    If there are Coordinates of the syntax record and no Coordinates of the Prepositional Phrase on that syntax record and no Coordinate Subordinate Clauses
    then
        for each of the Coordinates of syntax record
            if there is a Semantic node on record,
            then add that Semantic node to Coordinates attribute on this new Semantic record.

Fig. 34C

Sentence represented by syntax parse tree: "The book was written by John."

Syntax tree prior to application of rule SynToSem1:



Rule SynToSem1:

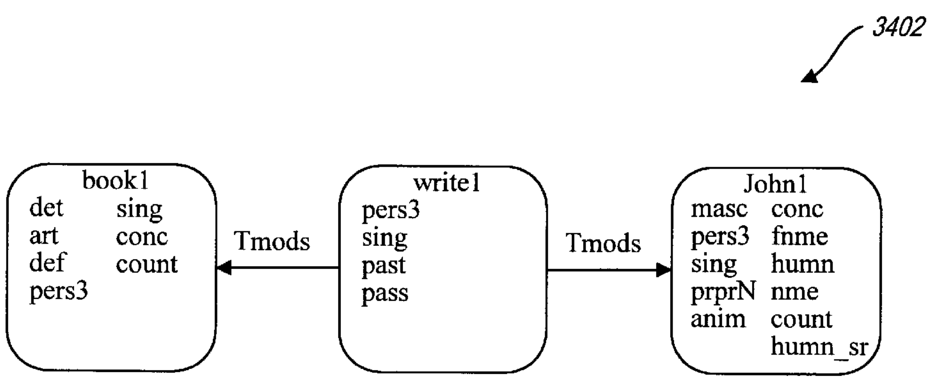


Fig. 34D

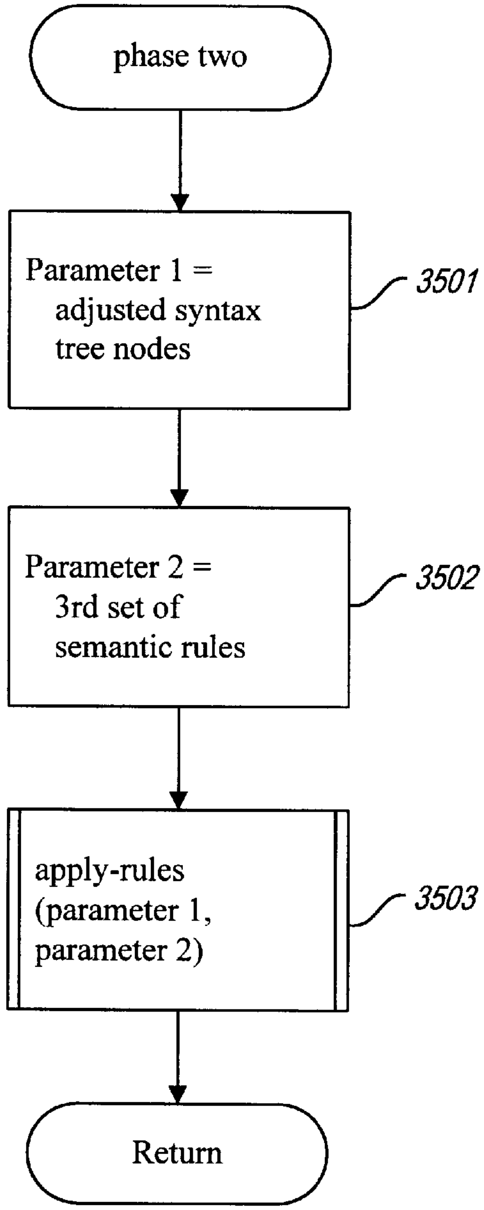

phase two
Parameter 1 = adjusted syntax tree nodes
3501
Parameter 2 = 3rd set of semantic rules
3502
apply-rules (parameter 1, parameter 2)
3503
Return

Fig. 35

1. LF_Dsub1: creates the Dsub (deep subject) label for subjects of clauses in the active voice

2. LF_Dsub2: for passive-voice clauses, if there is a "by"-PP, identifies this PP as the Dsub of the action

3. LF_Dobj1: creates the Dobj (deep object) label for, e.g., direct objects of clauses in the active voice

4. LF_Dobj2: for passive clauses, identifies the syntactic subject as the deep object of the action

5. LF_Dobj3: for clauses like "The door opened," identifies "the door" as the logical object of the action

6. LF_Dobj4: for constructions like "the nomination of the candidate," identifies "the candidate" as the logical object of an action of nominating

7. LF_Dind1: creates the Dind (deep indirect object) label for, e.g., "Mary" in "John gave Mary the book"

8. LF_Dind2: identifies the deep indirect object ("Mary") in paraphrases like "John gave the book to Mary"

Fig. 36

9. LF_Dind3: chooses the right deep indirect object in trickier constructions like "The book was given her"; "She was given the book"

10. LF_Dnom: creates the Dnom (deep nominative) label for predicate nominative, e.g., "our friends" in "They are our friends"

11. LF_Dcmp1: identifies the complement ("president"; "italic") in, e.g., "elect Tom president"; "make the word italic"

12. LF_Dcmp2: identifies the complement in trickier constructions, e.g., in "He gave Tom a place to call his own," "his own" is the Dcmp of "call"

13. LF_Dadj: creates the Dadj label for predicate adjectives, e.g., "blue" in "The sky is blue"

14. LF_CausBy: creates a causative relation where appropriate, e.g., "why" in "Why did you say that?"

15. LF_LocAt: creates a locative relation where appropriate, e.g., "where" in "Where did you find that?"

16. LF_TmeAt: creates a temporal relation where appropriate, e.g., "what day" in "What day did you read that?"

17. LF_Manr: creates a manner relation where appropriate, e.g., "how" in "How did you do that?"

Fig. 37

18. LF_Ptcl: creates a Ptcl node to refer to particles in phrasal verb constructions

19. LF_PrpCnjs: creates temporary relations for PPs and subordinate clauses by naming these elations with the word that is the preposition or conjunction

20. LF_PrpCoord: handles cases of coordinated PPs or subordinate clauses

21. LF_Props: lists remaining clausal adjuncts for any given node

22. LF_Ops: identifies logical operators in noun phrases, e.g., "all" in "all my children"

23. LF_Nadj: lists remaining adjectives that premodify nouns

24. LF_Mods: lists remaining non-clausal modifiers for any given node

Fig. 38

Rule LF_Dobj2

If the Semantic Record

doesn't already have a Deep Object,

and has a Passive attribute,

and has a Subject on its syntactic record (SynNode), and this Subject (which is a syntactic record) has a SemNode attribute (i.e., it has a corresponding semantic record)

and there are no Coordinates

and if there is a Predicate Complement attribute on its syntactic record, then the node type is not "COMPCL" (i.e., it is not a complement clause, as in: "some people were convinced <u>that he had written a book</u>"

and if the SynNode record has either a D5, D6, ObjC, or Psych feature[2]

then either the Object of the SynNode is not a noun phrase,

or the SynNode has an X1[3] feature (as in: He was named Arles")

or the Object of the SynNode has an Animate feature

or there is a Case feature on the Object of the SynNode and its Lemma is not "it"

Then,

give the Semantic record a Dobj attribute with, as its value, the semantic record corresponding to the Subject on the syntactic record

and, remove what is now the value of Dobj attribute from the list of Tmods

---

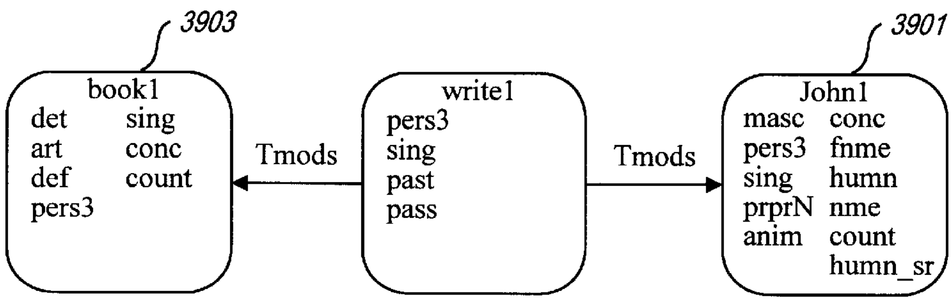
[2] D5, and D6 are features from Longman's Dictionary of Contemporary English; ObjC is verb subcategory for verbs which show object control (e.g., I want Harry to wash the car) and Psych is a verb subcategory for verbs like "scare" "excite".

[3] X1 is a feature from Longman's Dictionary of Contemporary English.

Fig. 39A

Sentence represented by the logical form: "The book was written by John."

Logical form prior to application of rule LF_Dobj2:



Rule LF_Dobj2:

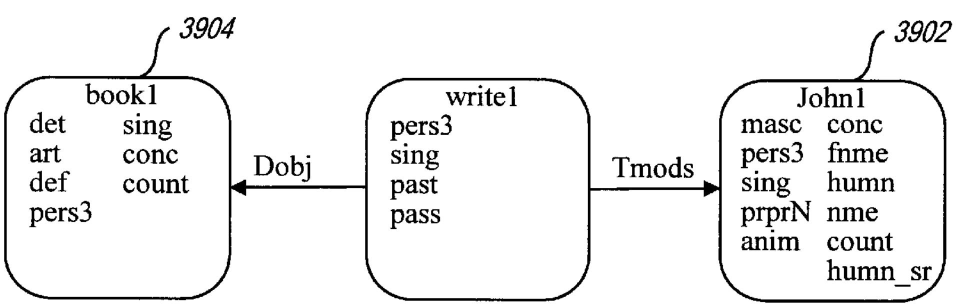


Fig. 39B

1. PsLF_RelPro: identifies proper referents for relative pronouns, e.g., "who" refers to "the man" in "the man who came to dinner"
2. PsLF_ReciprocalAnaphora: handles reciprocal pronouns like "each other" and "one another"
3. PsLF_ReflexiveAnaphora: handles reflexive pronouns like "myself, yourself, him/herself," etc.
4. PsLF_PronAnaphora: identifies possible NP referents for most pronouns
5. PsLF_ProtoAnaphora: handles special cases of pronouns which can agree with just about any NP
6. PsLF_NumberEllipsis: handles reference for number words, e.g., "A bird in the hand is worth two (birds) in the bush"
7. PsLF_FillInHead: adds "DUMMY" as a head word in special cases of unclear referents
8. PsLF_NumberCritique: takes note of pronouns that disagree in number with their referents
9. PsLF_FillDsub: fills in "x" as a placeholder for the deep subject in cases where that is missing, e.g., in passives like "The door was opened"
10. PsLF_UnifyProns: if two pronoun nodes refer to the same referent, this rule unifies them
11. PsLF_UnifyCopies1: unifies some nodes that should be identical
12. PsLF_UnifyCopies2: unifies other nodes that should be identical
13. PsLF_RaiseModality: deletes some verbs when they serve only an aspectual purpose, e.g., in "We used to go there," "used to" is deleted from the graph
14. PsLF_RaisePcs: makes fitted parses easier to read

Fig. 40

Rule PsLF_PronAnaphora

If the Semantic Record

has a Pers3 attribute, i.e., it is not either first (e.g., I or we) or second person (e.g., you) and the node type of the head of its syntactic record is either "PRON" (pronoun) or the node type of the head of its syntactic record is "ADJ" (adjective) and it has a possessive attribute

and is not Reflexive

and none of the premodifiers of the Parent of its syntactic node has the Lemma "own"

and the Pred of this semantic record is not "each other" or "one another"

and does not have NonRef attribute (NonRef is an attribute set on words that cannot have a reference, such as true numbers, as in: One plus one is two.

and does not have a Negation attribute

and if it has an Indefinite attribute, then there must also be a Definite attribute

and is not a Wh- word (it does not have a WH attribute)

and is not a Relative

and is not a Distal (Distl) or a Proxal (Proxl) determiner (e.g., "this" "that")

Then

add a FindRef attribute to the semantic record

for each of the records in the list of possible referents;[1]

if

the possible referent has a corresponding semantic record

and the possible referent is not the same as this record (i.e., the antecedent of a noun phrase can not be the noun phrase itself)

and if the head of both the possible referent and of this record's SynNode are pronouns (i.e., have the node type "PRON" as their head), then the possible referent must precede this record (no forward reference to a pronoun; an example of forwards (cataphoric) reference is: with his hat on, the teacher left the room, where "his" refers forward to "teacher"

and if the possible referent is the ancestor of the syntactic record of this record, then that ancestor must have a Prp attribute (i.e., must have a postmodifying Prepositional phrase), and its preposition must be either "in", "to", "for", or "by"

and there is no Time or Space feature on the possible referent

and this record and the possible referent agree in number

and this record and the possible referent agree in gender

and if the Lemma of the SynNode is "they" and the possible referent can be a Mass noun (i.e., the possible referent has a Mass feature),

then the possible referent must also be a Count noun (i.e., it must also have a Count feature).

and if the Lemma of the SynNode is "they" and the possible referent has a Sing feature (can be Singular), and the possible referent does not have a Plur feature (i.e., it cannot be Plural),

then the possible referent is either a Count noun, or the possible referent is a Coordinated noun phrase, or it has a Universal feature, or the possible referent is indefinite and has no possessive, or the possible referent has a Proxal feature,

---

[1] this list is created in a PrLf rule, so, after syntactic processing but before most logical form processing (it is a list of syntactic records). This is a list of all the words in the sentence which can be referred to, i.e., most of the nouns and pronouns in the sentence

Fig. 41A

Rule PsLF_PronAnaphora

and if there is an ancestor of the possible referent that has a Coords attribute (i.e., has coordinate constituents) (but before there is an ancestor with a Subject attribute) then this ancestor is the same as the ancestor of this record that has a Coords attribute (but before there is an ancestor with a Subject attribute)

then if this record is a possessive (e.g., "his" in "John saw his son")

add the possible referent to the list of possible referents (the value of the Refs attribute)

if:

the possible referent is a genitive

and node type of the head of the possible referent is not a Noun

and the possible referent precedes this record (i.e., the semantic record being processed in this rule

or if:

the possible referent is not the first of this record's Parents

and the first of the Parents of the possible referent is not the first of this record's Parents

and if the possible referent follows this record and if any of the possible referent's ancestors have Coordinate constituents, then there should be no ancestor of this record for which the Parent has Coordinate constituents and for which the Parent is the same as the ancestor of the possible referent that has Coordinate constituents (but before there is ancestor whose node type is "NP")

or else if the node type of Parent of this record's syntactic record is "TAG" (i.e., if the pronoun is in a tag question)

add the possible referent to the list of possible referents (the value of the Refs attribute)

if:

the possible referent is the Subject of the Parent of the Parent of this record (e.g., "they" refers to "someone" in: Someone painted in here, didn't they?)

or else:

if

this record is a prepositional phrase

and this record precedes the Subject of this record's Parent

and the possible referent is the Subject of this record's Parent

then add the possible referent to the list of possible referents (the value of the Refs attribute);

else if

this record is not possessive

and this record precedes the possible referent

and node type of the head of the possible referent is "NOUN" and is not a Dummy noun (i.e., one that cannot be a possible referent)

and if this record is not one of possible referent's ancestors

and if it is not the case that there is an ancestor of this record that has Coordinate constituents and the Lemma of that ancestor is "but" and that ancestor is also an ancestor of this record which has Coordinate constituents

then add the possible referent to the list of possible referents (the value of the Refs attribute)

Fig. 41B

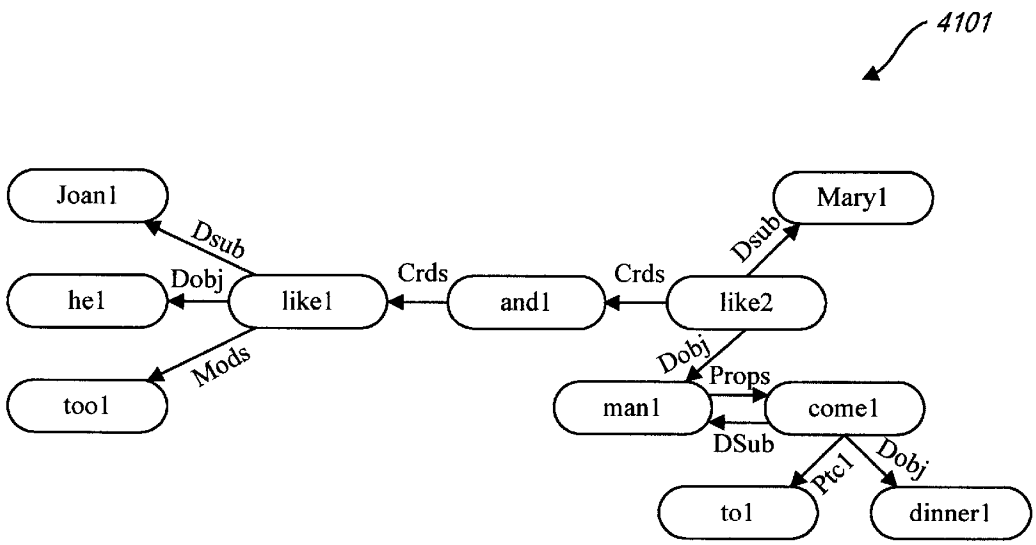
Rule PsLF_PronAnaphora

else if
    the possible referent is a Prepositional Phrase
    and the Parent of the possible referent is not the Parent of this record's syntactic record
    and if the Parent of the possible referent is an Adjective Phrase, then the Parent of the possible referent precedes this record
then add the possible referent to the list of possible referents (the value of the Refs attribute)
else if
    there is no ancestor of the possible referent for which the Lemma is "be" (but before there is an ancestor with a Subject) that is the same as the ancestor of this record for which the Lemma is "be" (but before there is an ancestor with a Subject)
    and none of the Parents on the semantic record of the possible referent is the same as the possible referent
    and if this record precedes the possible referent, then the Head of the possible referent is not either a Noun or an Adjective
then add the possible referent to the list of possible referents (the value of the Refs attribute)
if the possible referent was added to the list of possible referents (the value of the Refs attribute)
then add of RefOf attribute to the possible referent and add this record to that list (provide cross pointers: this record gets a Ref attribute pointing to possible referents, and the possible referents each get a RefOf attribute, pointing back to this record.

