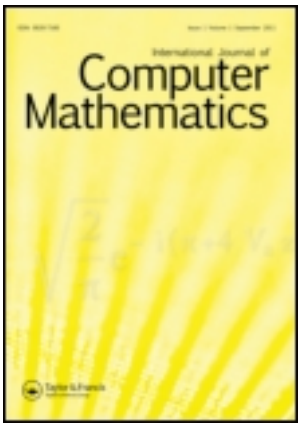


International Journal of Computer Mathematics
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/gcom20

The automatic transformational analysis of english sentences: An implementation

Jerry R. Hobbs [a] & Ralph Grishman [b]
[a] Computer Science Dept., City College, City University of New York, U.S.A.
[b] Computer Science Dept., New York University, U.S.A.

Available online: 21 Dec 2010

To cite this article: Jerry R. Hobbs & Ralph Grishman (1975): The automatic transformational analysis of english sentences: An implementation, International Journal of Computer Mathematics, 5:1-4, 267-283

To link to this article: http://dx.doi.org/10.1080/00207167608803117

Intern. J. Computer Math., 1976, Section A, Vol. 5, pp. 267–283

Printed in Great Britain

The Automatic Transformational Analysis of English Sentences: An Implementation

JERRY R. HOBBS

Computer Science Dept., City College, City University of New York, U.S.A.

and

RALPH GRISHMAN

Computer Science Dept., New York University, U.S.A.

We describe a system being developed for the transformational analysis of complex English sentences. The system is designed to be able to serve as a "front-end" for a variety of applications, such as question-answering, information retrieval, and command systems. It is a two-stage system, with the first stage being the Linguistic String Parser previously developed at New York University. The structure of the system and its relation to contemporary transformational parsers are considered. Several transformations, including those for nominalization, are described in detail, and several sentence analyses produced by the program are presented.

Work in natural language processing has been proceeding along two complementary lines. On the one hand, much effort has gone into the construction of large systems for syntactic analysis. On the other hand, systems with a more limited syntax have been developed for semantic processing in very small worlds (Winograd 1971, Charniak 1973) or on tabular data bases (Petrick 1973, Woods and Kaplan 1972). Ultimately, a large natural language processing system will have to draw from both lines, incorporating a large syntactic system while capitalizing on the insights gained from research in semantics.

One large syntactic system which is capable of handling a great proportion of English grammatical constructions, including conjunctions and comparatives, is the Linguistic String Parser (Sager 1967, Sager 1973, Grishman 1973, Grishman *et al.* 1973, Hobbs 1974a). This system has been in operation for a number of years at New York University. The parser and its string grammar were designed to be the first stage in a two-stage syntactic analysis of English

sentences into a simple underlying representation. The output of the first stage is a set of trees making explicit the surface structures of an English sentence; each tree represents a syntactically valid analysis of the sentence in accordance with the linguistic string theory of Harris (1962). In this paper we describe the initial work on the second stage—a program which takes these parse trees and transforms them into something very close to a predicate notation. Although this work is still in its early phases and much of it is provisional, a basic framework has been established and a number of transformations have been implemented. When completed, the parser and transformational program together could be used as a very powerful front end for a large variety of natural language processing systems.

The grammar for the first, or sentence segmentation, stage consists of a set of BNF productions and a set of conditions on the application of these productions; the conditions are expressed in a specially designed Restriction Language (Sager and Grishman 1975). The grammar for the second, or transformational, stage consists of transformations written in an extension of the Restriction Language. Each transformation performs certain tests on the structure of the tree and the attributes of the sentence words, and, if the requisite conditions are met, alters the tree. The sequencing among the transformations is specified entirely within the transformations themselves. The final result of all these tree modifications is a tree exhibiting the elementary assertions, as described in Section 2.

A comparison of this scheme with two other current transformational parsers is instructive, particularly with regard to the value of the sentence segmentation procedure:

1) The "traditional" transformational analysis procedures used by the MITRE group (Zwicky *et al.* 1965) and by Petrick and Plath (Petrick 1973, Plath 1974a, Plath 1974b) also proceed through two stages of surface analysis and transformational decomposition. Our surface grammar, however, is much more complex than theirs. It enables us to overcome two limitations on the efficiency of their transformational decomposition. First, our system can eliminate most syntactically invalid parses during the surface analysis. Their context-free surface grammars, in contrast, are inadequate for expressing many of the grammatical constraints; these constraints therefore cannot be applied until the transformational phase. As a result, a potentially large number of invalid parses must be followed through the entire surface analysis and part of the transformational phase before they are eliminated. Second, our surface trees, whenever practicable, indicate explicitly the transformational sources of the sentence. To this end, we provide distinct BNF definitions to analyze the word sequences resulting from different transformations, even though it would be possible to "cover" the language with a smaller set of definitions. In the transformational phase, the presence of a particular

structure will then trigger just that set of transformations needed to decompose the structure. In contrast, the traditional procedure tries every transformation, seeking one which applied to the current structure (one whose structural index matches); such a procedure becomes slower as the number of transformations increases.

