

4

DTIC FILE COPY



RADC-TR-88-76
In-House Report
March 1988

AD-A196 636

TRANSFORMATIONAL GENERATIVE GRAMMAR: A Survey

Anthony R. Stevens, 1Lt, USAF

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

DTIC
ELECTE
JUL 01 1988
S D E

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

88 6 1 1988

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-88-76 has been reviewed and is approved for publication.

APPROVED:



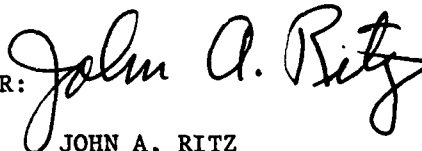
SAMUEL A. DINITTO, JR.
Chief, C2 Software Technology Division
Directorate of Command and Control

APPROVED:



RAYMOND P. URTZ, JR.
Technical Director
Directorate of Command and Control

FOR THE COMMANDER:



JOHN A. RITZ
Directorate of Plans & Programs

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (COES) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS N/A		
2a. SECURITY CLASSIFICATION AUTHORITY N/A			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) RADC-TR-88-76			5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A		
6a. NAME OF PERFORMING ORGANIZATION Rome Air Development Center		6b. OFFICE SYMBOL (if applicable) COES	7a. NAME OF MONITORING ORGANIZATION N/A		
6c. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700			7b. ADDRESS (City, State, and ZIP Code) N/A		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Rome Air Development Center		8b. OFFICE SYMBOL (if applicable) COES	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N/A		
8c. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO 62702F	PROJECT NO 5581	TASK NO 27
			WORK UNIT ACCESSION NO 30		
11. TITLE (Include Security Classification) TRANSFORMATIONAL GENERATIVE GRAMMAR: A Survey					
12. PERSONAL AUTHOR(S) Anthony R. Stevens, 1Lt, USAF					
13a. TYPE OF REPORT In-House		13b. TIME COVERED FROM Oct 87 TO Dec 87	14. DATE OF REPORT (Year, Month, Day) March 1988		15. PAGE COUNT 36
16. SUPPLEMENTARY NOTATION N/A					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD 12	GROUP 05	SUB-GROUP	Grammars, Phrase Structure Grammar Transformational Grammar, Natural Language Processing Generative Grammar, Artificial Intelligence		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
This report is a survey of past, current, and on-going research in the field of Generative Grammars. The paper assumes no previous knowledge in the field and includes an overview of phrase structure and transformational grammars. An overview is also given of the work done by Peters and Ritchie (1973) on Formalizing Transformations and is essential for an understanding of the field. in Computer					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL Anthony R. Stevens, 1Lt, USAF			22b. TELEPHONE (Include Area Code) (315) 330-4833		22c. OFFICE SYMBOL RADC (COES)

DD Form 1473, JUN 86

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

AIR FORCE 86145/ 10-F 78 - 4

UNCLASSIFIED

UNCLASSIFIED

Transformational Generative Grammar

The Theory of Transformational Grammar was introduced by Noam Chomsky in Syntactic Structures [Chomsky 1957] and revised in Aspects of the Theory of Syntax [Chomsky 1965]. The revised model is commonly referred to as the Standard Theory of Transformational Grammar, and has been extremely influential in the field of Linguistics. This paper explains the Theory of Transformational Grammar, surveys current work in the field, and identifies areas for further research.

Transformational grammars derived from phrase structure grammars, generalizing the notion of rewrite rules in order to handle the problem of discontinuous dependencies in a precise, uniform manner. A phrase structure grammar generates tree diagrams by a series of rewrite rules which indicate what lexical categories (parts of speech) make up larger categories. A transformational grammar adds transformational rules that operate on trees originally built by phrase structure rules to generate new trees. The important difference is that phrase structure grammars treat rules as constraints on the structure that can be assigned to a sentence; whereas, transformational grammars allow the application of new kinds of rules which transform structures in the course of a derivation, thereby creating new structures. An example of a transformational rule would be (from [Akmajian and Henry; 1975]):

Yes/No Question

To form a yes/no question, take a declarative sentence (statement) and move the first auxiliary to the left of the subject NP.

The generative paradigm models the derivation process of human language, succinctly capturing the generalized knowledge and rules humans use to create and understand utterances. Phrase structure analysis, on the other hand, is superficial in that it is determined solely by the ordering of elements within a sentence. This superficiality becomes evident when examining sentences which have similar syntactic structure. A person's intuitions about the similarity between sentences is not based solely on ordering. Winograd argues that this underlying similarity is recognized by the reader based on syntactic structure rather than on meaning. [Winograd;1983] If this is the case, then an analysis based on ordering alone is not sufficient.