Fig. 41C

Sentence represented by logical form: "Mary likes the man who came to dinner, and Joan likes him too."

Logical form prior to application of rule PsLF_PronAnaphora:



Rule PsLF_PronAnaphora:

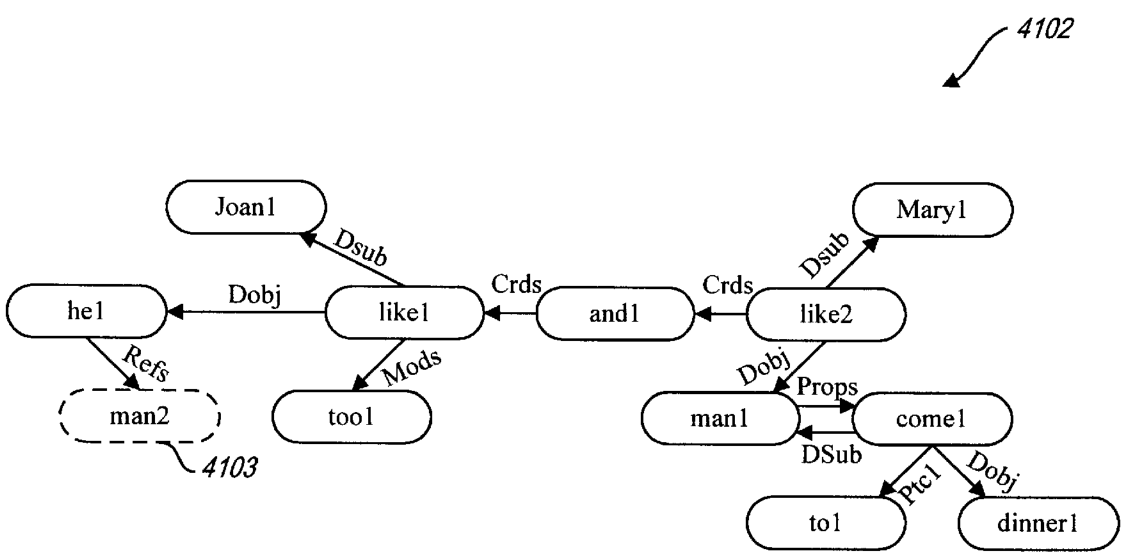


Fig. 41D

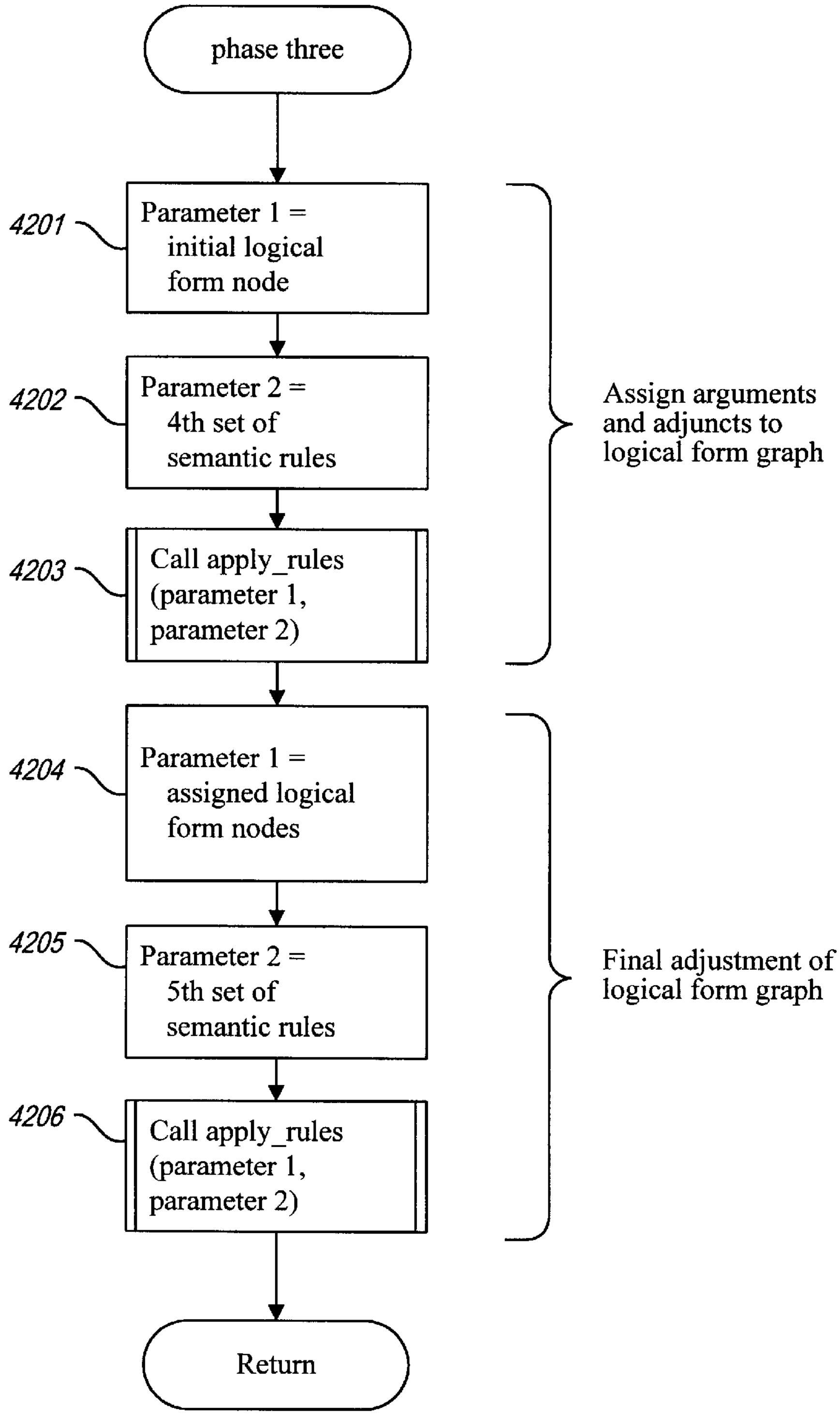

phase three
4201
Parameter 1 = initial logical form node
4202
Parameter 2 = 4th set of semantic rules
4203
Call apply_rules (parameter 1, parameter 2)
Assign arguments and adjuncts to logical form graph
4204
Parameter 1 = assigned logical form nodes
4205
Parameter 2 = 5th set of semantic rules
4206
Call apply_rules (parameter 1, parameter 2)
Final adjustment of logical form graph
Return

Fig. 42

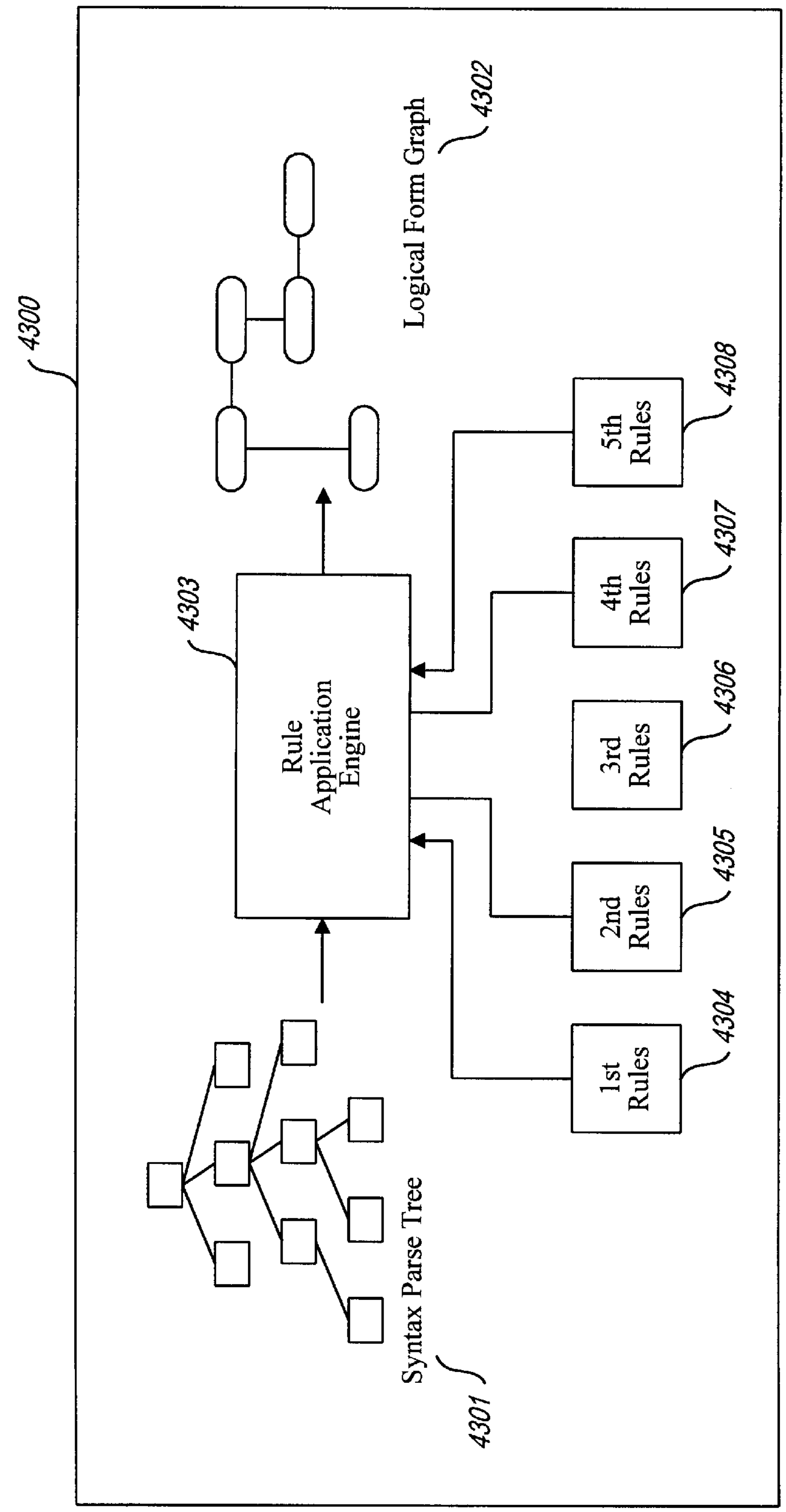

4300
Logical Form Graph
4302
Rule Application Engine
4303
Syntax Parse Tree
4301
1st Rules
4304
2nd Rules
4305
3rd Rules
4306
4th Rules
4307
5th Rules
4308

Fig. 43

Rule: TrLF_LongDist1 modifies RELCL1 ("whom I met")

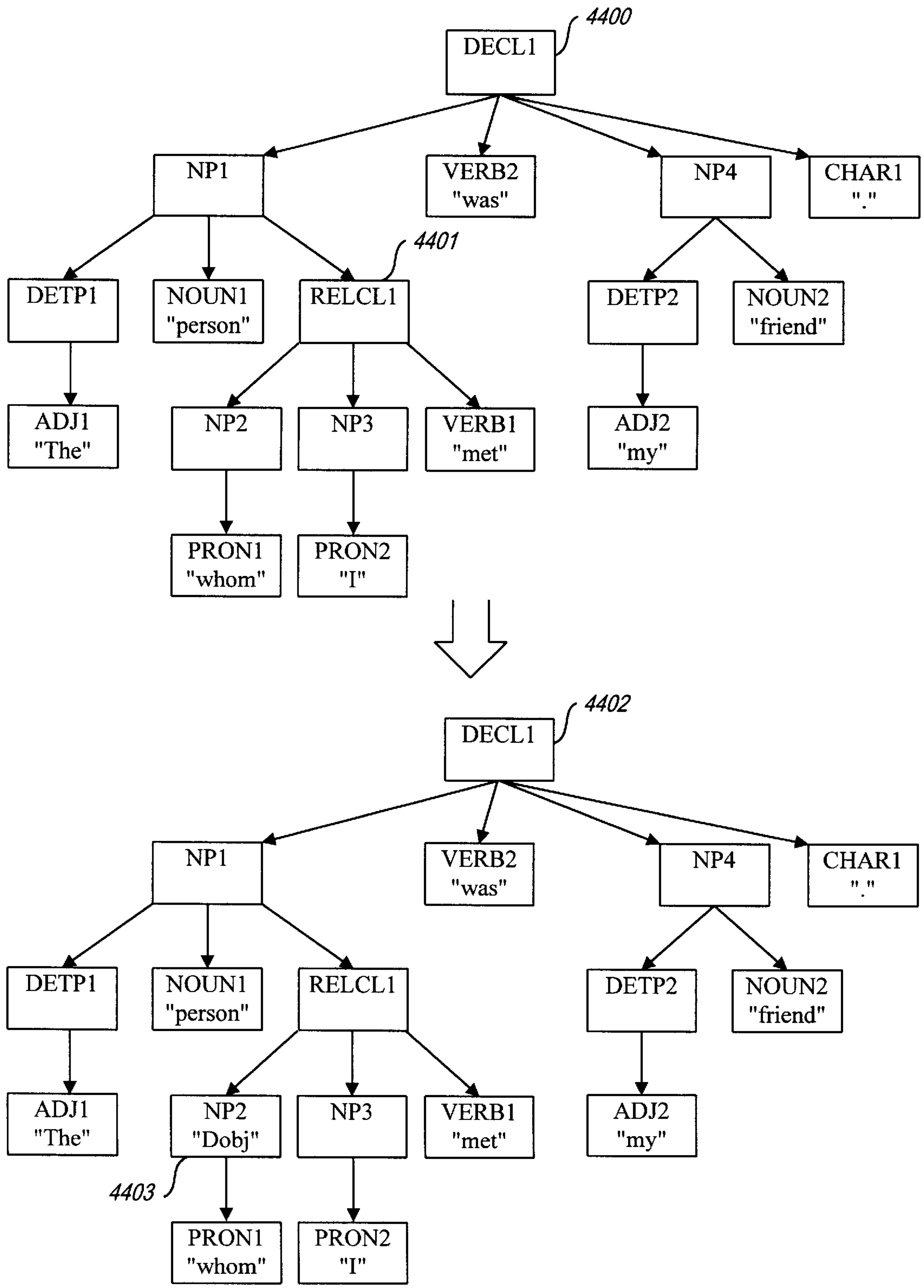
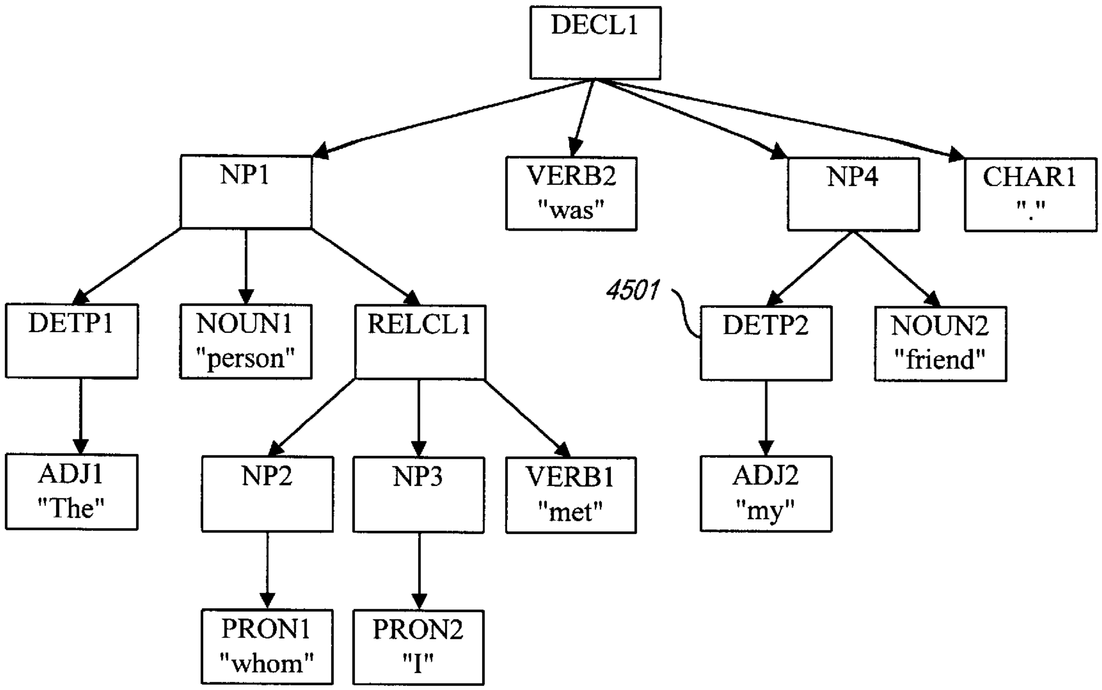