For example, the classic sentences

The missionaries are eager to eat.

The missionaries are easy to eat.

both contain the infinitive phrase "to eat". But in our surface analysis this phrase would be matched with the symbol ⟨TOVO⟩ in the first sentence, ⟨TOVO–N⟩ in the second. (TOVO stands for "to" + *V*erb + possible *O*bject. The "–N" in ⟨TOVO–N⟩ indicates that a noun object—"missionaries"—has been deleted.) The parser makes this choice on the basis of attributes associated with the adjective to which the infinitive is attached. The node ⟨TOVO⟩ triggers a transformation which results in a structure corresponding to

The missionaries are eager that the missionaries eat.

while the node ⟨TOVO–N⟩ triggers a transformation yielding a structure corresponding to

It is easy for someone to eat the missionaries.

2) Woods' augmented transition network grammar has only a single stage, which builds the deep structure while doing the surface analysis (Woods 1970, Woods and Kaplan 1972). In several respects, however, the grammar is similar to ours. In both systems, grammatical restrictions are implemented as procedural predicates incorporated into the surface grammar. In both, the surface analysis directly controls the selection of transformations.

Both systems recognize the importance for efficiency of applying grammatical constraints during the surface analysis. Woods' system does this primarily by testing the deep structure which is built up during the surface analysis. Our system, in contrast, performs the tests directly on the surface tree. It is able to do so because the relationships which linguistic string analysis makes explicit in the surface structure are precisely those needed to state many grammatical constraints. Consequently, our system need not perform transformational decomposition before surface analysis is complete.

1. TARGET REPRESENTATION

In devising the target representation we have taken several criteria into account.

Linguists have proposed a large variety of syntactic and semantic

representations underlying sentences. These representations differ greatly in outward appearance, but many are characterized by two features: they allow predication, and they allow some predications to be subordinated to others in a form analogous to relative clauses. These two features define the fundamental structure of our target representation. We believe that the basic problem is analyzing sentences into *some* representation based on these features. Translation from this into other structures, such as the underlying trees of generative semanticists (Lakoff 1972), the case grammar of Fillmore (1968), or the predicate notation of Hobbs (1974b) should be relatively simple. Moreover, a target representation should not be inconsistent with lexical decomposition of verbs of the sort done by the generative semanticists (Lakoff 1972) or in the conceptual dependency networks of Schank (1973). In fact, we expect to do a certain amount of lexical decomposition within the transformational program.

An additional criterion for the target representation is that the original sentence should be recoverable from the target representation. For this reason, elementary assertions are tagged with the names of the reverse transformations that produced them. We can thereby be sure no information is lost and the program's output will be useable in a greater variety of applications.

On a detailed level, the target representation was devised to be compatible with the String Grammar as it is now written. This allows us to work at every intermediate stage of the analysis with the machinery that has already been built up for parsing. Moreover, since it will be some time before the work is completed, it allows us to obtain useful partial analyses of real-world sentences in the interim. With a few exceptions the target representation is a very much pared-down version of the entire grammar.

The major element in the target representation is the elementary assertion, ⟨ASSERTION⟩, which has the structure†

⟨SUBJECT⟩ ⟨TENSE⟩ ⟨VERB⟩ ⟨OBJECT⟩

There are transformations which extract the tense from the verb, so it is in infinitive form. The subject is either a noun string, ⟨NSTG⟩, or an assertion. The ⟨OBJECT⟩ node may subsume a single object or multiple objects, including some having an associated preposition. Some possible values of ⟨OBJECT⟩ are ⟨NULLOBJ⟩, ⟨NSTGO⟩, ⟨PN⟩, ⟨NPN⟩, ⟨PNPN⟩, and ⟨NPNPN⟩. ⟨NULLOBJ⟩ is a null node used for intransitive verbs. ⟨NSTGO⟩ is a place holder for either a noun string or an assertion. ⟨PN⟩ is a prepositional phrase and has the structure ⟨P⟩ ⟨NSTGO⟩ where ⟨P⟩ is a preposition. ⟨NPN⟩ is an ⟨NSTGO⟩ followed by a prepositional phrase,

†For brevity, we have omitted throughout this paper adjunct nodes which, while present for compatibility with the String Grammar, will all be null in the target representation.