By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

Phrase Structure Grammar

A phrase structure grammar is a 4-tuple (V, T, P, S) where V and T are disjoint finite sets, called the set of variables (or nonterminals) and terminals, respectively. In a grammar describing a human language, for example, terminals are words; the set of nonterminals might include symbols such as N and NP , representing a noun and a noun phrase, respectively. The symbol P represents a set of productions (rewrite rules) of the form $\alpha \rightarrow \beta$. The symbol S is a special symbol called the start symbol. Basic *phrase markers* (tree diagrams) are generated by rewrite rules of the sort:

(1a) $S \rightarrow NP (Aux) VP$

(1b) $NP \rightarrow N$

(1c) $VP \rightarrow V (S)$

(1d) $VP \rightarrow V (NP)$

Optional constituents are shown in parenthesis. Rule (1c) can be interpreted as VP (verb phrase) consists of V (verb) followed by an optional S (sentence). Note the recursive nature of (1a) and (1c). The PS (phrase structure) rule for (1a) expands to include a VP ; rule (1c), in turn, may include S as a constituent. Thus, an infinite set of structures can be generated from a finite set of rules, as in Figure 1.

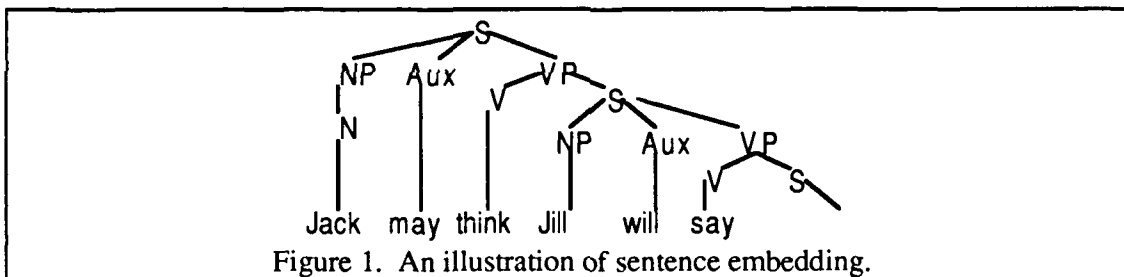
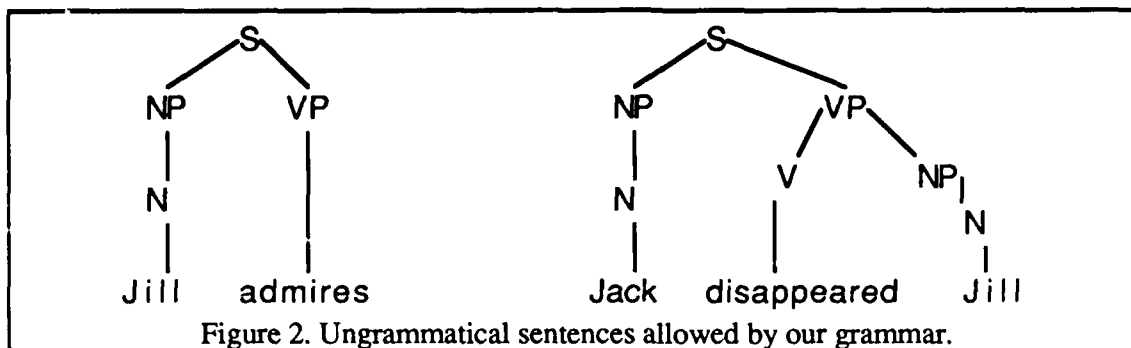


Figure 1. An illustration of sentence embedding.

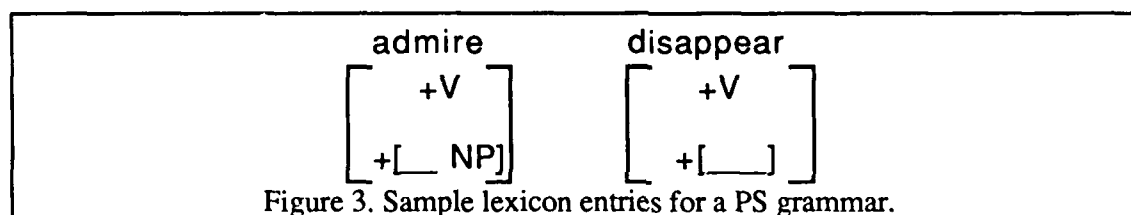
To be concise, we may abbreviate rules (1c) and (1d) as follows:

(2) $VP \rightarrow V ([S NP])$

The notation used in rule (2) means that a verb phrase is composed of a verb which may be followed by either a sentence or a noun phrase. So, parentheses indicate an option, as before, and braces indicate alternative choices. Let's investigate one of the problems with our PS grammar as described thus far. Our grammar does not allow us to make any distinctions about transitive verbs (those that require objects) and intransitive verbs (those that may not occur with an object). (See Figure 2.)