Fig. 44



Rule: SynToSem1 produces logical form graph node from DETP2 ("my")

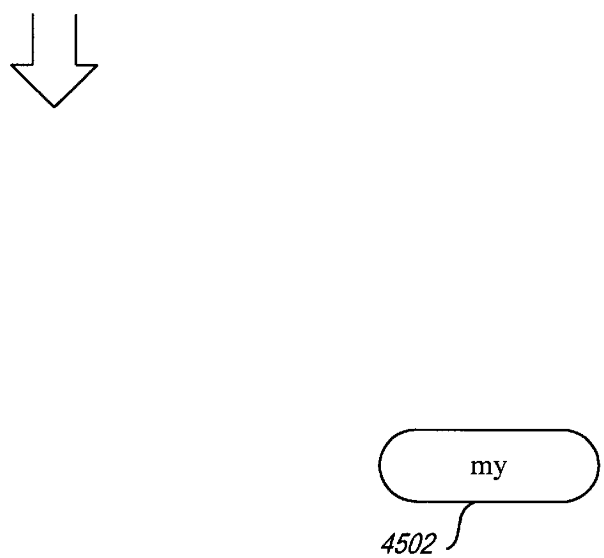
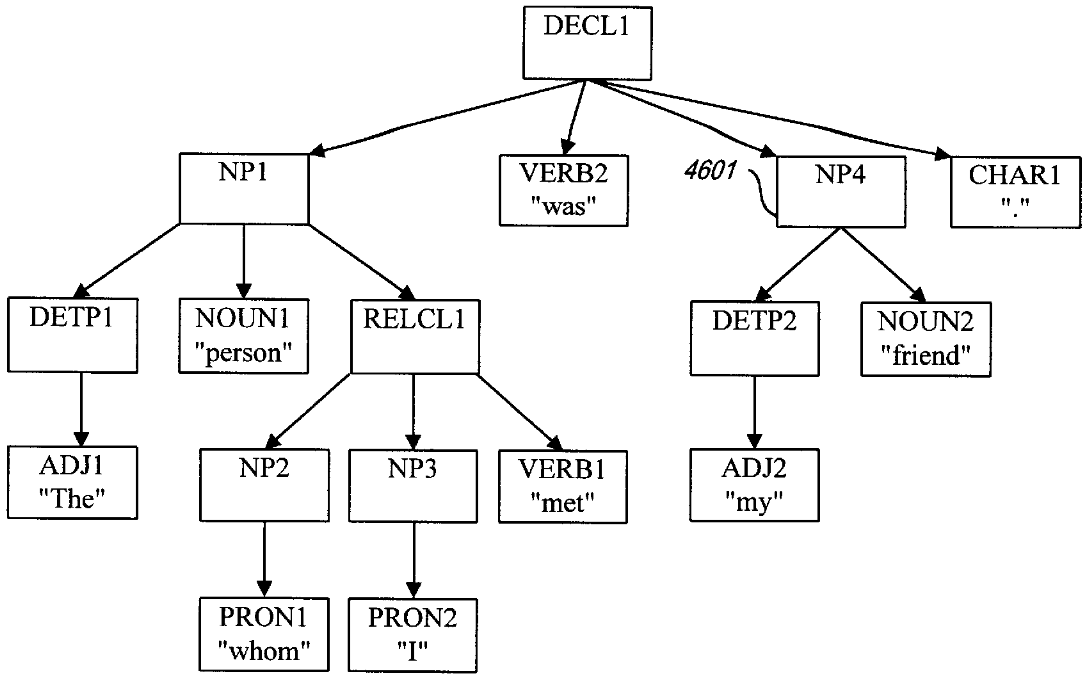


Fig. 45



Rule: SynToSem1 produces logical form graph node "friend" from NP4 ("my friend")

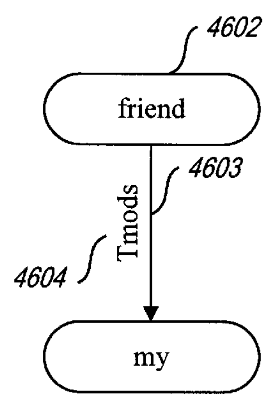
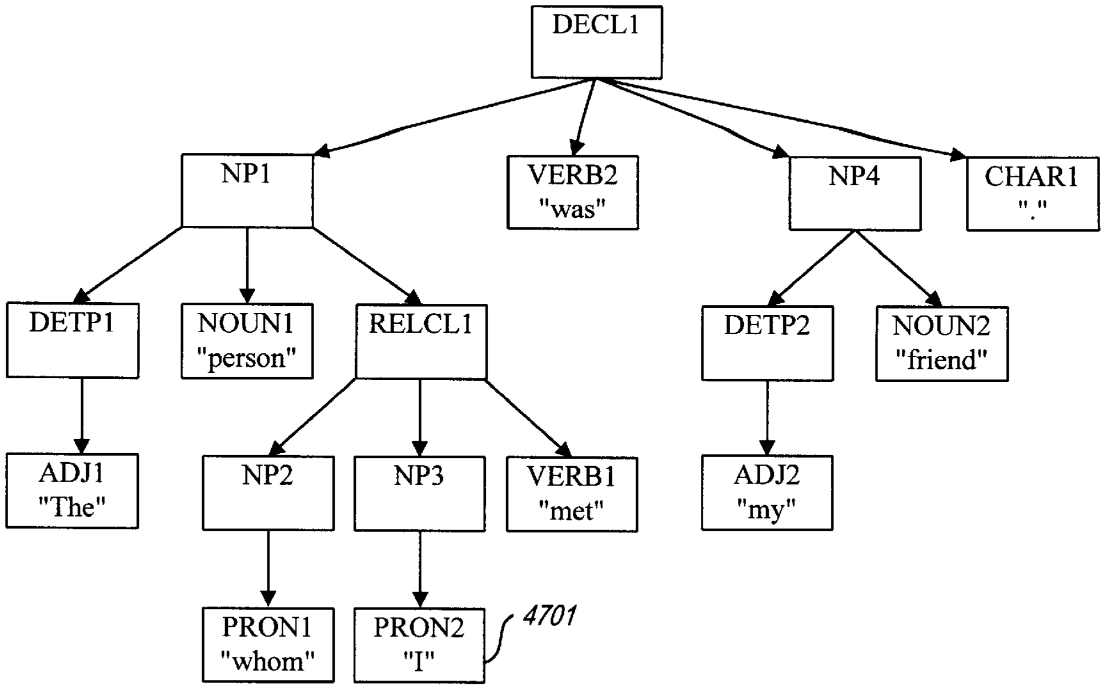


Fig. 46



Rule: SynToSem1 produces logical form graph node "I" from NP3 ("I")

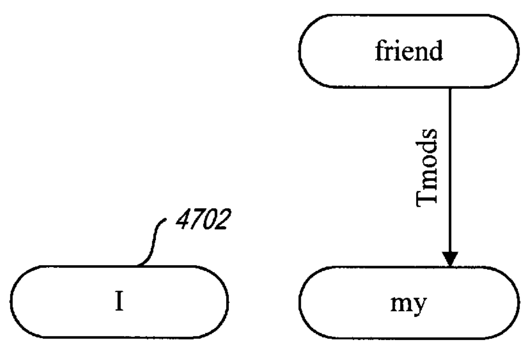
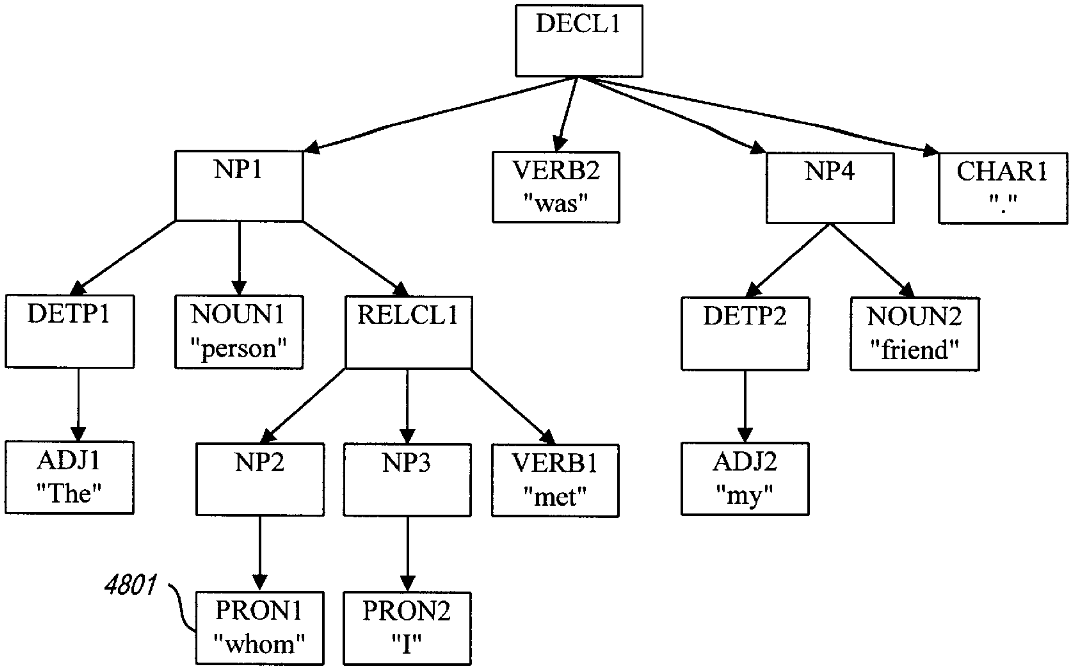


Fig. 47



Rule: SynToSem1 produces logical form graph node "whom" from NP2 ("whom")

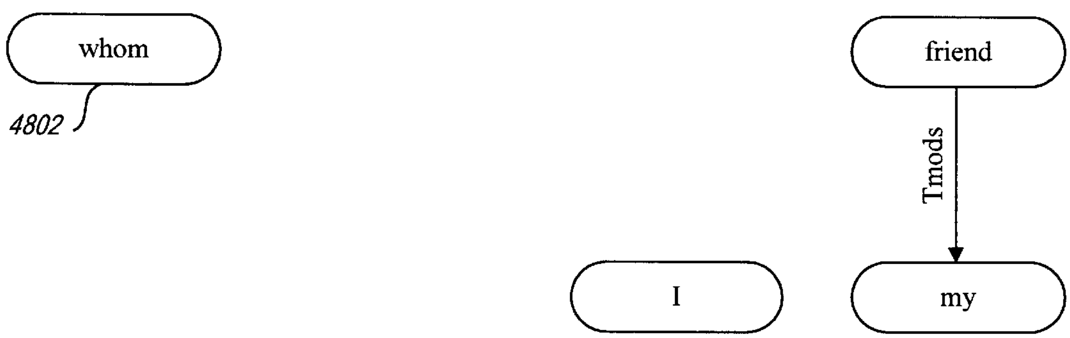
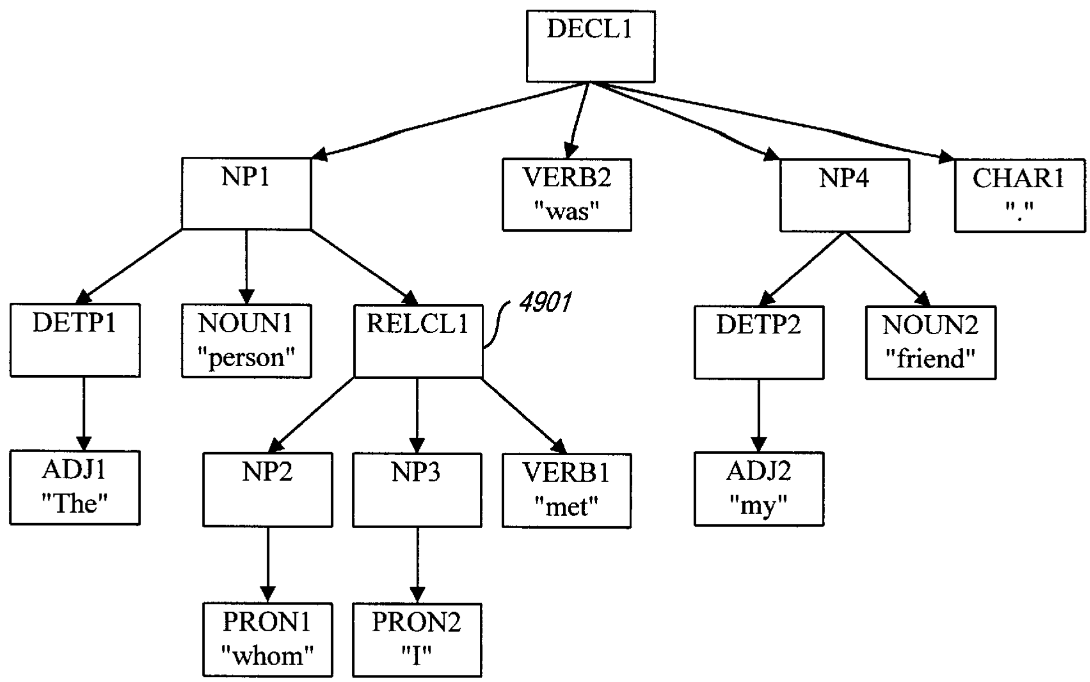


Fig. 48



Rule: SynToSem1 produces logical form graph node "meet" from RELCL1 ("whom I met")

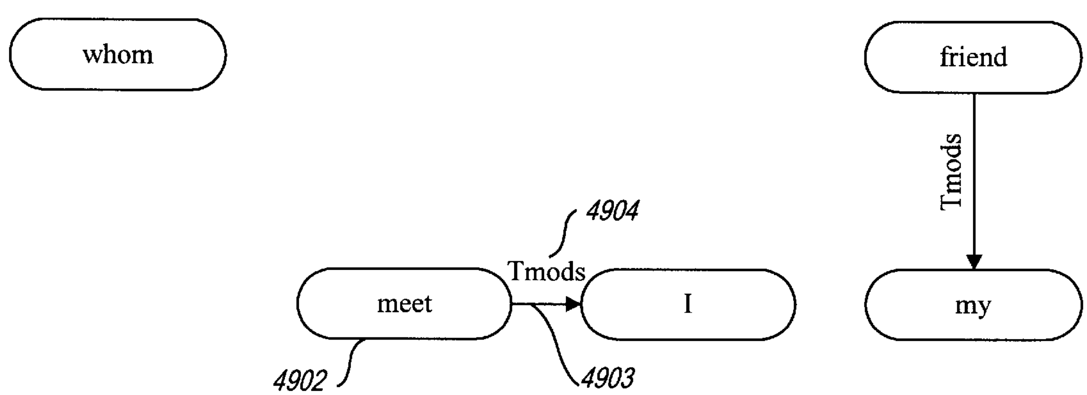
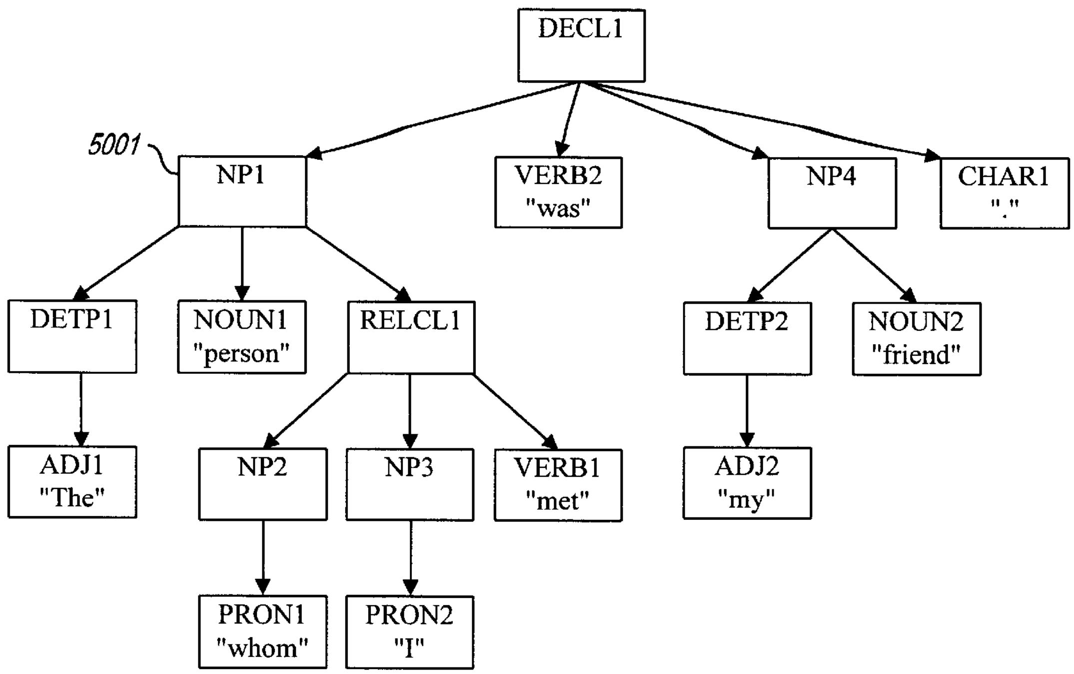