⟨PNPN⟩ two prepositional phrases, etc. An example of the last of these values, ⟨NPNPN⟩, is

Iran exchanged *oil for weapons with France.*

The prepositions are retained in the target representation because they often signal case relations. While a more generally acceptable representation for multiple objects might be

⟨SUBJECT⟩ ⟨TENSE⟩ ⟨VERB⟩ ⟨OBJECT1⟩ ⟨OBJECT2⟩ ⟨OBJECT3⟩

we have rejected this in order to be more compatible with the present grammar.

Other possible values of ⟨OBJECT⟩ are an adjective string and strings involving adverbial prepositions or particles.

When a transformation creates an assertion out of another string, a "*T*-node" giving the name of the transformation is inserted above the ⟨ASSERTION⟩ node, making the original structure recoverable.

Subordinated material in a noun string—e.g., adjectives, relative clauses—is stored in trees, dominated by a "*T*-node", attached to the right of the ⟨NSTG⟩ node. The "*T*-node" carries the name of the transformation that removed the subordinated material. Below a "*T*-node" is an ⟨ASSERTION⟩, involving the noun string as one of its arguments. The noun string is referred to by the literal "HOST".

Subordinated material in an assertion—e.g., adverbials, subordinate clauses—is stored in similar trees to the right of the ⟨ASSERTION⟩ node. They contain another ⟨ASSERTION⟩ involving the first ⟨ASSERTION⟩ as an argument. Again, the first ⟨ASSERTION⟩ is referred to by the literal "HOST".

Only ⟨NSTG⟩ and ⟨ASSERTION⟩ nodes have trees with subordinated material.

A typical sentence and its target representation are shown in Figure 1.

2. IMPLEMENTATION

As part of our sentence segmentation component, we already had a powerful language for stating conditions on parse trees, the Restriction Language (RL) (Sager and Grishman 1975). For the transformational component, we had to add two types of operations to the language: one for transforming the tree and one for sequencing the transformations.

The operations for transforming the tree are:

REPLACE *node* BY *structure*
BEFORE *node* INSERT *structure*
AFTER *node* INSERT *structure*
DELETE *node*

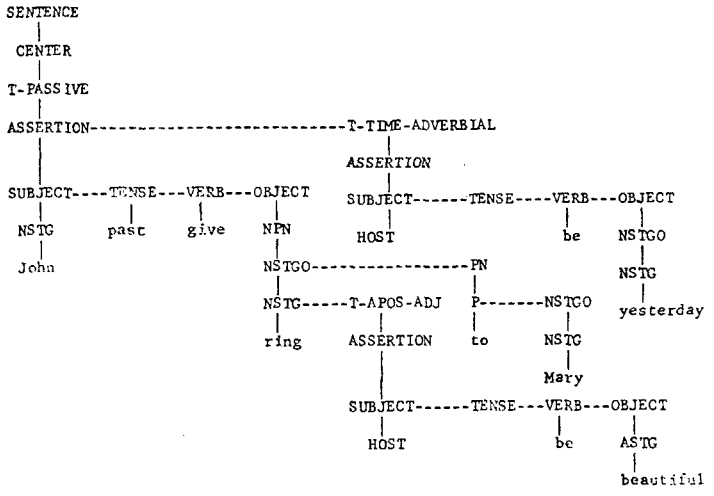
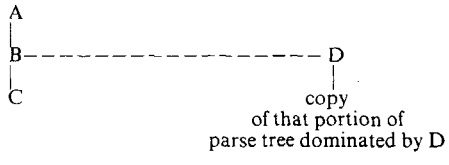


FIGURE 1 Target representation for "Mary was given a beautiful ring yesterday by John". Parse trees in this paper are represented by a binary notation: siblings are connected horizontally, with only the leftmost connected to the parent. Adjunct nodes, which are present but empty in the target representation, are not shown.

The first three operations create a new tree *structure* and put it in the parse tree in place of, before, or after some specified *node*; the last operation deletes a *node* together with all the nodes it dominates. The node may be specified by any construction acceptable as the subject of an RL statement. Typically, this will be the name of a node present in the tree or the name of an RL routine which locates a node in the tree. The new tree structure is described in a parenthesis notation: sibling nodes are separated by a "+", and the structure dominated by a node is written after that node, enclosed in parentheses. Each node in the structure may be either a newly created node or a copy of a node already present in the parse tree. A newly created node is indicated by a node name enclosed in "⟨ ⟩". A copied node is specified by any construction acceptable as an RL subject. If a copied node is nonterminal, and the new tree structure does not explicitly specify the descendants of the node, the node is copied together with the subtree it dominates in the parse tree. For example, the description

$$\langle A\rangle \quad (\quad B \quad (\quad \langle C\rangle \quad)+D)$$

would create the following structure:



A similar set of operations is employed in the Petrick–Plath parser (Plath 1974b).