The verb *admires* is transitive; hence, the sentence Jill admires is ungrammatical, while sentences like Jill admires Jack are good. Also, the verb *disappear* is intransitive; therefore, sentences like Jack disappeared Jill are ungrammatical, while sentences like Jack disappeared are acceptable. One solution to this problem is to assume our grammar has access to a lexicon in addition to a set of rules for building trees. Sample entries in our lexicon would be:



The entries indicate a syntactic context for insertion into phrase markers. The syntactic category of both entries is verb. Contextual features are also indicated: the verb *admires* has the feature $+[\text{NP}]$ which means a NP must follow it, whereas the verb *disappear* has the feature $+[\text{ }]$ which means no NP may follow it. Other possibilities for syntactic features include: [1st person], [3rd person], [+ plural], and [- plural]. A "+" indicates that a word has that feature, and a "-" indicates that a word is without the feature, for example, $[+N, +\text{common}]$ indicates a common noun, $[+N, -\text{common}]$ indicates a proper noun, and [3rd person, -plural] indicates third person singular. Despite this convenient notation, phrase structure grammars still have several problems, described in the following section.

Inadequacy of Phrase Structure Grammars

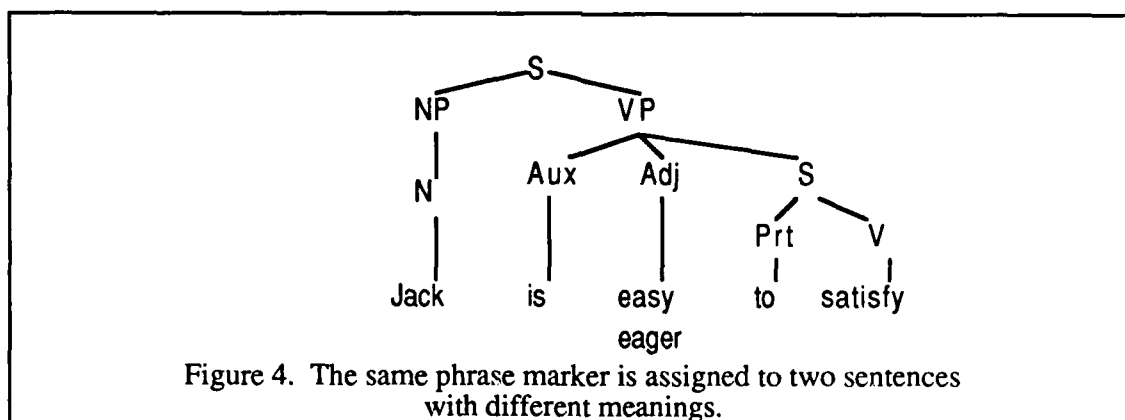
There are three basic problems with phrase structure grammars in representing all the significant aspects of language structure, as discussed in [Barr and Feigenbaum; 1981]:

1. PS grammars make the description of English unnecessarily clumsy and complex -- for example, in the treatment of conjunction, auxiliary verbs, and passive sentences. The important point here is that PS grammars fail to capture linguistically significant generalizations about the English language. For instance, the fairly complicated rule (4) is required to represent the simple generalization in (3). This is due, in part, to the special way in which the verbs 'have' and 'be' must be handled.

(3) In questions, the auxiliary verbs appear in the same relative order as in declarative sentences, but the first auxiliary verb occurs to the left of the subject.
[Akmajian and Heny; 1975].

$$(4) \quad S \longrightarrow \left\{ \begin{array}{l} \text{NP Aux} \\ \text{Modal NP (HAVE) (BE)} \\ \text{HAVE NP (BE)} \\ \text{BE NP} \end{array} \right\} VP$$

2. PS grammars assign identical phrase markers to sentences that have unique meanings, as in Figure 4. The difference in meaning can be attributed to a difference in underlying syntactic structure, as we will see later in this paper.



3. PS grammars provide no basis for identifying as similar the sentences that have different surface structures but much of their "meaning" in common.

A porcupine nibbled that elm.

That elm was nibbled by a porcupine.

My car isn't working.

The automobile that belongs to me is out of order.

Which elm did the porcupine nibble?

...the elm which the porcupine nibbled...