Fig. 49



Rule: SynToSem1 produces logical form graph node "person" from NP1 ("The . . . met")

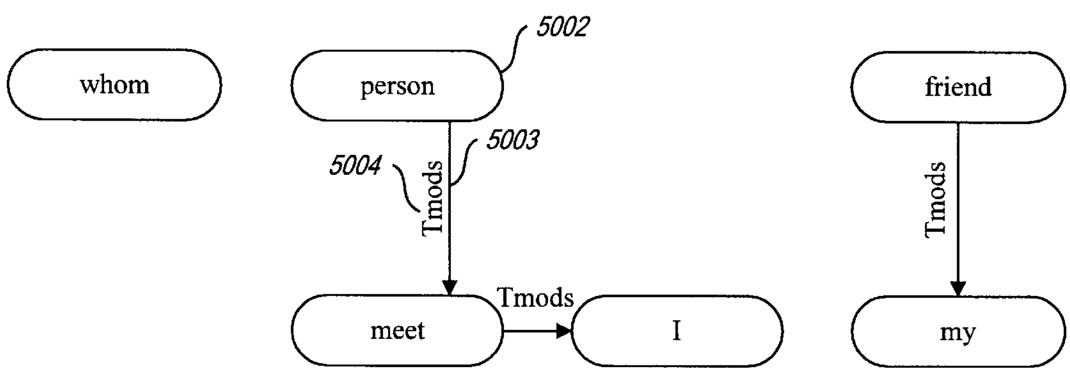
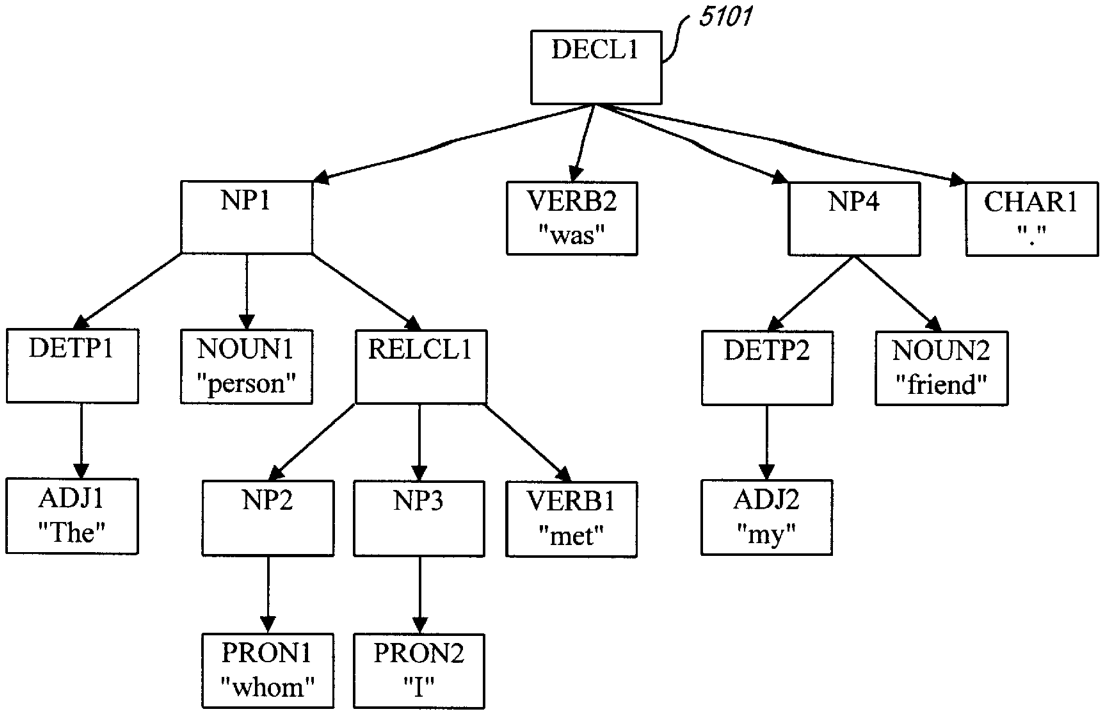


Fig. 50



Rule: SynToSem1 produces logical form graph node "be" from DECL1

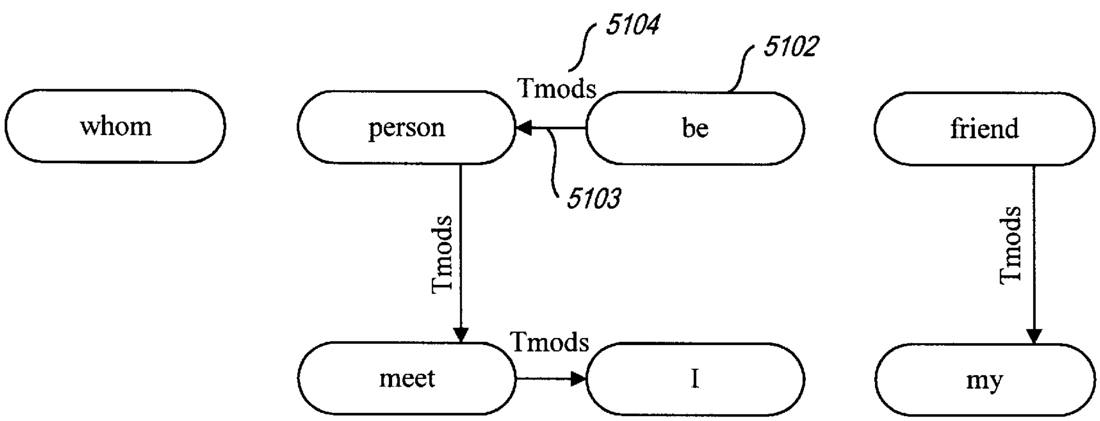
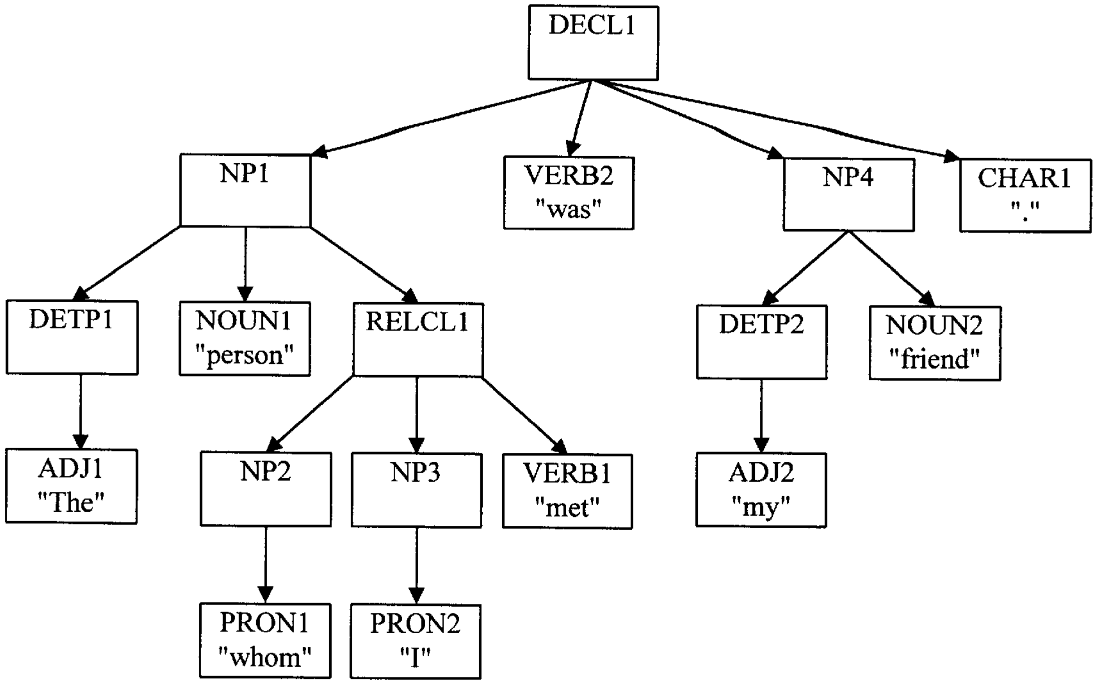


Fig. 51



Rule: LF_Dsub1 with node "be" labels link and creates another link

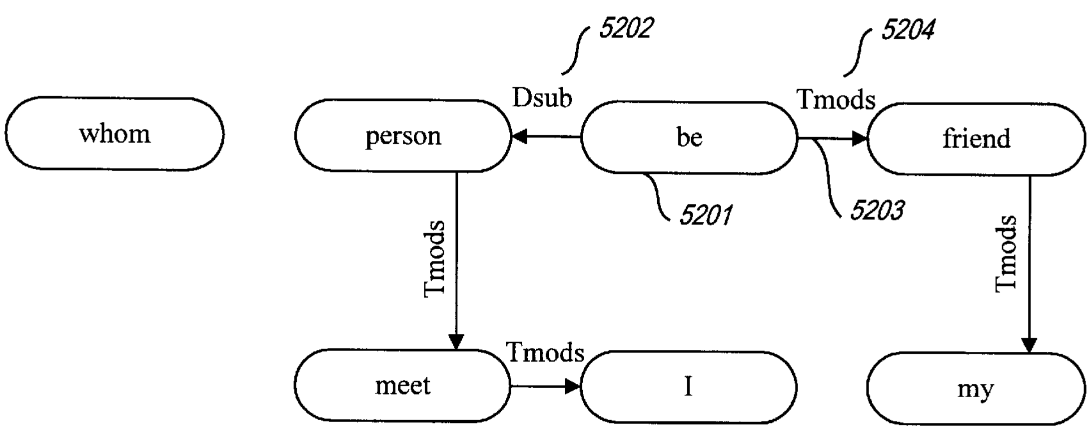
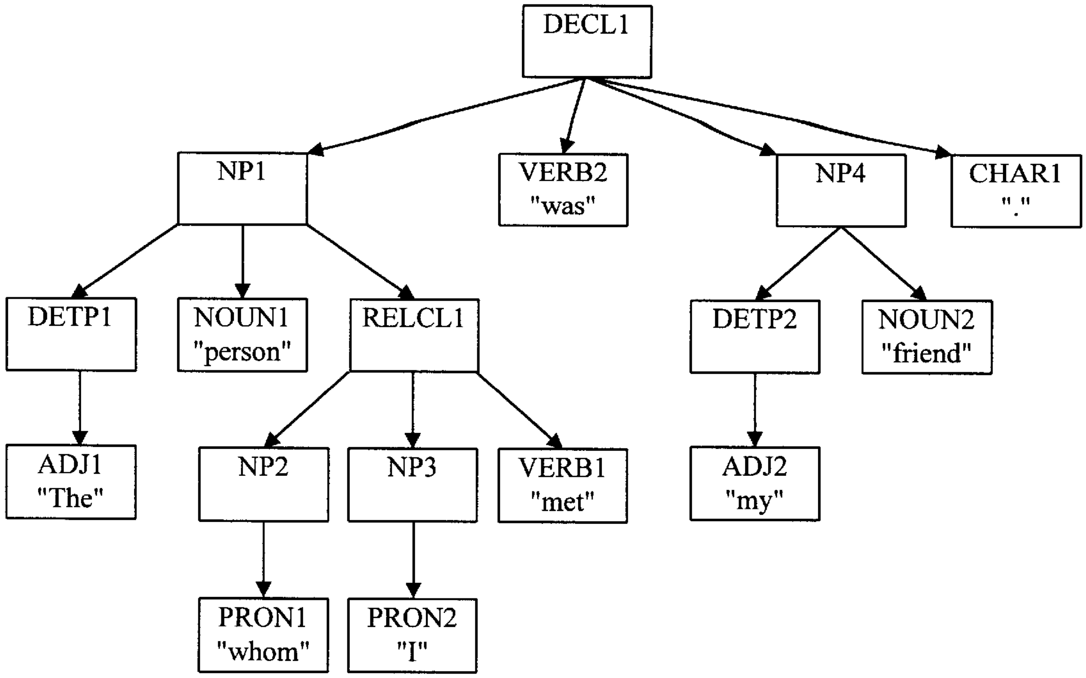


Fig. 52



Rule: LF_Dnom with node "be" labels link

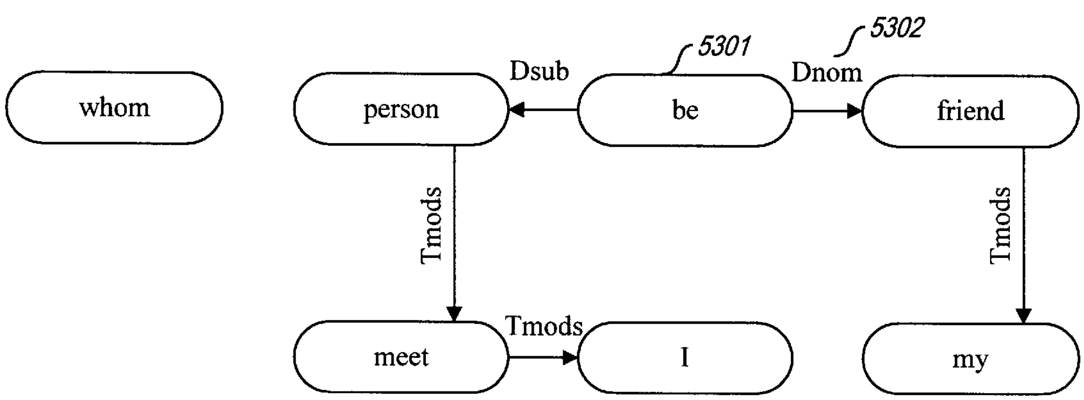
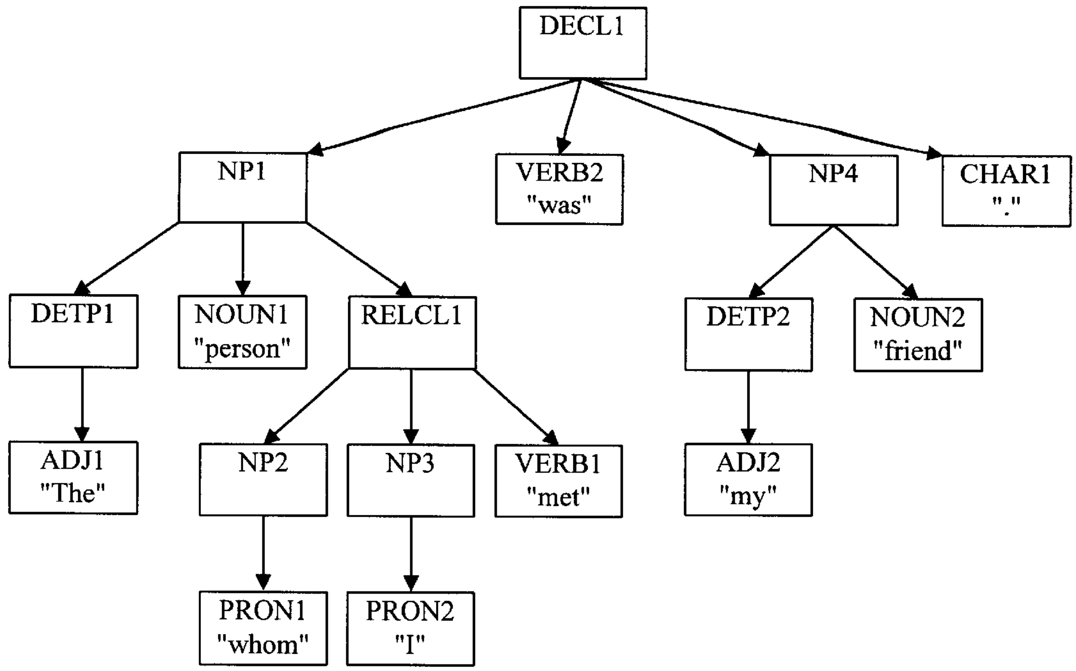