We have sought to provide a flexible and efficient facility for sequencing the transformations. A part of each transformation, called the "housing", lists the set of node names (non-terminal symbols) with which this transformation is associated. In the course of the transformational analysis, the nodes of the tree are "activated" in a sequence specified by the user. When a node is activated, the transformations associated with that node are executed.

The user activates nodes in the analysis tree by executing the TRANSFORM command in a transformation. The operation

TRANSFORM *node*

where *node* specifies some node N of the tree, does the following: when the current transformation is finished, the currently active node is suspended and node N is activated; when all the transformations associated with N have executed, the suspended node is reactivated and the next transformation associated with that node is performed. This process may be nested: some of the transformations on N may include TRANSFORM operations on yet other nodes. The system starts things off by activating the root node (SENTENCE); from there on it is up to the user to pass control down the tree.

Some transformations may be marked as optional. Before such a transformation is executed, the current tree is saved. After the decomposition is complete (after the last transformation on the root node has been executed), the system reloads the saved tree, skips the optional transformation, and then continues the normal decomposition process.

A large number of optional transformations would cause an explosive increase in the time required for decomposition. We have found so far that only a small fraction of the transformations need be optional; these transformations reflect genuine syntactic ambiguities in the sentence. We should point out, however, that where the ambiguity is predictable, e.g., in cases of successive prepositional phrases, we suppress all but one parse. If a subsequent system incorporates semantic information, it will be a simple matter to produce all analyses so that a choice may be made on semantic grounds.

3. EXAMPLES OF TRANSFORMATIONS

In this section we describe three transformations. The first two illustrate the advantage the surface analysis gives the transformational component—a node in the tree identifies the transformation to be applied, so the statement of the transformation becomes quite simple. The first example is a transformation

which handles subordinated material. The second produces an embedded assertion.

1) Passive right adjunct: The right adjunct of a noun can contain a passive construction, as in

a construction *identified by the surface analysis*

There is a rule which transforms this construction into a passive assertion subordinated to the noun string (see Figure 2), then deletes the passive construction from the right adjunct, and finally calls the set of transformations applicable to the new assertion. Among these will be the passive transfor-

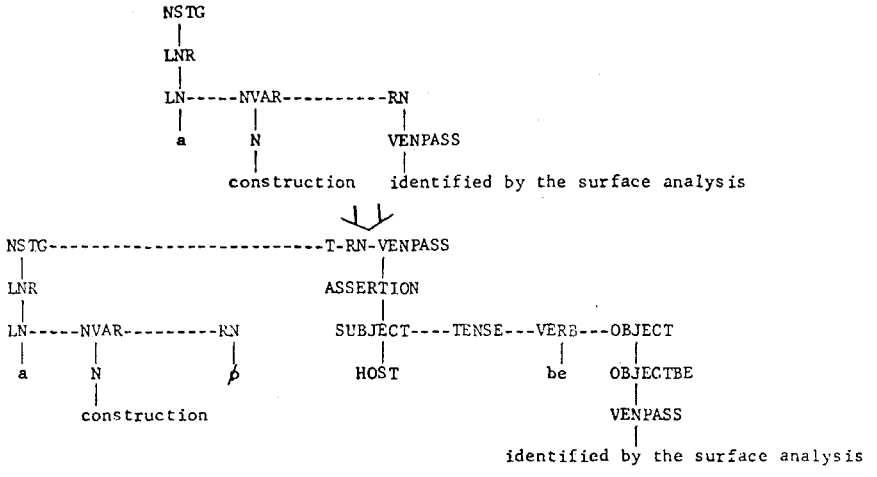


FIGURE 2 The effect of transformation TRN–VENPASS on "a construction identified by the surface analysis". In the trees, node LNR is a noun flanked by its left and right adjuncts, LN and RN; NVAR is a noun or one of its variants. VENPASS is the passive construction; it is one of many values of RN.

mation, which will transform the assertion into the active voice. Since the construction has already been identified by the surface analysis, no conditions have to be checked before applying this rule.

The transformation is as follows:

```
TRN–VENPASS=IN RN:
    IF VALUE IS VENPASS
    THEN BOTH AFTER LAST–COELEMENT OF
    IMMEDIATE NSTG–INSERT
        ⟨T–RN–VENPASS⟩ (⟨ASSERTION⟩ X1
```

(⟨SUBJECT⟩ ("HOST")
+⟨TENSE⟩ (⟨NULL⟩)
+⟨VERB⟩ ("BE")
+⟨OBJECT⟩ (⟨OBJECTBE⟩ (VENPASS))))
AND BOTH DELETE VENPASS AND TRANSFORM X1.