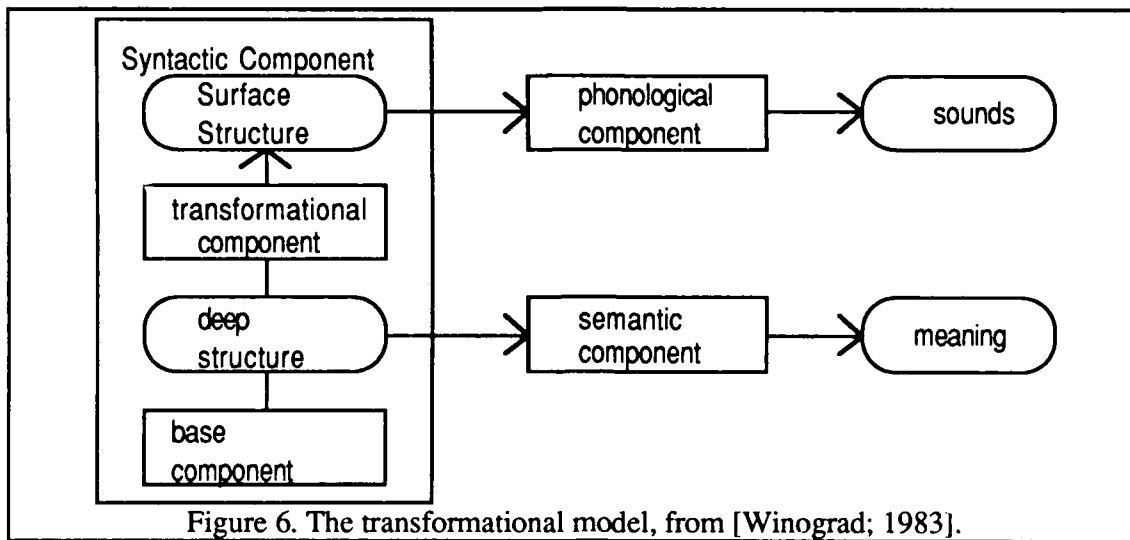
Figure 5. Different forms with underlying similarity, from
[Winograd; 1983].

Making the connection between the above sentences relies on understanding the underlying similarity of structure. Although the first pair of sentences are paraphrases, as are the second pair, it is understood by the reader that the first pair is more closely related than the second. Furthermore, another close connection is intuitively made between the sentences in the last pair, although these two sentences have entirely different functions -- one is a question, and the other is a phrase referring to an object.

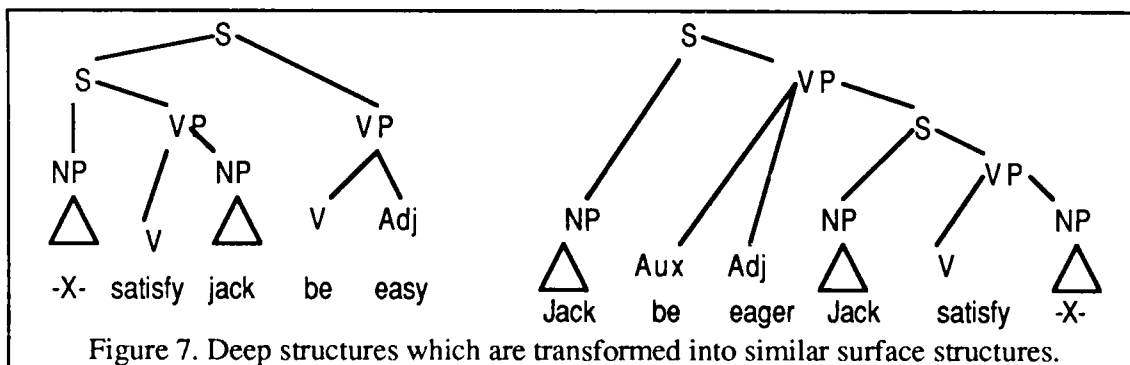
Motivation for Transformational Grammars

The examples discussed thus far suggest that some properties of sentences in *natural language cannot be accounted for by single phrase markers alone*, that is, in terms of relations between immediate words in a sentence that are connected in some sense but which, nevertheless, are not contiguous in the linear ordering of the words.

One way to account for discontinuous dependencies of this kind is to come up with a way by which two or more phrase markers can themselves be related to each other in some specified way. Akmajian points out that this is the fundamental insight of the Theory of Transformational Grammar. We are now ready to explore the Theory of Transformational Generative Grammars.



In the standard transformational model, depicted in Figure 6, the base component is a CFG (context-free grammar) that generates deep structures. The deep structure contains everything relevant to its meaning, for example, the deep structures for the surface structure shown earlier in Figure 4 would look something like those in Figure 7. The triangles indicate a portion of the tree which has been left unspecified, since it is not relevant to the example at hand.



The second component is the transformational component, consisting of a set of transformational rules that operate on phrase markers. The transformational component is used in a derivation process by which a deep structure is converted to a surface structure, which can then be used to produce an actual sequence of sounds or words in a sentence. The deep structures of Figure 7 would be converted by a sequence of transformations into the same surface structure (shown in Figure 4). The difference in meaning between the

two sentences can be attributed to the difference in their underlying structures. Conversely, a single deep structure can give rise to more than one surface structure, depending on which transformations the deep structure undergoes.