Fig. 53



Rule: LF_Props with node "person" labels link

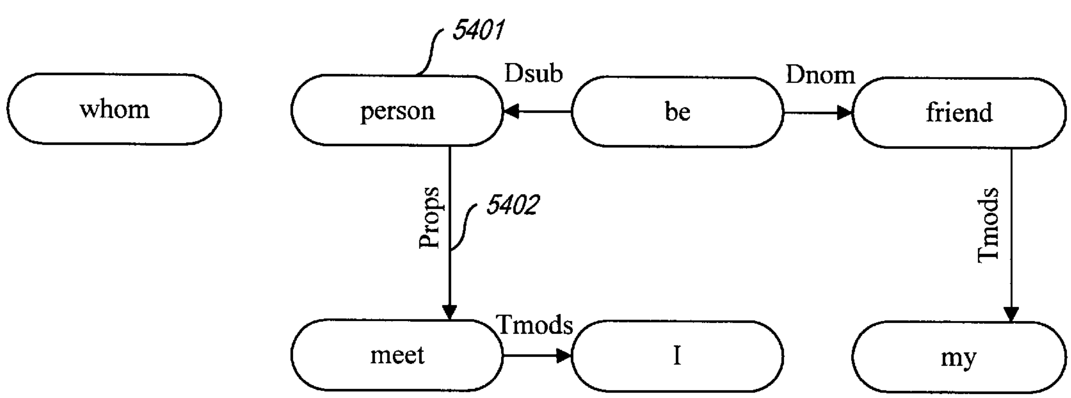
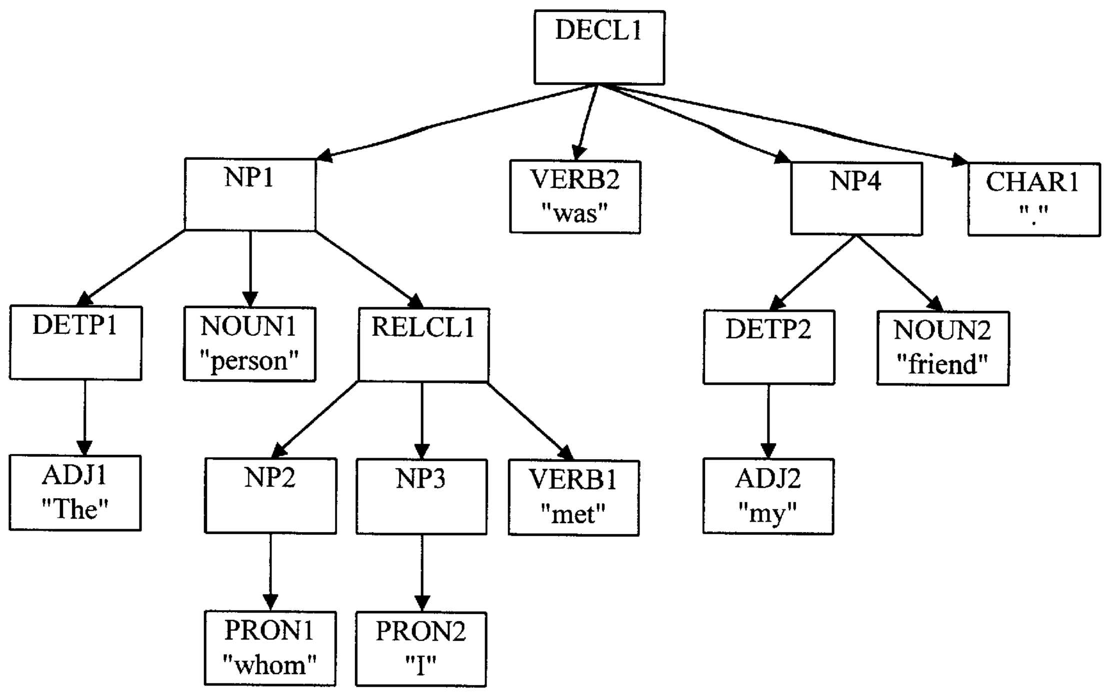


Fig. 54



Rule: LF_Dsub1 with node "meet" labels link

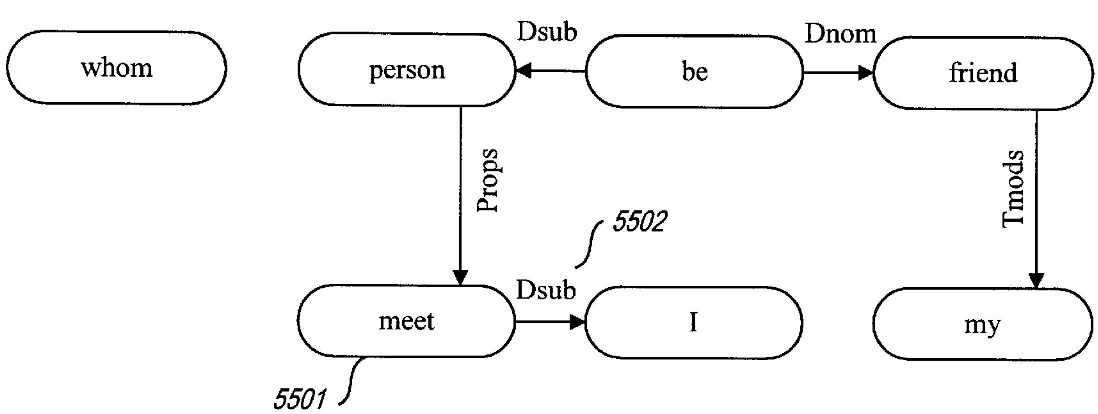
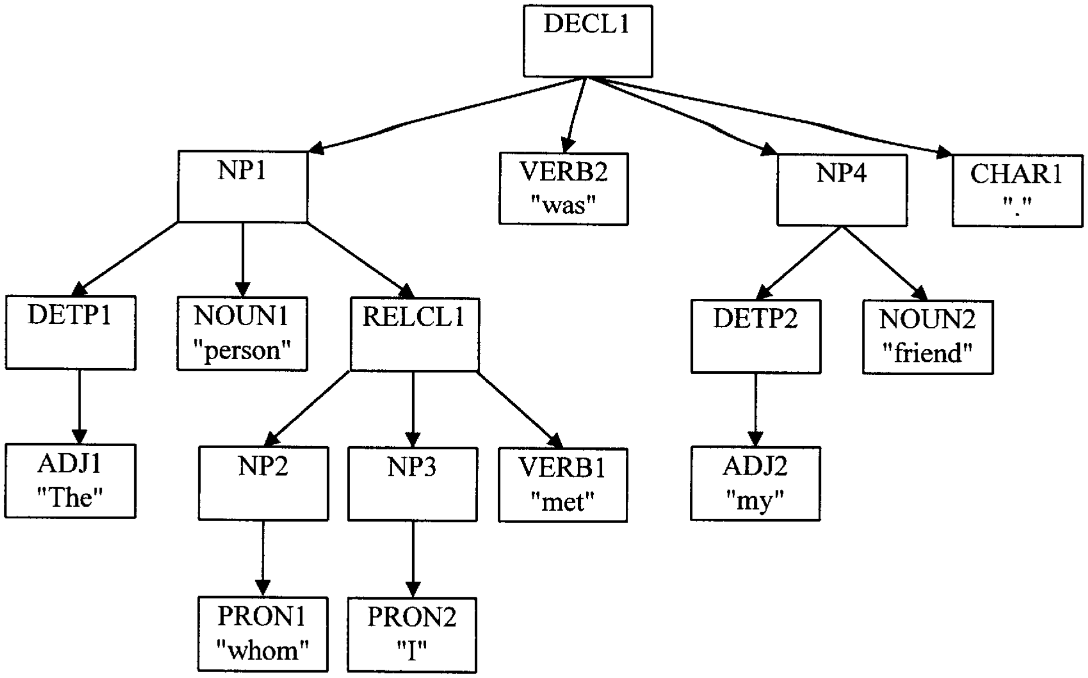


Fig. 55



Rule: LF_Dobj1 with node "meet" adds link and labels it

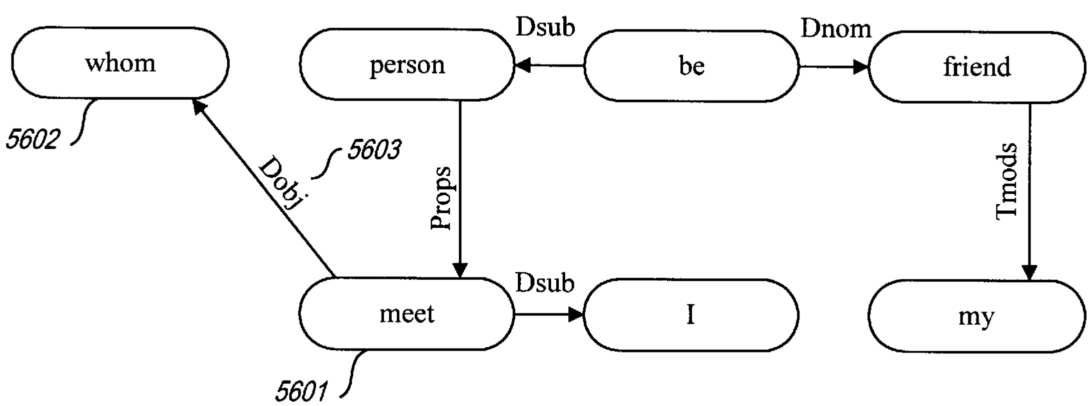
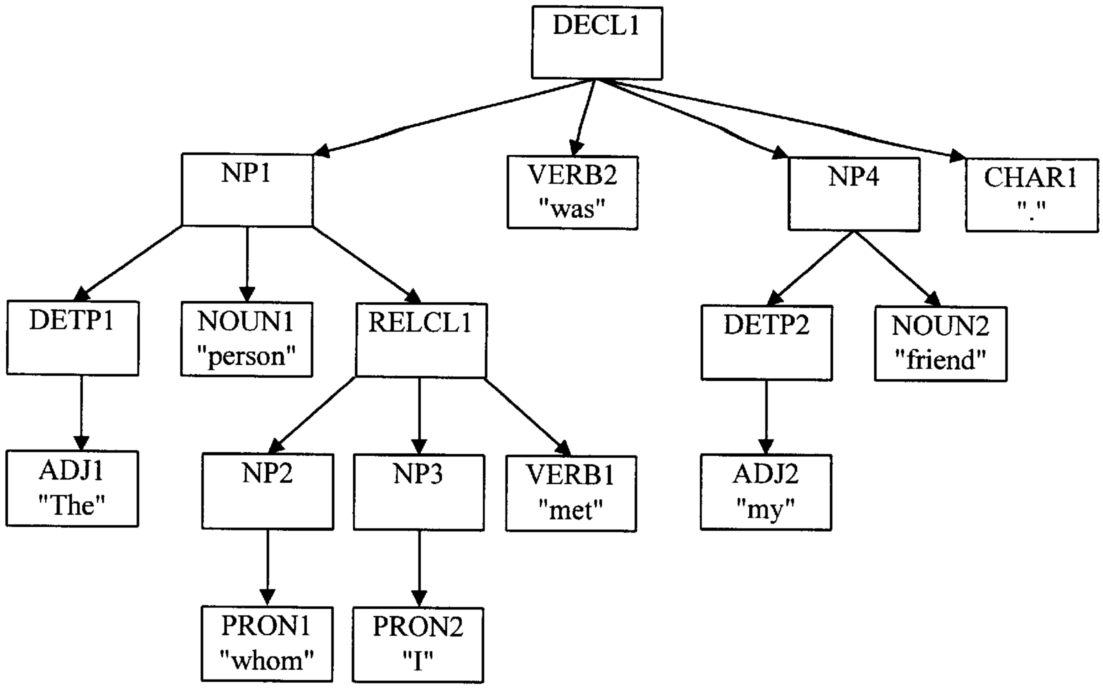


Fig. 56



Rule: LF_Ops with node "friend" labels link

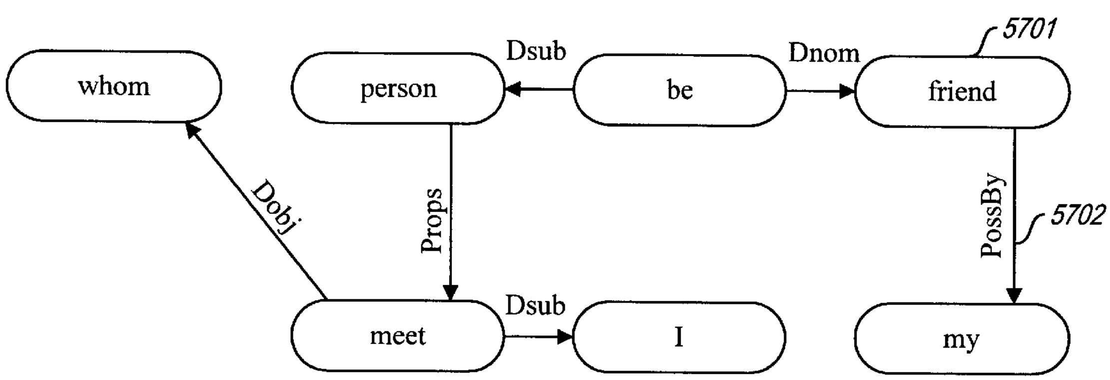
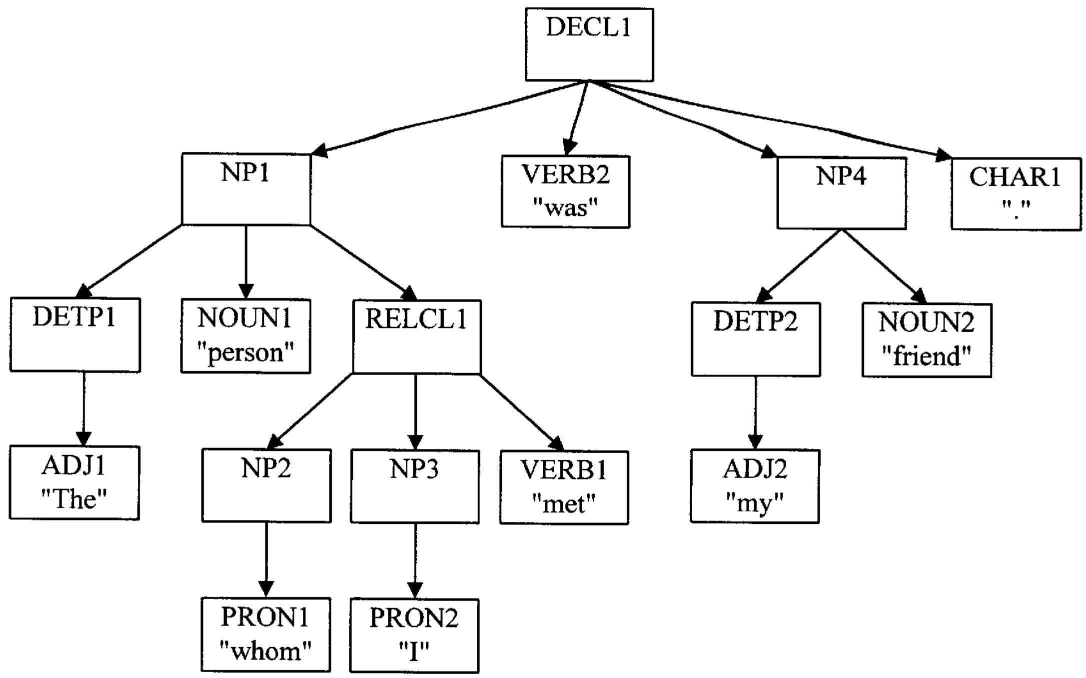