The first line indicates the node with which the transformation is to be stored. IMMEDIATE NSTG refers to the first NSTG node encountered going up the tree from RN. LAST–COELEMENT refers here to the rightmost "*T*-node" already subordinated to the noun string. The "X1" following "⟨ASSERTION⟩" causes the register (i.e., variable) X1 to point to the ASSERTION node. The instruction "TRANSFORM X1" causes all the transformations applicable at an ASSERTION node to be called for the new ASSERTION. These transformations include the passive, which produces a tree corresponding to

the surface analysis identify HOST.

2) Gerundive nominal: Assertions can be embedded in other assertions by means of the gerundive nominal construction, as in

His frequently reading newspapers in class annoys me.

There is a transformation which converts this into the corresponding assertion:

(He frequently read newspapers in class) annoys me.

It is possible that certain information is conveyed by the grammatical construction itself. Compare, for example,

John remembered that Kennedy was assassinated.
John remembered Kennedy's being assassinated.

Although both complements have the same underlying assertion, in the first John remembered the fact, while in the second the event. Our transformational analysis does not lose this distinction, since each assertion produced carries with it a tag indicating the transformation that produced it.

Since the surface analysis has already identified the construction, there are again no conditions to check; the transformation need only alter the tree. The possessive noun or pronoun is converted into its nominative form and placed in the subject of the new assertion, the present participle is converted into its infinitive form, and the set of transformations applicable to assertions is invoked for the new assertion.

3) The final example is a set of transformations representing a first approach to a major problem of linguistic analysis—nominalizations of verbs. These are verbs, or whole assertions, which have been converted into noun phrases. For example

John's refusal to go

may be viewed as a nominalization of the assertion

John refuse to go.

A major problem is determining the arguments, i.e., the subject and object, of the nominalization. Here the surface analysis offers virtually no help. String Project studies indicated that it was not profitable to analyze nominalizations at the surface level. They are given the same representation as other noun phrases.

There are two principal ways in English to embed predications in other predications. The first is by means of the assertion-like strings identified by the String Parser, such as the gerundive nominal. The second is by means of nominalizations and other noun strings. The second is more difficult to analyze because the relations of the various parts of a noun string to the main predication are rarely explicit. Thus we should not be surprised if the set of nominalization transformations becomes a significant portion of the grammar. The machinery developed here can be useful in analyzing other noun strings in which two or more nouns are related. For example, in "John's father", "cell membrane", and "king of England", "John", "cell", and "England" may be viewed as arguments of their respective head nouns.

It should be pointed out that the nominalization transformation is somewhat controversial. Lees (1960) and Vendler (1968) viewed nominalizations as transformationally derived, while Chomsky (1970) argued that the similarities between the verb and the noun ought to be expressed in the lexicon. Regardless of one's point of view, however, it is necessary to locate the arguments of the head noun. The difference seems to be whether we call the rules transformations or lexical redundancy rules. The rules themselves don't change. Nominalizations have been a focus of our work so far because one of the ongoing concerns of the Linguistic String Project is the automatic determination of noun and verb co-occurrence subclasses in scientific sublanguages of English (Sager 1972, Hirschman *et al.*, 1975), and most of the sublanguage information is buried in nominalizations. For example, one is more likely to encounter the noun string "the calcium concentration in the cells" than the assertion "calcium concentrates in the cells", although the latter occurs.

The first step in the analysis of nominalizations is to identify the noun as a

nominalization and to create an ASSERTION whose verb is the verb underlying the noun. Two kinds of nominalizations are distinguished at this point.

First there are the "action nominalizations" in which the noun stands for the entire action, e.g.

the action of digitalis on the heart	digitalis act on the heart
Nixon's dismissal of Dean	Nixon dismiss Dean
heart failure	heart fail

The second kind is the "argument nominalizations", in which the head noun stands for one of the arguments of the underlying assertion rather than for the action itself. For example,

dancer	one who dances
threat	that which someone threatens
cause	that which causes something
?picture	that in which someone depicts something

Each such noun is tagged in the dictionary with the type of nominalization it can be.

Some nominalizations are ambiguous between the argument and action type. For example,

argument: John's *gift* to Mary was a diamond ring.
action: John's *gift* of a diamond ring to Mary was a beautiful gesture.

Since, as with "gift", the argument interpretation seems to be more common in ambiguous cases, the argument transformation is applied first. It is made optional to allow the second analysis to be produced as well.

The next two sets of transformations build up the ⟨ASSERTION⟩ by supplying its arguments and adjuncts while cleaning out the left and right adjuncts in the noun string. Here there is no need to distinguish between action and argument nominalizations.

In searching the left and right adjuncts for arguments, the problem is that there is often very little information as to which argument an element is. This phase is broken up into four cases, according to the kinds of information available.