The transformational model is not a model of how language is produced but, rather, a model which formalizes the knowledge a person must have about the syntax of a particular language.

Transformational Grammar Explained

Before we define a TG (transformational grammar) formally, several features of TG will be explained. There is no standard, accepted notation for transformational rules, but a version described by Akmajian and Heny (1975) will serve our purpose.

A transformational rule consists basically of an SD (structural description), and an SC (structural change). An SD is a pattern which must be matched against a tree in the course of a derivation in order for the corresponding SC to take place. In addition to an SD, a rule may specify a set of conditions which must be met in order for the rule to fire. There are three *elementary transformations* which can appear in a structural change description: *deletion*, *substitution*, and *adjunction*. Adjunction can alternatively be *sister adjunction*, *daughter adjunction*, or *chomsky adjunction*. To illustrate these concepts, we begin by explaining the transformational rule of Dative Movement given below as rule (5).

(5) Dative Movement (optional)

SD: V - NP - {to, for} - NP
1 2 3 4

SC: 1+4 2 0 0

- a. Mary gave a book to the man.
- b. Mary gave the man a book.

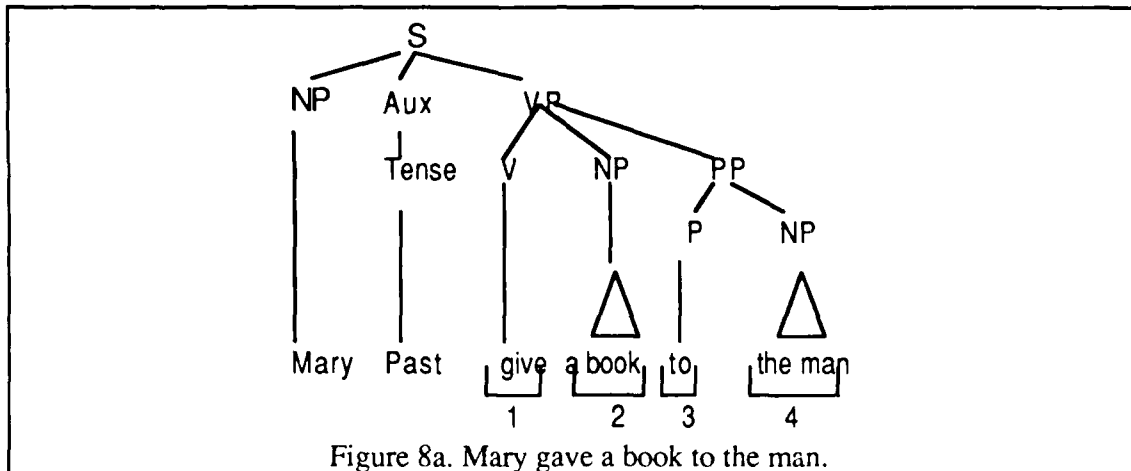


Figure 8a. Mary gave a book to the man.

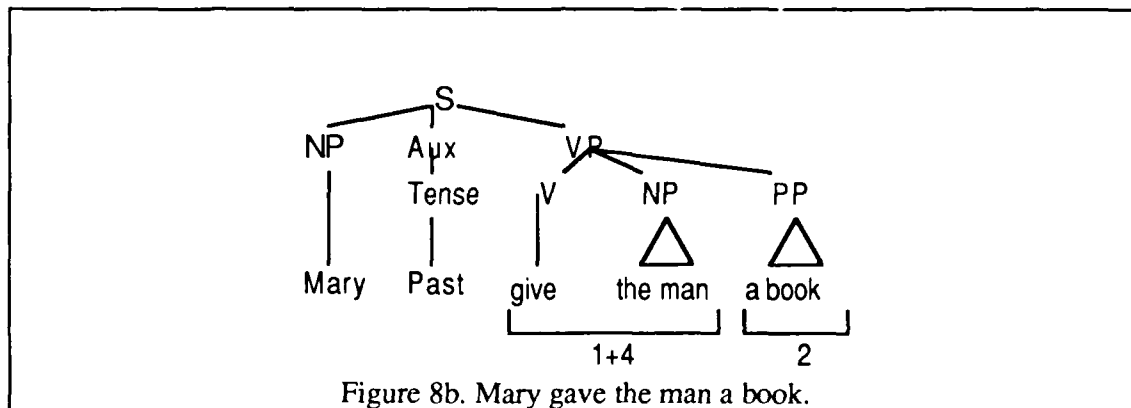


Figure 8b. Mary gave the man a book.