Fig. 57



Rule: PsLF_RelPro with node "whom" removes node and adds link

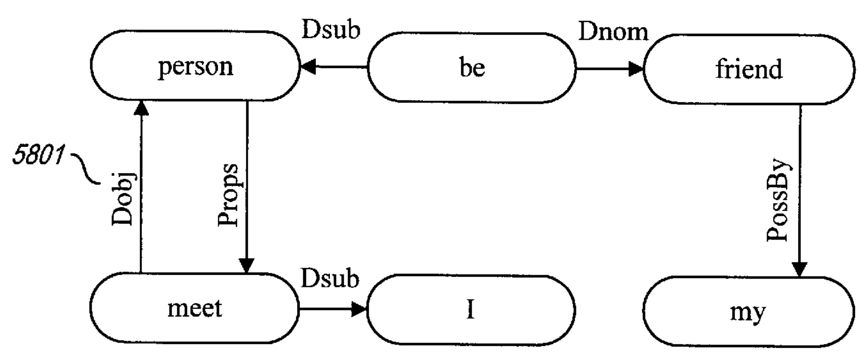
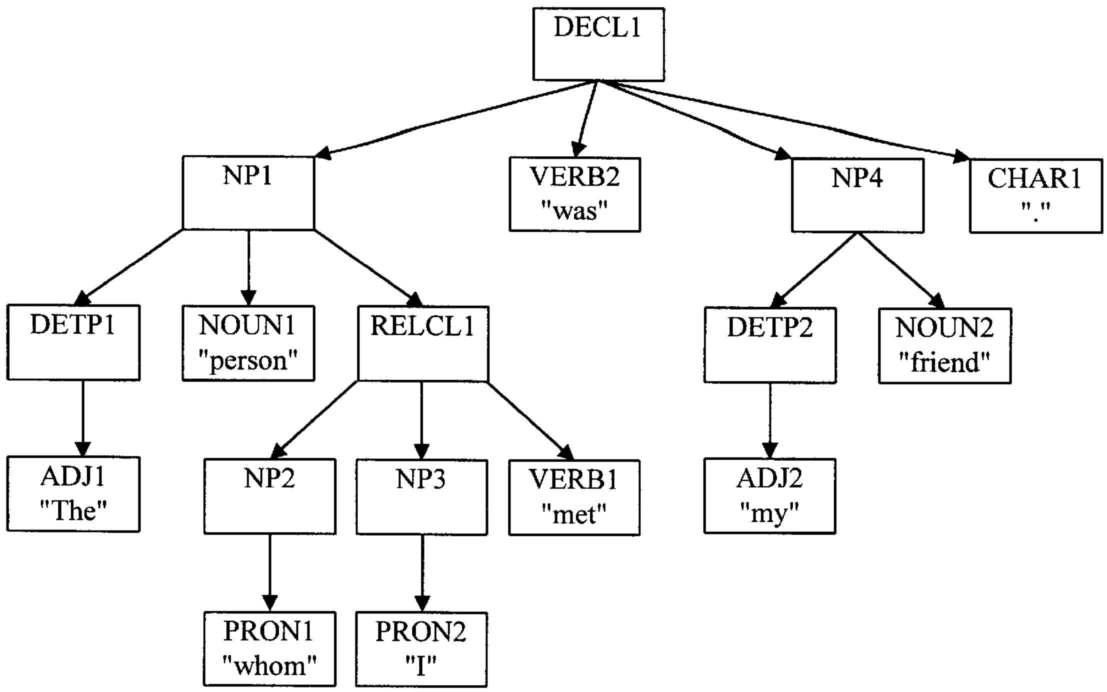


Fig. 58



Rule: PsLF_UnifyProns consolidates nodes "I" and "my" into a single node

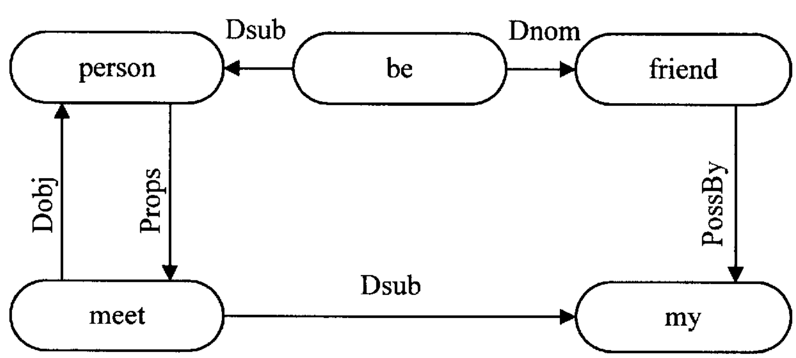


Fig. 59

METHOD AND SYSTEM FOR COMPUTING SEMANTIC LOGICAL FORMS FROM SYNTAX TREES

TECHNICAL FIELD

The present invention relates to the field of natural language processing ("NLP"), and more particularly, to a method and system for generating a logical form graph from a syntax tree.

BACKGROUND OF THE INVENTION

Computer systems for automatic natural language processing use a variety of subsystems, roughly corresponding to the linguistic fields of morphological, syntactic, and semantic analysis to analyze input text and achieve a level of machine understanding of natural language. Having understood the input text to some level, a computer system can, for example, suggest grammatical and stylistic changes to the input text, answer questions posed in the input text, or effectively store information represented by the input text.

Morphological analysis identifies input words and provides information for each word that a human speaker of the natural language could determine by using a dictionary. Such information might include the syntactic roles that a word can play (e.g., noun or verb) and ways that the word can be modified by adding prefixes or suffixes to generate different, related words. For example, in addition to the word "fish," the dictionary might also list a variety of words related to, and derived from, the word "fish," including "fishes," "fished," "fishing," "fisher," "fisherman," "fishable," "fishability," "fishbowl," "fisherwoman," "fishery," "fishhook," "fishnet," and "fishy."

Syntactic analysis analyzes each input sentence, using, as a starting point, the information provided by the morphological analysis of input words and the set of syntax rules that define the grammar of the language in which the input sentence was written. The following are sample syntax rules:

sentence=noun phrase+verb phrase

noun phrase=adjective+noun

verb phrase=adverb+verb Syntactic analysis attempts to find an ordered subset of syntax rules that, when applied to the words of the input sentence, combine groups of words into phrases, and then combine phrases into a complete sentence. For example, consider the input sentence: "Big dogs fiercely bite." Using the three simple rules listed above, syntactic analysis would identify the words "Big" and "dogs" as an adjective and noun, respectively, and apply the second rule to generate the noun phrase "Big dogs." Syntactic analysis would identify the words "fiercely" and "bite" as an adverb and verb, respectively, and apply the third rule to generate the verb phrase "fiercely bite." Finally, syntactic analysis would apply the first rule to form a complete sentence from the previously generated noun phrase and verb phrase. The result of syntactic analysis, often represented as an acyclic downward branching tree with nodes representing input words, punctuation symbols, phrases, and a root node representing an entire sentence, is called a parse.

Some sentences, however, can have several different parses. A classic example sentence for such multiple parses is: "Time flies lie an arrow." There are at least three possible parses corresponding to three possible meanings of this sentence. In the first parse, "time" is the subject of the sentence, "flies" is the verb, and "like an arrow" is a

prepositional phrase modifying the verb "flies." However, there are at least two unexpected parses as well. In the second parse, "time" is an adjective modifying "flies," "like" is the verb, and "an arrow" is the object of the verb. This parse corresponds to the meaning that flies of a certain type, "time flies," like or are attracted to an arrow. In the third parse, "time" is an imperative verb, "flies" is the object, and "like an arrow" is a prepositional phrase modifying "time." This parse corresponds to a command to time flies as one would time an arrow, perhaps with a stopwatch.

Syntactic analysis is often accomplished by constructing one or more hierarchical trees called syntax parse trees. Each leaf node of the syntax parse tree generally represents one word or punctuation symbol of the input sentence. The application of a syntax rule generates an intermediate-level node linked from below to one, two, or occasionally more existing nodes. The existing nodes initially comprise only leaf nodes, but, as syntactic analysis applies syntax rules, the existing nodes comprise both leaf nodes as well as intermediate-level nodes. A single root node of a complete syntax parse tree represents an entire sentence.

Semantic analysis generates a logical form graph that describes the meaning of input text in a deeper way than can be described by a syntax parse tree alone. The logical form graph is a first attempt to understand the input text at a level analogous to that achieved by a human speaker of the language.

The logical form graph has nodes and links, but, unlike the syntax parse tree described above, is not hierarchically ordered. The links of the logical form graph are labeled to indicate the relationship between a pair of nodes. For example, semantic analysis may identify a certain noun in a sentence as the deep subject or deep object of a verb. The deep subject of a verb is the doer of the action and the deep object of a verb is the object of the action specified by the verb. The deep subject of an active voice verb may be the syntactic subject of the sentence, and the deep object of an active voice verb may be the syntactic object of the verb. However, the deep subject of a passive voice verb may be expressed in an agentive-by phrase, and the deep object of a passive voice verb may be the syntactic subject of the sentence. For example, consider the two sentences: (1) "Dogs bite people" and (2) "People are bitten by dogs." The first sentence has an active voice verb, and the second sentence has a passive voice verb. The syntactic subject of the first sentence is "Dogs" and the syntactic object of the verb "bite" is "people." By contrast, the syntactic subject of the second sentence is "People" and the verb phrase "are bitten" is modified by the agentive-by phrase "by dogs." For both sentences, "dogs" is the deep subject, and "people" is the deep object of the verb or verb phrase of the sentence. Although the syntax parse trees generated by syntactic analysis for sentences 1 and 2, above, will be different, the logical form graphs generated by semantic analysis will be the same, because the underlying meaning of the two sentences is the same.

Further semantic processing after generation of the logical form graph may draw on knowledge databases to relate analyzed text to real world concepts in order to achieve still deeper levels of understanding. An example knowledge base would be an on-line encyclopedia, from which more elaborate definitions and contextual information for particular words can be obtained.

In the following, the three NLP subsystems—morphological, syntactic, and semantic—are described in the context of processing the sample input text: "The person whom I met was my friend." FIG. 1 is a block diagram illustrating the flow of information between the NLP subsystems. The morphological subsystem **101** receives the input text and outputs an identification of the words and senses for each of the various parts of speech in which each word can be used. The syntactic subsystem **102** receives this information and generates a syntax parse tree by applying syntax rules. The semantic subsystem **103** receives the syntax parse tree and generates a logical form graph.

FIGS. **2–5** display the dictionary information stored on an electronic storage medium that is retrieved for the input words of the sample input text during morphological analysis. FIG. **2** displays the dictionary entries for the input words "the" **201** and "person" **202**. Entry **201** comprises the key "the" **203** and a list of attribute/value pairs. The first attribute "Adj" **204** has, as its value, the symbols contained within the braces **205** and **206**. These symbols comprise two further attribute/value pairs: (1) "Lemma"/"the" and (2) "Bits"/ "Sing Plur Wa6 Det Art B0 Def." A lemma is the basic, uninflected form of a word. The attribute "Lemma" therefore indicates that "the" is the basic, uninflected form of the word represented by this entry in the dictionary. The attribute "Bits" comprises a set of abbreviations representing certain morphological and syntactic information about a word. This information indicates that "the" is: (1) singular; (2) plural; (3) not inflectable; (4) a determiner; (5) an article; (6) an ordinary adjective; and (7) definite. Attribute **204** indicates that the word "the" can serve as an adjective. Attribute **212** indicates that the word "the" can serve as an adverb. Attribute "Senses" **207** represents the various meanings of the word as separate definitions and examples, a portion of which are included in the list of attribute/value pairs between braces **208-209** and between braces **210-211**. Additional meanings actually contained in the entry for "the" have been omitted in FIG. **2**, indicated by the parenthesized expression "(more sense records)" **213**.

In the first step of natural language processing, the morphological subsystem recognizes each word and punctuation symbol of the input text as a separate token and constructs an attribute/value record for each part of speech of each token using the dictionary information. Attributes are fields within the records that can have one of various values defined for the particular attribute. These attribute/value records are then passed to the syntactic subsystem for further processing, where they are used as the leaf nodes of the syntax parse tree that the syntactic subsystem constructs. All of the nodes of the syntax parse tree and the logical form graph constructed by subsequent NLP subsystems are attribute/value records.

The syntactic subsystem applies syntax rules to the leaf nodes passed to the syntactic subsystem from the morphological subsystem to construct higher-level nodes of a possible syntax parse tree that represents the sample input text. A complete syntax parse tree includes a root node, intermediate-level nodes, and leaf nodes. The root node represents the syntactic construct (e.g., declarative sentence) for the sample input text. The intermediate-level nodes represent intermediate syntactic constructs (e.g., verb, noun, or prepositional phrases). The leaf nodes represent the initial set of attribute/value records.

In some NLP systems, syntax rules are applied in a top-down manner. The syntactic subsystem of the NLP system herein described applies syntax rules to the leaf nodes in a bottom-up manner. That is, the syntactic subsystem attempts to apply syntax rules one-at-a-time to single leaf nodes to pairs of leaf nodes, and, occasionally, to larger groups of leaf nodes. If the syntactic rule requires two leaf nodes upon which to operate, and a pair of leaf nodes both contain attributes that match the requirements specified in the rule, then the rule is applied to them to create a higher-level syntactic construct. For example, the words "my friend" could represent an adjective and a noun, respectively, which can be combined into the higher-level syntactic construct of a noun phrase. A syntax rule corresponding to the grammar rule, "noun phrase=adjective+ noun," would create an intennediate-level noun phrase node, and link the two leaf nodes representing "my" and "friend" to the newly created intermediate-level node. As each new intermediate-level node is created, it is linked to already-existing leaf nodes and intermediate-level nodes, and becomes part of the total set of nodes to which the syntax rules are applied. The process of applying syntax rules to the growing set of nodes continues until either a complete syntax parse tree is generated or until no more syntax rules can be applied. A complete syntax parse tree includes all of the words of the input sentence as leaf nodes and represents one possible parse of the sentence.

This bottom-up method of syntax parsing creates many intermediate-level nodes and sub-trees that may never be included in a final, complete syntax parse tree. Moreover, this method of parsing can simultaneously generate more than one complete syntax parse tree.

The syntactic subsystem can conduct an exhaustive search for all possible complete syntax parse trees by continuously applying the rules until no additional rules can be applied. The syntactic subsystem can also try various heuristic approaches to first generate the most probable nodes. After one or a few complete syntax parse trees are generated, the syntactic subsystem typically can terminate the search because the syntax parse tree most likely to be chosen as best representing the input sentence is probably one of the first generated syntax parse trees. If no complete syntax parse trees are generated after a reasonable search, then a fitted parse can be achieved by combining the most promising sub-trees together into a single tree using a root node that is generated by the application of a special aggregation rule.

FIG. **6** illustrates the initial leaf nodes created by the syntactic subsystem for the dictionary entries initially displayed in FIGS. **2–5**. The leaf nodes include two special nodes, **601** and **614**, that represent the beginning of the sentence and the period terminating the sentence, respectively. Each of the nodes **602–613** represent a single part of speech that an input word can represent in a sentence. These parts of speech are found as attribute/value pairs in the dictionary entries. For example, leaf nodes **602** and **603** represent the two possible parts of speech for the word "The," that are found as attributes **204** and **212** in FIG. **2**.

FIG. **7–22** show the rule-by-rule construction of the final syntax parse tree by the syntactic subsystem. Each of the figures illustrates the application of a single syntax rule to generate an intermediate-level node that represents a syntactic structure. Only the rules that produce the intermediate-level nodes that comprise the final syntax tree are illustrated. The syntactic subsystem generates many intermediate-level nodes which do not end up included in the final syntax parse tree.

In FIGS. **7–14**, the syntactic subsystem applies unary syntax rules that create intermediate-level nodes that represent simple verb, noun, and adjective phrases. Starting with FIG. **15**, the syntactic subsystem begins to apply binary syntax rules that combine simple verb, noun, and adjective phrases into multiple-word syntactic constructs. The syntactic subsystem orders the rules by their likelihood of successful application, and then attempts to apply them one-by-one until it finds a rule that can be successfully applied to the existing nodes. For example, as shown in FIG. **15**, the syntactic subsystem successfully applies a rule that creates a node representing a noun phrase from an adjective phrase and a noun phrase. The rule specifies the characteristics required of the adjective and noun phrases. In this example, the adjective phrase must be a determiner. By following the pointer from node **1501** back to node **1503**, and then accessing morphological information included in node **1503**, the syntactic subsystem determines that node **1501** does represent a determiner. Having located the two nodes **1501** and **1502** that meet the characteristics required by the rule, the syntactic subsystem then applies the rule to create from the two simple phrases **1501** and **1502** an intermediate-level node that represents the noun phrase "my friend." In FIG. **22**, the syntactic subsystem generates the final, complete syntax parse tree representing the input sentence by applying a trinary rule that combines the special Begin1 leaf node **2201**, the verb phrase "The person whom I met was my friend" **2202**, and the leaf node **2203** that represents the final terminating period to form node **2204** representing the declarative sentence.