A) For certain sentential nouns, sentential complements in the right adjunct are recognized as the object of the underlying assertion. For example,

an indication *that she left*	That which indicates that she left

Aspectual nouns with infinitive complements are treated similarly;

Mary's attempt to leave	Mary attempt to leave

B) "Non-transitive" verbs are those for which the subject is the only argument not tagged by a preposition in the target representation†. For example,

The death of a genius	a genius die
John's reliance on secondary sources	John rely on secondary sources

The heuristic we have used is that the noun closer to the head noun should be favored as the subject. The left adjunct positions are searched in the following order:

pre-nominal noun: heart failure	heart fail
noun-like adjective: cellular function	cell function
possessive: John's arrival	John arrive

In the noun phrase "the patient's heart failure", "heart" will correctly be identified as the subject of "fail".

If the subject slot remains open after the left adjunct is searched, the right adjunct is searched for an "of" or a "by" prepositional phrase. Before a noun is placed in the subject slot, selectional constraints are applied; thus, this procedure can be improved by the use of the tighter selectional constraints available within a sublanguage.

C) Transitive verbs are more complex since there are two argument slots, a subject and an untagged object, competing for the nouns. Difficult cases occur. For example, in "cell requirements", the cell requires something, while in "oxygen requirements", something requires oxygen. We have decided to look at possessives and prepositional phrases first since they carry slightly more information than the pre-nominal noun and adjective.

As Chomsky (1970) has noted, some nominalizations favor the object interpretation for the possessive, others favor the subject. Examples of the former are "Rome's destruction", "Dean's dismissal". However, this object interpretation can be overridden by an "of" prepositional phrase in the right adjunct:

the barbarians' destruction of Rome
Nixon's dismissal of Dean

Hence, for these nouns the possessive is interpreted as the object unless an "of" phrase is found.

---

†We have used the term "non-transitive" rather than "intransitive", because the latter apparently does not include verbs like "rely" which require a prepositional phrase as its object.

The remaining nominalizations favor the subject interpretation:

Jaworski's investigation Jaworski investigate someone

The subject interpretation can be overridden by a "by" prepositional phrase, although it sounds somewhat awkward:

?Nixon's investigation by Jaworski.

Hence the possessive is interpreted as the subject unless a "by" phrase is found.

For argument nominalizations this transformation is optional, to allow the possessive to be interpreted also as a true possessive on the entity itself. For example, "Napoleon" in "Napoleon's picture" can be an argument—the object as in "that in which someone depicts Napoleon" or subject as in "that in which Napoleon depicts something"—or a true possessive as in "that in which someone depicts something and which belongs to Napoleon".

Next the prepositional phrases are searched for arguments. It is assumed that "by" signals a subject. For certain nominalizations there are other prepositions which may signal an argument, as "in" signals the subject for "interest":

John's interest in chess chess interests John

"Of" most frequently signals the object, so this interpretation is favored. However, genuine ambiguities may occur, as in

The criticism of Coleridge was shallow.

Hence this transformation is optional. It is followed by a rule effecting a subject interpretation, allowing alternate readings to be produced.

Finally the prenominal noun and adjective positions are searched. Here we favor the object interpretation, since this is generally the case in the data we have examined. These transformations are made optional to handle ambiguities.

In every case, selectional restrictions are applied before plugging a noun into the subject or object slot.

D) Those arguments which are tagged with prepositions in the target representation are extracted. Examples of this are

John's gift *to Mary*
my exchange *with Bill* of books *for records.*

This would be a fairly simple operation if the transformational component could depend on prepositional phrases being attached at the correct place. However, numerous examples have shown that a correct analysis can require unlimitedly detailed encyclopedic knowledge. Since there seemed to be little point in generating multiple parses when no means existed for deciding

between them, it was decided in the Linguistic String Parser to suppress all but one of the parses for sentences with such "permanent predictable ambiguities" (Sager 1967).

In particular, prepositional phrases are attached at the lowest level. In the noun string

the reliance of students in my class on secondary sources

"in my class" will be analyzed correctly as the right adjunct of "students" and "on secondary sources" incorrectly as the right adjunct of "class". When "students" is recognized as the subject, we must search down the tree for the next prepositional phrase which signals an argument, in this case "on secondary sources". Only the material between these two prepositional phrases is moved into the subject slot.

We should point out that these procedures are not universally successful. For example, in

the destruction of books by modern writers by South Dakota schools,

the noun phrase "modern writers by South Dakota schools" would be picked up as the subject.

After the arguments are found, adjunct material is removed from the noun phrase. For example,

the sudden arrival of spring	spring arrive suddenly

We have not yet implemented any transformations to handle this. In many cases these transformations will have to be optional when applied to argument nominalizations. An adjective, for example, may be derived from an adverb in the underlying assertion, or may be a true adjective describing the head noun itself. For example, "beautiful dancer" can mean "one who dances beautifully" or "one who dances and is beautiful."