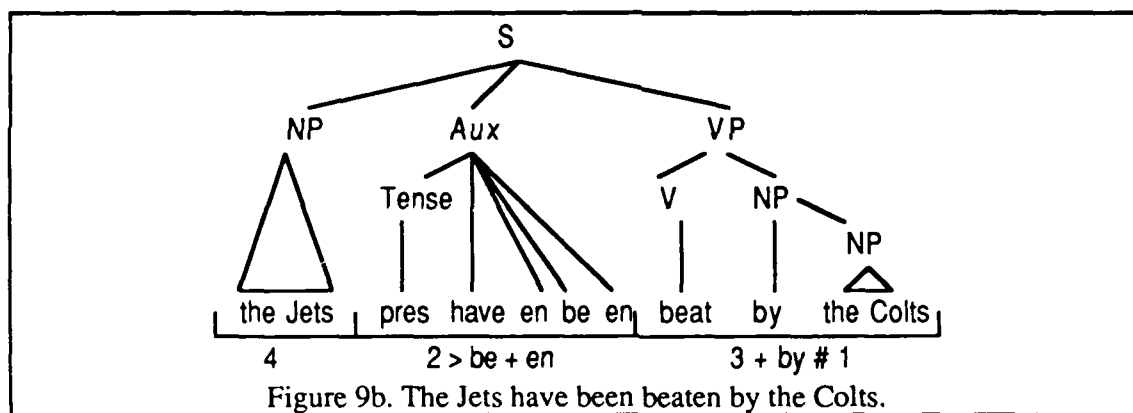
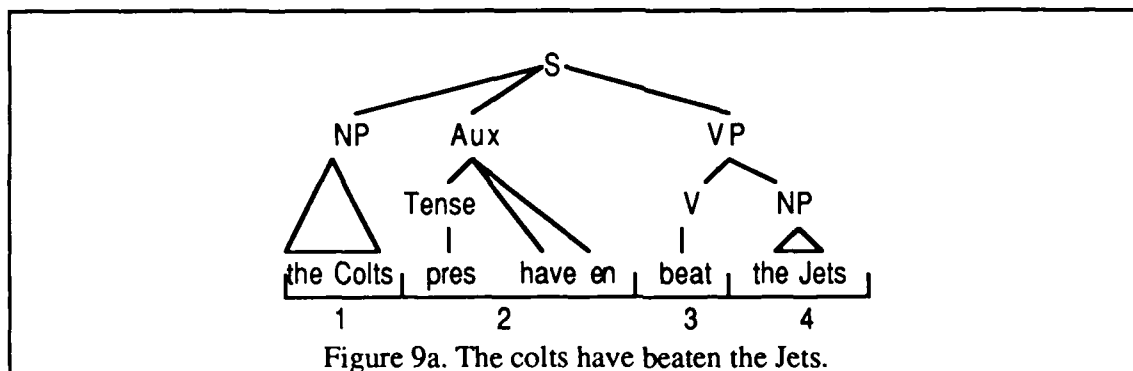
Dative movement applied to the phrase marker in Figure 8a gives Figure 8b. Since the SD matches the subtree of Figure 8a indicated by the numbered constituents, we can apply the SC, which derives the phrase marker shown in Figure 8b. Again, the constituents are numbered for ease of reference. The "+" in the SC of Rule (5) indicates *sister adjunction* is to be performed on the first and fourth constituents. The fourth term (the man) is to be placed immediately to the right of the first term (give). The first and fourth terms are now sisters to each other and are daughters of the same VP node, as shown in Figure 8b.

Referring again to the SC in Rule (5), we note that the second constituent remains as is. Continuing, the symbol "0" in the SC indicates deletion is to be performed for the corresponding term in the SD, so the third and fourth constituents are deleted.

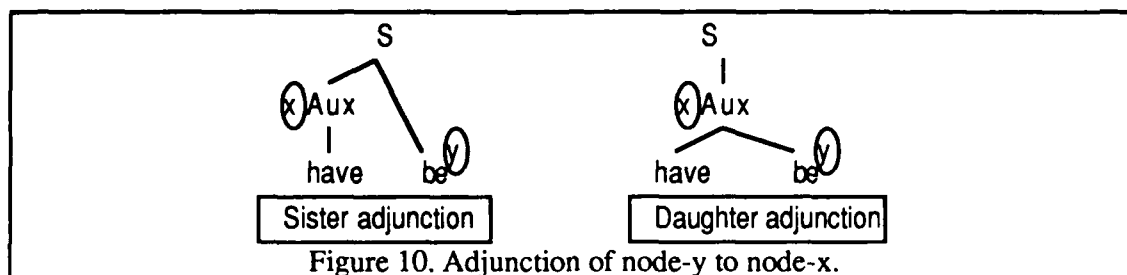
To illustrate *chomsky adjunction* and *daughter adjunction*, we will explain the Passive transformation.