The semantic subsystem generates a logical form graph from a complete syntax parse tree. In some NLP systems, the logical form graph is constructed from the nodes of a syntax parse tree, adding to them attributes and new bi-directional links. The logical form graph is a labeled, directed graph. It is a semantic representation of an input sentence. The information obtained for each word by the morphological subsystem is still available through references to the leaf nodes of the syntax parse tree from within nodes of the logical form graph. Both the directions and labels of the links of the logical form graph represent semantic information, including the functional roles for the nodes of the logical form graph. During its analysis, the semantic subsystem adds links and nodes to represent (1) omitted, but implied, words; (2) missing or unclear arguments and adjuncts for verb phrases; and (3) the objects to which prepositional phrases refer.

FIG. **23** illustrates the complete logical form graph generated by the semantic subsystem for the example input sentence. Meaningful labels have been assigned to links **2301–2306** by the semantic subsystem as a product of the successful application of semantic rules. The six nodes **2307–2312**, along with the links between them, represent the essential components of the semantic meaning of the sentence. In general, the logical form nodes roughly correspond to input words, but certain words that are unnecessary for conveying semantic meaning, such as "The" and "whom" do not appear in the logical form graph, and the input verbs "met" and "was" appear as their infinitive forms "meet" and "be." The nodes are represented in the computer system as records, and contain additional information not shown in FIG. **23**. The fact that the verbs were input in singular past tense form is indicated by additional information within the logical form nodes corresponding to the meaning of the verbs, **2307** and **2310**.

The differences between the syntax parse tree and the logical form graph are readily apparent from a comparison of FIG. **23** to FIG. **22**. The syntax parse tree displayed in FIG. **22** includes **10** leaf nodes and **16** intermediate-level nodes linked together in a strict hierarchy, whereas the logical form graph displayed in FIG. **23** contains only 6 nodes. Unlike the syntax parse tree, the logical form graph is not hierarchically ordered, obvious from the two links having opposite directions between nodes **2307** and **2308**. In addition, as noted above, the nodes no longer represent the exact form of the input words, but instead represent their meanings.

Further natural language processing steps occur after semantic analysis. They involve combining the logical form graph with additional information obtained from knowledge bases, analyzing groups of sentences, and generally attempting to assemble around each logical form graph a rich contextual environment approximating that in which humans process natural language.

Prior art methods for generating logical form graphs involve computationally complex adjustments to, and manipulations of the syntax parse tree. As a result, it becomes increasingly difficult to add new semantic rules to a NLP system. Addition of a new rule involves new procedural logic that may conflict with the procedural logic already programmed into the semantic subsystem. Furthermore, because nodes of the syntax parse tree are extended and reused as nodes for the logical form graph, prior art semantic subsystems produce large, cumbersome, and complicated data structures. The size and complexity of a logical form graph overlayed onto a syntax parse tree makes further use of the combined data structure error-prone and inefficient. Accordingly, it would be desirable to have a more easily extended and manageable semantic subsystem so that simple logical form graph data structures can be produced.

SUMMARY OF THE INVENTION

The present invention is directed to a method and system for performing semantic analysis of an input sentence within a NLP system. The semantic analysis subsystem receives a syntax parse tree generated by the morphological and syntactic subsystems. The semantic analysis subsystem applies two sets of semantic rules to make adjustments to the received syntax parse tree. The semantic analysis subsystem then applies a third set of semantic rules to create a skeletal logical form graph from the syntax parse tree. The semantic analysis subsystem finally applies two additional sets of semantic rules to the skeletal logical form graph to provide semantically meaningful labels for the links of the logical form graph, to create additional logical form graph nodes for missing nodes, and to unify redundant logical form graph nodes. The final logical form graph generated by the semantic analysis subsystem represents the complete semantic analysis of an input sentence.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram illustrating the flow of information between the subsystems of a NLP system.

FIGS. **2–5** display the dictionary information stored on an electronic storage medium that is retrieved for each word of the example input sentence: "The person whom I met was my friend."

FIG. **6** displays the leaf nodes generated by the syntactic subsystem as the first step in parsing the input sentence.

FIGS. **7–22** display the successive application of syntax rules by the syntactic subsystem to parse of the input sentence and produce a syntax parse tree.

FIG. **23** illustrates the logical form graph generated by the semantic subsystem to represent the meaning of the input sentence.

FIG. **24** shows a block diagram illustrating a preferred computer system for natural language processing.

FIG. **25** illustrates the three phases of the preferred new semantical subsystem.

FIG. **26** is a flow diagram for the new semantic subsystem (NSS).

FIG. **27** displays the first set of semantic rules.

FIG. **28**A displays a detailed description of the semantic rule PrLF_You from the first set of semantic rules.

FIG. **28**B displays an example application of the semantic rule PrLF_You from the first set of semantic rules.

FIG. **29** displays the second set of semantic rules.

FIGS. **30**A-**30**B display a detailed description of the semantic rule TrLF_MoveProp from the second set of semantic rules.

FIG. **30**C displays an example application of the semantic rule TrLF_MoveProp from the second set of semantic rules.

FIG. **31** displays a flow diagram for apply_rules.

FIG. **32** displays a flow diagram for phase one of the NSS.

FIG. **33** displays the third set of semantic rules.

FIGS. **34**A–**34**C display a detailed description of the semantic rule SynToSem1 from the third set of semantic rules.

FIG. **34**D displays an example application of the semantic rule SynToSem1 from the third set of semantic rules.

FIG. **35** displays a flow diagram for phase two of the NSS.

FIGS. **36**–**38** display the fourth set of semantic rules.

FIG. **39**A displays a detailed description of the semantic rule LF_Dobj2 from the fourth set of semantic rules.

FIG. **39**B displays an example application of the semantic rule LF_Dobj2 from the fourth set of semantic rules.

FIG. **40** displays the fifth set of semantic rules.

FIGS. **41**A–**41**C display a detailed description of the semantic rule PsLF_PronAnaphora from the fifth set of semantic rules.

FIG. **41**D displays an example application of the semantic rule PsLF_PronAnaphora from the fifth set of semantic rules.

FIG. **42** displays a flow diagram for phase three of the NSS.

FIG. **43** is a block diagram of a computer system for the NSS.

FIGS. **44**–**59** display each successful rule application by the NSS as it processes the syntax parse tree generated for the example input sentence.

DETAILED DESCRIPTION OF THE INVENTION

The present invention provides a new semantic method and system for generating a logical form graph from a syntax tree. In a preferred embodiment, a new semantic subsystem (NSS) performs the semantic analysis in three phases: (1) filling in and adjusting the syntax parse tree, (2) generating an initial logical form graph, and (3) generating meaningful labels for links of the logical form graph and constructing a complete logical form graph. Each phase constitutes the application of one or two sets of rules to either a set of syntax tree nodes or to a set of logical form graph nodes.

The NSS addresses the recognized deficiencies in prior art semantic subsystems described above in the background section. Each phase of the NSS is a simple and extensible rule-based method. As additional linguistic phenomena are recognized, rules to handle them can be easily included into one of the rule sets employed by the NSS. In addition, the second phase of the NSS generates an entirely separate logical form graph, rather than overlaying the logical form graph onto an existing syntax parse tree. The logical form graph data structure generated by the NSS is therefore simple and space efficient by comparison with prior art logical form graph data structures.

FIG. **24** is a block diagram illustrating a preferred computer system for a NLP system. The computer system **2401** contains a central processing unit, a memory, a storage device, and input and output devices. The NLP subsystems **2406**–**2409** are typically loaded into memory **2404** from a computer-readable storage device such as a disk. An application program **2405** that uses the services provided by the NLP system is also typically loaded into memory. The electronic dictionary **2411** is stored on a storage device, such as a disk **2410**, and entries are read into memory for use by the morphological subsystem. In one embodiment, a user typically responds to a prompt displayed on the output device **2403** by entering one or more natural language sentences on an input device **2404**. The natural language sentences are received by the application, processed, and then passed to the NLP system by way of the morphological subsystem **2406**. The morphological subsystem uses information from the electronic dictionary to construct records describing each input word, and passes those records to the syntactic subsystem **2407**. The syntactic subsystem parses the input words to construct a syntax parse tree and passes the syntax parse tree to the semantic subsystem **2408**. The semantic subsystem generates a logical form graph from the received syntax parse tree and passes that logical form graph to other NLP subsystems **2409**. The application program then can send and receive information to the natural language subsystem **2409** in order to make use of the machine understanding of the input text achieved by the NLP system, and then finally output a response to the user on an output device **2403**.

FIG. **25** illustrates the three phases of the preferred new semantic subsystem. Phases **1**–**3** of the NSS are shown as **2502**, **2504**, and **2506**, respectively. The states of the relevant data structures input and output from each phase of the NSS are displayed in FIG. **25** as labels **2501**, **2503**, **2505**, and **2507**. The NSS receives a syntax parse tree **2501** generated by the syntactic subsystem. The first phase of the NSS **2502** completes the syntax parse tree using semantic rules, and passes the completed syntax parse tree **2503** to the second phase of the NSS **2504**. The second phase of the NSS generates an initial logical form graph **2505** and passes that initial logical form graph to the third phase of the NSS **2506**. The third phase of the NSS applies semantic rules to the initial logical form graph in order to add meaningful semantic labels to the links of the logical form graph, to add new links and nodes to fill out the semantic representation of the input sentence, and occasionally to remove redundant nodes. The complete logical form graph **2507** is then passed to other NLP subsystems for use in further interpreting the input sentence represented by the logical form graph or in answering questions or preparing data based on the input sentence.

A flow diagram for the NSS is displayed in FIG. **26**. The flow diagram shows successive invocation of the three phases of the NSS, **2601**, **2602**, and **2603**. In the following, each phase of the NSS will be described in detail.

NSS Phase One—Completing Syntactic Roles of the Syntax Tree

In phase one of the NSS, the NSS modifies a syntax parse tree received from the syntactic subsystem by applying two different sets of semantic rules to the nodes of the syntax parse tree. These semantic rules can alter the linkage structure of the syntax tree or cause new nodes to be added.

The NSS applies a first set of semantic rules to resolve a variety of possible omissions and deficiencies that cannot be addressed by syntactical analysis. Application of these first set of semantic rules effect preliminary adjustments to the input syntax parse tree. The linguistic phenomena addressed by the first set of semantic rules include verbs omitted after the words "to" or "not," but understood to be implicit by a human listener, missing pronouns, such as "you" or "we" in imperative sentences, expansion of coordinate structures involving the words "and" or "or," and missing objects or elided verb phrases. FIG. **27** lists a preferred first set of semantic rules applied by the NSS in phase one. For each rule, the name of the rule followed by a concise description of the linguistic phenomenon that it addresses is shown. The general format of each semantic rule is a set of conditions which are applied to a syntax parse tree node or logical form graph node and a list of actions that are applied to the syntax parse tree or logical form graph. For example, the NSS applies the conditions of each rule of the first set of semantic rules to the list of syntax records that represents the syntax parse tree and, for each rule for which all the conditions of that rule are satisfied, the NSS performs the list of actions contained in the rule, resulting in specific changes to the syntax parse tree. Of course, the actual form of each semantic rule depends on the details of the representation of the syntax parse tree and logical form graph, for which many different representations are possible. In the following figures, a semantic rule is described by a conditional expression preceded by the word "If" in bold type, followed by a list of actions preceded by the word "Then" in bold type. The "If" part of the semantic rule represents the conditions that must be applied to a syntax parse tree node or logical form graph node and found to be true in order for the rule, as a whole, to be applied to the node, and the "Then" expression represents a list of actions to be performed on the syntax parse tree or logical form graph. The displayed expression closely corresponds to the computer source code expression for the semantic rule.

FIG. **28**A displays an English-language representation of the semantic rule PrLF_You from the first set of semantic rules. As can be seen in FIG. **28**A, the "If" expression concerns the values of various attributes of the syntax parse tree node to which the rule is applied, and the "Then" expression specifies the creation of a pronoun node for the lemma "you" and a noun phrase node parent for the pronoun node and the attachment of the created nodes to the syntax parse tree.

FIG. **28**B shows an example of the application of the semantic rule PrLF_You to the syntax parse tree **2801** generated by the syntactic subsystem for the sentence "Please close the door." Application of PrLF_You results in the modified syntax parse tree **2802**, with two new nodes **2803** and **2804** connected to the root node for the sentence. This semantic rule has the purpose of explicitly placing an understood "you" of an imperative sentence into the syntax parse tree.

After all semantic rules of the first set of semantic rule that can be applied to the input syntax parse tree have been applied, the NSS makes main adjustments to the preliminarily-adjusted syntax parse tree by applying to the nodes of the preliminarily-adjusted syntax parse tree a second set of semantic rules. This second set of rules include rules that serve to identify and resolve long-distance attachment phenomena, to transform verbal phrases into verbs with prepositional phrase objects, and to replace, in certain cases, the word "if" with an infinitive clause.

FIG. **29** lists a preferred second set of semantic rules applied by the NSS in phase one. For each rule, the name of the rule followed by a concise description of the linguistic phenomenon that it addresses is shown. FIGS. **30**A-**30**B display an English-language representation of the semantic rule TrLF_MoveProp from the second set of semantic rules. As can be seen in FIGS. **30**A-**30**B, the "If" expression concerns the values of various attributes of the syntax parse tree node to which the rule is applied and various related syntax parse tree nodes, and the "Then" expression specifies a rather complex rearrangement of the syntax parse tree.

FIG. **30**C shows an example of the application of the semantic rule TrLF_MoveProp to the syntax parse tree **3001** generated by the syntactic subsystem for the sentence "I have no desire to see the man." Application of TrLF_MoveProp results in the modified syntax parse tree **3002**. The infinitive clause represented by node **3003** in the original syntax parse tree has been moved from its position as a child of node **3004** to being a child **3005** of the root DECL1 node **3006** of the modified syntax parse tree. This semantic rule has the purpose of moving clauses like the infinitive clause **3003** from a lower level to a higher level in the syntax tree to facilitate the subsequent transition from the syntax parse tree to a logical form graph.

In the preferred embodiment of the present invention, semantic rules are statements in a programming language that, when executed, create a new tree or graph node from one, two, or occasionally more existing tree or graph nodes and create appropriate links between the newly created node and the existing tree or graph nodes. In the preferred embodiment, the left-hand portion of a semantic rule specifies characteristics that the existing node or nodes must have in order for the rule to be applied. The right-hand portion of the semantic rule specifies the type of new node to be created, and the values for the new node's attributes. The rules described in FIG. **28** and in FIG. **30** exemplify this form.

In the preferred embodiment of the present invention, each syntax parse tree and each logical form graph is represented as a list of nodes, with the links between the nodes represented by attribute values within the nodes. Each set of rules is also represented as a list. Application of set of rules to a syntax parse tree involves selecting successive nodes from the list of nodes and attempting to apply to each selected node each rule from the list of rules representing the set of rules. A particular rule can be successfully applied to a node if that node has the characteristics specified in the left-hand portion of the rule. Occasionally, a new node may be created as a result of a successful rule application, or an existing node may be marked for deletion.

A flow diagram for the subroutine "apply_rules" which applies a set of rules to a list of nodes representing a syntax parse tree or logical form graph is displayed in FIG. **31**. The subroutine "apply_rules" is called by the NSS to apply each set of rules during each of the three phases of the NSS. In step **3101**, apply_rules receives a list of nodes as its first argument and a list of rules as its second argument. Steps **3102** through **3107** represent an outer loop, each iteration of which attempts to apply all of the input rules from the input list of rules to successive nodes selected from the input list. Steps **3103** through **3106** represent an inner loop, each iteration of which attempts to apply a rule selected from the list of input rules to a node selected from the input list of nodes. In step **3102**, apply_rules selects the next node from the input list of nodes, starting with the first. In step **3103**, apply_rules selects the next rule from the input list of rules, starting with the first. In step **3104**, apply_rules determines whether the selected node has the characteristics specified in the left-hand part of the selected rule. If the node has the specified characteristics, then apply_rules applies in step

3105 the selected rule to the selected of node. If apply_rules determines in step **3106** that there are more rules to attempt to apply to the selected node, apply_rules returns to step **3103** to select the next rule. If apply_rules determines in step **3107** that there are more nodes to attempt to apply the rules of the input rule list, apply_rules returns to step **3102** to select the next node.