Finally, for action nominalizations the ⟨ASSERTION⟩ which has been created and filled in replaces the noun string node. It is expected that the left and right adjuncts will be empty at this point. For argument nominalizations, the ⟨ASSERTION⟩ node is subordinated to the noun string node and the remaining transformations for the noun string are called to process any material left over in the left and right adjuncts.

4. FUTURE DIRECTIONS

We have decided on the target representation, built up the necessary machinery, and made a substantial beginning on some major problems in transformational analysis. We have developed a small set of transformations,

including those for nominalization, mentioned above, and conjunction, and are now testing a set of transformations for comparative constructions. Several major problems remain, however. In particular, transformations yielding an appropriate analysis of sentence adjuncts, or adverbials, must be implemented, and the various functions of adjectives and prenominal nouns must be traced out.

Acknowledgements

This work was supported in part by the Office of Naval Research under Contract N00014-67A-0467-0032 and by the National Science Foundation, Office of Science Information Services, under Contract GN-39879. Computer time was provided to us by the Atomic Energy Commission Computing Center at New York University. We have profited from discussions with Naomi Sager and Lynette Hirschman. Portions of the transformational grammar were debugged and tested by Joel Remde.

References

[1] E. Charniak, Jack and Janet in search of a theory of knowledge, in *Proc. Third Int'l Conf. on Artificial Intelligence* (Stanford Univ., Stanford, Calif.), (1973), 337–343.

[2] N. Chomsky, Remarks on nominalization, in Jacobs and Rosenbaum (eds.), *Readings in English Transformational Grammar* (Waltham, Mass.), (1970), 184–221.

[3] C. Fillmore, The case for case, in Bach and Harms (eds.), *Universals in Linguistic Theory* (New York), (1968), 1–90.

[4] R. Grishman, Implementation of the string parser of English, in R. Rustin (ed.), *Natural Language Processing* (New York), (1973), 89–109.

[5] R. Grishman, N. Sager, C. Raze, and B. Bookchin, The linguistic string parser, in *Proc. National Computer Conf. 1973* (Montvale, New Jersey) (1973), 427–434.

[6] Z. Harris, *String Analysis of Sentence Structure* (The Hague), (1962).

[7] L. Hirschman, R. Grishman, and N. Sager, Grammatically-based automatic word class formation, *Information Processing and Management*, **11**, (1975), 39.

[8] J. Hobbs, A metalanguage for expressing grammatical restrictions in nodal spans parsing of natural language, *Courant Computer Science Report No. 2*, New York Univ. (New York), (1974a).

[9] J. Hobbs, A model for natural language semantics, Part I: The model, *Dept. of Computer Science Research Report No. 36*, Yale Univ. (New Haven, Conn.), (1974b).

[10] G. Lakoff, Linguistics and natural logic, in Davidson and Harmon (eds.), *Semantics of Natural Language* (Boston, Mass) (1972), 545–665.

[11] R. Lees, *The Grammar of English Nominalizations* (The Hague), (1960).

[12] S. Petrick, Transformational analysis, in R. Rustin (ed.), *Natural Language Processing* (New York) (1973), 27–41.

[13] W. Plath, Transformational grammar and transformational parsing in the REQUEST system, in A. Zampolli (ed.), *Computational and Mathematical Linguistics, Proc. of the Int'l Conf. on Computational Linguistics* (Firenze). (1974a).

[14] W. Plath, String transformations in the REQUEST System, Report RC 4947 (No. 21963), IBM Corp. (1974b).

[15] N. Sager, Syntactic analysis of natural language, in F. Alt and M. Rubinoff (eds.), *Advances in Computers*, **8**, (New York), (1967).

[16] N. Sager, Syntactic formatting of scientific information, in *Proc. Fall Joint Comp. Conf. 1972* (Montvale, New Jersey), (1972).

[17] N. Sager, and R. Grishman, The restriction language for computer grammars of natural language, *Comm. of the Assn. for Computing Machinery*, (1975), 390.
[18] R. Schank, The fourteen primitive actions and their inferences, A. I. Memo 183, Computer Science Dept., Standford Univ. (Stanford, Calif.), (1973).
[19] Z. Vendler, *Adjectives and Nominalizations* (The Hague), (1968).
[20] T. Winograd, Procedures as a representation for data in a computer program for understanding natural language, MAC TR–84, MIT Project MAC (Cambridge, Mass.), (1971).
[21] W. Woods, Transition network grammars for natural language analysis, *Comm. of the Assn. for Computing Machinery* (1970), 591–606.
[22] W. Woods, R. M. Kaplan, and B. Nash-Webber, The lunar sciences natural language information system, Final Report, BBN Report No. 2378, Bolt Beranek and Newman (Cambridge, Mass.), (1972).
[23] A. Zwicky, J. Friedman, B. Hall, and D. E. Walker, The MITRE syntactic analysis procedure for transformational grammars, in *Proc. Fall Joint Comp. Conf. 1965* (Washington, D.C.), (1965), 317–326.