(6) Passive (optional)

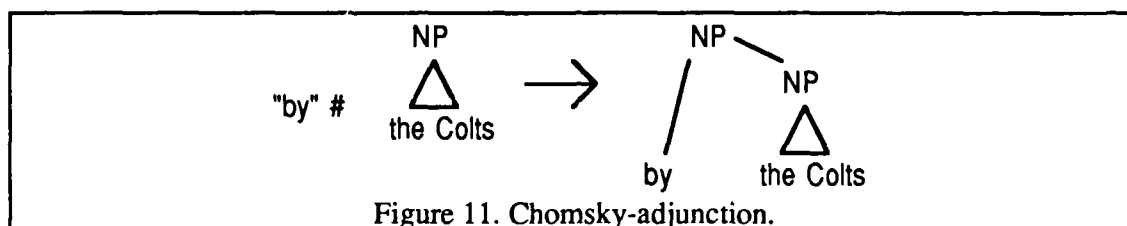
SD: NP - Aux - V - NP
 1 2 3 4
 SC: 4 2 > be+en 3 by # 1



First, term 4 is substituted for term 1. Next, we see by examining the SD that the second constituent of our new, derived structure is to be "2 > be + en". The symbol ">" indicates that "be" and "en" are to be daughter-adjoined as the rightmost daughters of "Aux" (term 2). Similarly, "<" indicates leftmost daughter-adjunction. Recall that sister adjunction of node-y to node-x implies making node-y a daughter of whatever node dominates node-x; whereas, daughter-adjunction of node-y to node-x merely means adding a descendant to node-x, as shown in Figure 10.

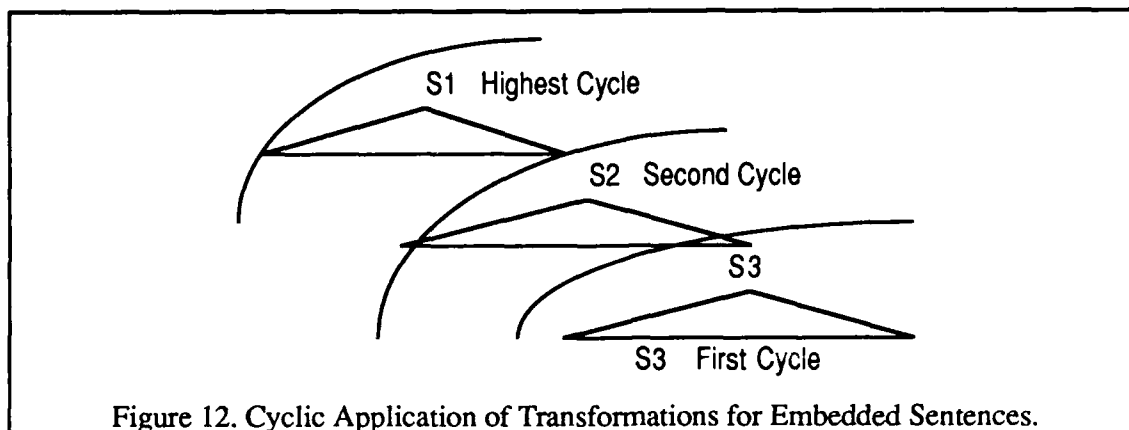


Our last elementary transformation is Chomsky-adjunction. Referring to Figure 11, Chomsky-adjunction consists of adjoining a new node with a node that is already there, in this case NP, as children of a new parent node whose label will be identical to the label of the node that was previously there. Substituting this derived structure for our original NP-node completes the transformation.



Ordering of Transformations

It turns out that individual transformational rules can interact with each other to derive complex surface structures in a straightforward way. Transformations are applied to deep structures in a specific linear order, based on dependencies between the rules. Furthermore, if a deep structure contains embedded sentences, the entire collection of transformations is applied in a cyclic fashion, first to the most deeply embedded sentence in a tree, then to the next higher sentence and so on. Referring to Figure 12, all the rules that could be applied would first apply to S3 in their proper order, then to S2, and then to S1.



Some transformations are optional and others are obligatory. Originally, housekeeping rules were obligatory and all others were optional; however, later versions of the grammar theory adopted a convention whereby all transformations are *meaning preserving*. In other words, a deep structure should capture all the meaning of a sentence, and a transformation should not change its meaning. For example, a transformation should not turn a statement into a question. In the case of yes/no questions, this meant that the very first PS rule should be modified as shown in Rule (7).

(7) $S \rightarrow (Q) NP Aux VP$

Then the Question transformation would be made obligatory. If the phrase marker Q is present, the Question transformation will apply.