A flow diagram for the processing done in the first phase of the NSS is displayed in FIG. **32**. In step **3201**, the variable "parameter1" is assigned to be the list of syntax parse tree nodes that comprise the syntax parse tree generated by the syntactic subsystem and input into the NSS. In step **3202**, the variable "parameter2" is assigned to be a list of the first set of semantic rules displayed in FIG. **27**. In step **3203**, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the first set of semantic rules to the syntax parse tree to effect preliminary adjustments. In step **3204**, the variable "parameter1" is assigned to be the list of syntax parse tree nodes that comprise the preliminarily-adjusted syntax parse tree. In step **3205**, the variable "parameter2" is assigned to be a list of the second set of semantic rules displayed in FIG. **29**. In step **3206**, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the second set of semantic rules to the syntax parse tree to effect main adjustments.

NSS Phase Two—Generating an Initial Logical Form Graph

In phase two of the NSS, the NSS applies a third set of semantic rules to the nodes of the adjusted syntax tree. Each successful rule application in phase two creates a new logical form graph node. By applying this third set of rules, the NSS creates a new logical form graph. The nodes of the logical form graph consist of only semantically meaningful attributes and a pointer back to the corresponding syntax tree node. Unlike in prior art semantic subsystems, the logical form graph nodes created by the NSS in phase two are completely separate and distinct from the syntax parse tree nodes. The NSS constructs a skeleton of the logical form graph that comprises links, stored as attributes within the nodes, that interconnect the nodes of the logical form graph.

In FIG. **33**, a list of the third set of semantic rules applied by the NSS in phase two is displayed. For each rule, FIG. **33** displays the name of the rule followed by a concise description of the linguistic phenomenon that it addresses. There are only three rules in this third set of rules, and only the first rule, SynToSem1, is commonly used. The second and third rules apply only to special situations when a fitted parse was generated by the syntactic subsystem, and the adjusted syntax parse tree therefore contains a fitted parse node.

FIGS. **34A–34C** display an English-language representation of the semantic rule SynToSem1 from the third set of semantic rules. As can be seen in FIGS. **34A–34C**, the "If" expression concerns the values of various attributes for the syntax parse tree node to which the rule is applied and various related syntax parse tree nodes, and the "Then" expression specifies the creation of a logical form graph node and placement of the new node within the incipient logical form graph.

FIG. **34D** shows an example of the application of the semantic rule SynToSem1 to the syntax parse tree **3401** generated by the syntactic subsystem for the sentence "The book was written by John." Application of SynToSem1 results in the skeletal logical form graph **3402**. The skeletal logical form graph has three nodes with temporary modifiers labeling the links. Attributes have been assigned to the new nodes, based on the syntactic attributes of the syntax parse tree nodes from which they were created. There are far fewer nodes in the logical form graph than in the corresponding syntax parse tree, because the logical form graph represents the semantic meaning of the sentence. The linguistic significance of the words "the," "was," and "by" in the original sentence is or will be incorporated into the attributes and labels of the logical form graph, and the complex node hierarchies which emanated from their presence as leaf nodes in the syntax parse tree are not necessary in the logical form graph.

FIG. **35** displays a flow diagram for phase two of the NSS. In step **3501**, the variable "parameter**1**" is assigned the list of nodes representing the adjusted syntax parse tree. In step **3502**, the variable "parameter2" is assigned to be a list of the third set of semantic rules displayed in FIG. **33**. In step **3503**, the NSS invokes subroutine "apply_rules" to apply the third set of semantic rules to the nodes of the adjusted syntax parse tree, thereby creating a new logical form graph corresponding to the adjusted syntax parse tree.

NSS Phase Three—Completing the Logical Form Graph

In phase three of the NSS, the NSS applies a fourth set of semantic rules to the skeletal logical form graph to add semantically meaningful labels to the links of the logical form graph. These new labels include "deep subject" ("Dsub"), "deep object" ("Dobj"), "deep indirect object" ("Dind"), "deep predicate nominative" ("Dnom"), "deep complement" ("Dcmp"), and "deep predicate adjective" ("Dadj"). In FIGS. **36–38**, a list of the fourth set of semantic rules applied by the NSS in phase three is displayed. For each rule, FIGS. **36–38** display the name of the rule followed by a concise description of the linguistic phenomenon that it addresses.

FIG. **39A** displays an English-language representation of the semantic rule LF_Dobj2 from the fourth set of semantic rules. As can be seen in FIG. **39A**, the "If" expression concerns the values of various attributes of the logical form graph node to which the rule is applied, and the "Then" expression specifies the labeling of a link in the logical form graph.

FIG. **39B** shows an example of the application of the semantic rule LF_Dobj2 to the logical form graph **3901** generated by the NSS for the sentence "The book was written by John." Application of LF_Dobj2 to a logical form graph containing a passive clause identifies the syntactic subject as the deep object of the action. This is accomplished, in FIG. **39B**, by relabeling link **3903** from a temporary modifier to the label **3904** indicating a deep object relationship.

As the final step in phase three, the NSS makes final adjustments to the logical form graph by applying a fifth set of semantic rules. This set of rules include rules that serve to unify a relative pronoun with its antecedent, find and explicitly include missing pronouns, resolve number ellipsis, provide missing deep subjects, unify redundant instances of personal pronouns, and contract coordinate structures expanded in the first sub-step of semantic analysis. These rules also deal with the problem of taking a pronoun (or "proform") and identifying the noun phrase to which it refers. In many cases, it is not possible to identify the correct noun phrase referent with the level of information that the logical form graph provides. In these cases, a list of the most likely candidates is created, and further processing is postponed until later steps of the NLP system that employ more global information. In FIG. **40**, a list of the fifth set of semantic rules applied by the NSS in phase three is displayed. For each rule, FIG. **40** displays the name of the rule followed by a concise description of the linguistic phenomenon that it addresses.

FIGS. **41**A–**41**C display an English-language representation of the semantic rule PsLF_PronAnaphora from the fifth set of semantic rules. As can be seen in FIGS. **41**A–**41**C, the "If" expression concerns the values of various attributes of the logical form graph node to which the rule is applied, and of related logical form graph nodes, and the "Then" expression specifies the addition of a logical form graph node representing an omitted referent of a pronoun.

FIG. **41**D shows an example of the application of the semantic rule PsLF_PronAnaphora to the logical form graph **4101** generated by the NSS for the sentence "Mary likes the man who came to dinner, and Joan likes him too." Application of PsLF_PronAnaphora to a logical form graph containing a pronoun node with a referent in a different part of the logical form graph adds a new node to which the pronoun node is directly linked. In FIG. **41**D, the new node **4103** has been added by application of PsLF_PronAnaphora to indicate that the node "he1" refers to "man."

A flow diagram for the processing done in phase three of the NSS is displayed in FIG. **42**. In step **4201**, the variable "parameter**1**" is assigned to be the list of logical form graph nodes that comprise the logical form graph generated during phase two of the NSS. In step **4202**, the variable "parameter2" is assigned to be a list of the fourth set of semantic rules displayed in FIGS. **36**–**38**. In step **4203**, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the fourth set of semantic rules to the logical form graph to add semantically meaningful labels to the links of the logical form graph. In step **4204**, the variable "parameter1" is assigned to be the list of the logical form graph nodes that comprise the meaningfully-labeled logical form graph generated in step **4203**. In step **4205**, the variable "parameter2" is assigned to be a list of the fifth set of semantic rules displayed in FIG. **40**. In step **4206**, the NSS invokes the subroutine "apply_rules," passing to the subroutine the variables "parameter1" and "parameter2." The subroutine "apply_rules" applies the fifth set of semantic rules to the logical form graph to effect final adjustments.

FIG. **43** is a block diagram of a computer system for the NSS. The computer **4300** contains memory with the semantic rules **4304**–**4308** and rule application engine **4303**. The rule application engine, under control of a central processing unit, applies the five sets of rules to the syntax parse tree **4301** to generate a corresponding logical form graph **4302**. The syntax parse tree is preferably generated by the morphological and syntactic subsystems, which are not shown. The syntax tree and logical form graph can also be used to accomplish a subsequent task requiring information analogous to that which a human reader would obtain from the input sentences. For example, a grammar checker program might suggest a new phrasing for the input sentence that more accurately or concisely states what was stated in the input sentence. As another example, a computer operating system might perform computational tasks described by the input sentence. As still another example, information contained in the input sentence might be categorized and stored away for later retrieval by a database management system.

Semantic Processing of the Example Input Sentence

The following discussion and FIGS. **44**–**59** describe the complete NSS processing of the example sentence "The person whom I met was my friend." Each semantic rule that is applied by the NSS will be described, along with a representation of the results of the rule application.

No preliminary adjustment rules from the first set of semantic rules are successfully applied to the syntax parse tree input into the NSS from the syntactic subsystem during phase one. One main adjustment rule from the second set of semantic rules is applied to the input syntax parse tree. FIG. **44** displays the syntax parse tree **4400** in the form it is input. Note that it is represented in FIG. **44** slightly more simply than in FIG. **22**. The NSS successfully applies the semantic rule TrLF_LongDist1, displayed in FIG. **29** as rule 1, to the relative clause node RELCL**1**, **4401**, of the syntax parse tree **4400** to generate the adjusted syntax parse tree **4402**. The effect of applying rule TrLF_LongDist1 is the introduction of a direct object attribute in the noun phrase node **4403** to indicate that the word "whom" is the direct object of the phrase "I met." Normally, in English, the direct object of a verb follows the verb. Because "whom" does not follow "I met" in the sentence that was parsed to produce the syntax tree **4400**, the fact that "whom" is the direct object of "I met" was not identified by the application of syntactic rules.

Seven rules from the third set of semantic rules are successfully applied in phase two of the NSS. In FIG. **45**, the NSS successfully applies the semantic rule SynToSem1, displayed in FIG. **33** as rule 1, to the determinate pronoun node DEPT2, **4501**, of the syntax parse tree to generate the logical form graph node "my" **4502**. In FIG. **46**, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP**4**, **4601**, of the syntax parse tree to generate the logical form graph node "friend" **4602** and the link **4603** with the temporary semantic label "Tmods" **4604**. In FIG. **47**, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP**3**, **4701**, of the syntax parse tree to generate the logical form graph node "I" **4702**. In FIG. **48**, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP**2**, **4801**, of the syntax parse tree to generate the logical form graph node "whom" **4802**. In FIG. **49**, the NSS successfully applies the semantic rule SynToSem1 to the relative clause node RELCL**1**, **4901**, of the syntax parse tree to generate the logical form graph node "meet" **4902** and the link **4903** with the temporary semantic label "Tmods" **4904**. In FIG. **50**, the NSS successfully applies the semantic rule SynToSem1 to the noun phrase node NP**1**, **5001**, of the syntax parse tree to generate the logical form graph node "person" **5002** and the link **5003** with the temporary semantic label "Tmods" **5004**. In FIG. **51**, the NSS successfully applies the semantic rule SynToSem1 to the declarative sentence node DECL**1**, **5101**, of the syntax parse tree to generate the logical form graph node "be" **5102** and the link **5103** with the temporary semantic label "Tmods" **5104**. Thus, with the completion of phase two of the NSS, a skeletal logical form graph has been created.

Six rules from the fourth set of semantic rules are successfully applied in phase three of the NSS. In FIG. **52**, the NSS successfully applies the semantic rule LF_Dusb1, displayed in FIG. **36** as rule 1, to the logical form graph node "be" **5201** to generate the link label "Dsub" **5202** and the link **5203** with the temporary semantic label "Tmods" **5204**. In FIG. **53**, the NSS successfully applies the semantic rule LF_Dnom, displayed in FIG. **36** as rule 10, to the logical form graph node "be" **5301** to generate the link label "Dnom" **5302**. In FIG. **54**, the NSS successfully applies the semantic rule LF_Props, displayed in FIG. **38** as rule 21, to the logical form graph node "person" **5401** to generate the link label "Props" **5402**. In FIG. **55**, the NSS successfully applies the semantic rule LF_Dusb1, displayed in FIG. **36** as rule 1, to the logical form graph node "meet" **5501** to generate the link label "Dsub" **5502**. In FIG. **56**, the NSS successfully applies the semantic rule LF_Dobj1, displayed in FIG. **36** as rule 3, to the logical form graph node "meet" **5601** to generate the link labeled "Dobj" **5603** to link the node "meet" to the node "whom" **5602**. In FIG. **57**, the NSS successfully applies the semantic rule LF_Ops, displayed in FIG. **38** as rule 22, to the logical form graph node "friend" **5701** to generate the link label "PossBy" **5702**.

One rule from the fifth set of semantic rules is successfully applied in phase three of the NSS. In FIG. **58**, the NSS successfully applies the semantic rule PsLF_RelPro, displayed in FIG. **40** as rule 1, to the logical form graph node "whom," displayed as **5602** in FIG. **56**, to generate the link labeled "Dobj" **5801** and to remove the node "whom." In FIG. **59**, the NSS successfully applies the semantic rule PsLF_UnifyPron, displayed in FIG. **40** as rule 10, to the logical form graph to consolidate the nodes "I" and "my" into a single node. This is the last rule applied successfully by the NSS. FIG. **59** thus displays the final, complete logical form graph generated by the NSS for the input sentence "The person whom I met was my friend."

Although the present invention has been described in terms of a preferred embodiment, it is not intended that the invention be limited to this embodiment. Modifications within the spirit of the invention will be apparent to those skilled in the art. The scope of the present invention is defined by the claims that follow.

We claim:

1. A method in a computer system for generating a logical form graph for a phrase of words specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the computer system having a memory the method comprising:

generating in the memory all initial syntax parse tree of the phrase based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of the words of the phrase;

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that arc implicit in the phrase;

generating in the memory a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skeletal logical form graph to identify semantic constructs to complete the logical form graph.

2. The method of claim **1** wherein the step of adjusting the initial syntax parse tree includes adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the phrase.

3. The method of claim **1** wherein the step of adjusting the skeletal logical form graph includes adding semantic labels to the generated skeletal logical form graph.

4. A computer-readable medium containing instructions for causing a computer system to generate a logical form graph for a sentence specified in a natural language, the natural language having a grammar specifying syntax of the natural language, the computer system having an initial syntax parse tree of the sentence that represents a parse of the sentence based on the grammar of the natural language, the initial syntax parse tree containing nodes representing syntactic construct of words of the sentence, the initial syntax parse tree being stored in memory of the computer system by:

adjusting the initial syntax parse tree to complete syntactic analysis for syntactic constructs that are implicit in the sentence;

generating in memory of The computer system a skeletal logical form graph for the adjusted syntax parse tree, the skeletal logical form graph being represented in a data structure that is independent of a data structure of the syntax parse tree; and

adjusting the skelctal logical form graph to identify semantic constructs to complete the logical form graph for the sentence.

5. The computer-readable medium of claim **4** wherein the adjusting of the initial syntax parse tree includes adding syntactic roles to the syntax parse tree for any syntactic constructs that are implicit in the sentence.

6. The computer-readable medium of claim **4** wherein adjusting of the skeletal logical form graph includes adding semantic labels to the generated skeletal logical form graph.

7. A method in a computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the method comprising the steps of:

generating in memory of the computer system a first data structure for a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

generating in memory of the computer system a second data structure for a logical form graph to represent a semantic analysis of the input text, the second data structure being generated from the syntax parse tree but being a separate data structure from the first data structure.

8. A computer system for processing input text representing a phrase or sentence of a natural language in order to represent in the computer system at least one meaning of the input text that a human speaker of the natural language would understand the input text to represent, the system comprising:

a component that generates in memory of the computer system a syntax parse tree from the input text to represent a syntactic analysis of the input text; and

a component that generates in memory of the computer system a logical form graph to represent a semantic analysis of the input text, the logical form graph being stored in a data structure that is separate from a data structure in which the generated syntax parse tree is stored, the logical form graph being generated based in part on the generated syntax parse tree.

9. The system of claim **8** wherein the component that generates a separate logical form graph comprises the following sub-components:

a first sub-component that generates an initial skeletal logical form graph; and

a second sub-component that identifies semantic roles for the nodes of the skeletal logical form graph and labels the directed links of the skeletal logical form graph to produce a final, complete logical form graph.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,966,686 Page 1 of 2

DATED : October 12, 1999

INVENTOR(S) : George Heidorn et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 1, line 42, a new paragraph should have been started with "Syntactic".

Column 9, line 64, "if" should be --it--.

Column 14, line 32, "DEPT2" should be --DETP2--.

Column 14, line 62, "LF__Dusbl" should be --LF__Dsub1--.

Column 15, line 6, "LF__Dusbl" should be --LF__Dsub1--.

Column 15, line 23, "PsLF_UnifyPron" should be --PsLF_UnifyProns--.

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,966,686 Page 2 of 2

DATED : October 12, 1999

INVENTOR(S) : George Heidorn et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Claims:

Claim 1, Column 15, line 41, "all" should be --an--.

Claim 4, Column 16, line 11, "The" should be --the--.

Signed and Sealed this

Second Day of January, 2001

Attest:

Q. TODD DICKINSON

Attesting Officer *Commissioner of Patents and Trademarks*