APPENDIX

We show below (on pages 282–283) the target structures for three sentences, in precisely the form produced by the program. This form compresses the output by suppressing some levels of the tree. A number below an element indicates the line on which the subtree it dominates can be found. Numbers to the left and right of a word refer to its left and right adjuncts. These outputs show adjunct nodes such as SA (sentence adjunct), which were omitted from trees shown in the paper because they are always null in the target representation.

1. Target structure for
"Cardiac failure was prevented by the doctor's injection of digitalis."
(Note: A second analysis was obtained, in which "doctor's" was taken in a descriptive rather than possessive sense, as in the phrase "printer's error"; this analysis is not shown here.)

```
DECOMPOSITION TREE:
1. SENTENCE    = INTRODUCER  CENTER  ENDMARK
                             2.
2. TPASSIVE    = ASSERTION
                   3.
3. ASSERTION   = SA  SUBJECT  SA  TENSE  SA  VERB     SA  OBJECT  RV  SA
                     4.           PAST       PREVENT      5.
4. T VN ACT    = ASSERTION
                   6.
5. T VN ACT    = ASSERTION
                   7.
6. ASSERTION   = SA  SUBJECT    SA  TENSE  SA  VERB    SA  OBJECT    RV  SA
                     8. DOCTOR                  INJECT      DIGITALIS
7. ASSERTION   = SA  SUBJECT  SA  TENSE  SA  VERB  SA  OBJECT  RV  SA
                     HEART                    FAIL
8. LN          = TPOS  QPCS  APOS  NSPOS  NPOS
                   THE
```

2. Target structure for
"John remembered Oswald's being shot by Ruby."

```
DECOMPOSITION TREE:
1. SENTENCE     = INTRODUCER  CENTER  ENDMARK
                              2.
2. T-TVTENSE    = ASSERTION
                    3.
3. ASSERTION    = SA  SUBJECT  SA  TENSE  SA  VERB      SA  OBJECT  RV  SA
                      JOHN         PAST      REMEMBER      4.
4. TNSVINGO     = TPASSIVE
                    5.
5. TPASSIVE     = ASSERTION
                    6.
6. ASSERTION    = SA  SUBJECT  SA  TENSE  SA  VERB   SA  OBJECT   RV  SA
                      RUBY                  SHOOT     OSWALD
```

3. Target structure for
"An octal number is a sequence of octal digits followed by a B."
(Note: the parse shown below corresponds to the reading where "followed" modifies "digits". The other reading, in which "followed" modifies "sequence", is normally surpressed by the parser as a permanent predictable ambiguity; see text, page 12).

```
DECOMPOSITION TREE:
 1. SENTENCE     = INTRODUCER  CENTER  ENDMARK
                               2.
 2. T-TVTENSE    = ASSERTION
                     3.
 3. ASSERTION    = SA  SUBJECT      SA  TENSE  SA  VERB  SA  OBJECT        RV  SA
                       4. NUMBER 5.                 BE      6. SEQUENCE 7.
 4. LN           = TPOS  QPCS  APOS  NSPOS  NPOS
                   AN
 5. TAPOS-ADJ    = ASSERTION
                     8.
 6. LN             TPOS  QPCS  APOS  NSPOS  NPOS
                   \
 7. TRNP           \SSERTION
                     9.
 8. ASSERTION    = SA  SUBJECT  SA  TENSE  SA  VERB  SA  OBJECT  RV  SA
                       HOST                  BE      OCTAL
 9. ASSERTION    = SA  SUBJECT  SA  TENSE  SA  VERB  SA  OBJECT  RV  SA
                       HOST                  BE      10.
10. PN           = LP  P    NSTGO
                       OF   DIGITS 11. 12.
11. TAPOS-ADJ    = ASSERTION
                     13.
12. T-RNVENPAS   = TPASSIVE
                     14.
13. ASSERTION    = SA  SUBJECT  SA  TENSE  SA  VERB  SA  OBJECT  RV  SA
                       HOST                  BE      OCTAL
14. TPASSIVE     = ASSERTION
                     15.
15. ASSERTION    = SA  SUBJECT  SA  TENSE  SA  VERB     SA  OBJECT  RV  SA
                       16. B                 FOLLOW      HOST
16. LN           = TPOS  QPOS  APOS  NSPOS  NPOS
                   A
```