(8) Question (Obligatory)

SD: Q - NP - Tense ({Modal, HAVE, BE})

1 2 3

SC: 1 3 2

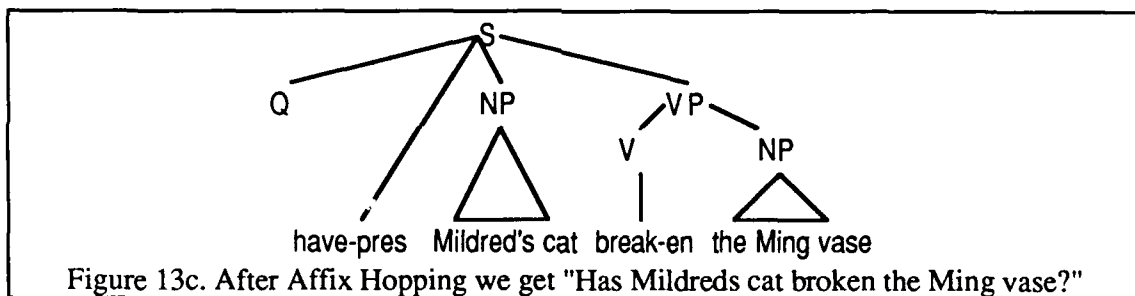
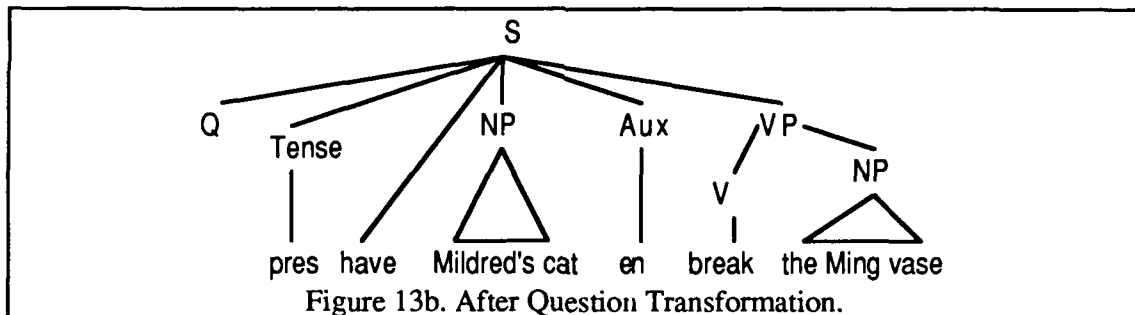
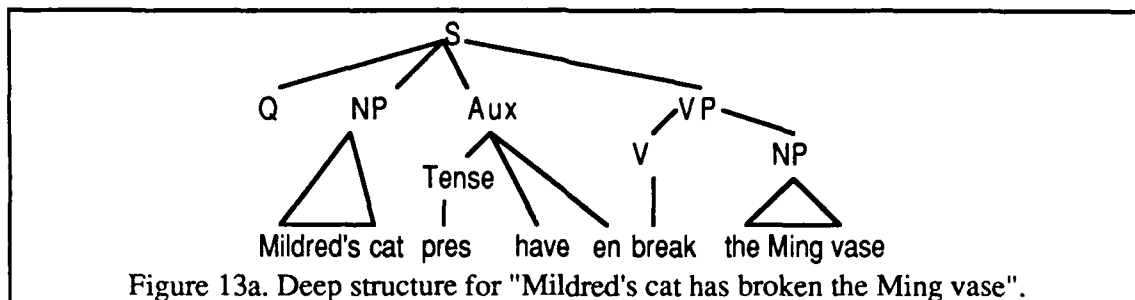
We are now ready to illustrate how two rules can be applied in sequence. First, we introduce another obligatory transformation, Affix Hopping:

(9) Affix Hopping (Obligatory)

SD: Affix - V

1 2

SC: 2 # 1



Let us begin with the deep structure shown in Figure 13a. Applying Rule (8), we get Figure 13b. Now, applying Rule (9) to this intermediary structure, the result is Figure 13c. At this point it is not clear why the Question transformation should be applied first. Actually, applying Affix Hopping first will work in some cases; however, for sentences containing a naked tense, if Affix Hopping is applied first, the tense is fronted along with the main verb, which may not be an auxiliary verb. If this happens, the Question transformation will not apply, as it applies only to structures which contain a NP followed by an auxiliary verb. For a detailed argument, see Akmajian and Heny (1975).

Before formalizing the notion of TG, consider the ordering of the Passive and Dative transformations discussed earlier. It can easily be proven that the Dative Movement rule must precede the Passive rule in order to generate the following sentences in a simple,

concise manner:

(10a) Mary gave a book to the man.

(10b) Mary gave the man a book.

(10c) A book was given to the man by Mary.

(10d) The man was given a book by Mary.

A complete ordering of all the transformational rules is given in Appendix A. The proof for ordering Dative Movement before Passive is as follows (from [Akmajian and Heny;1975]):

Proof by contradiction:

Assume the contrary; that is, assume the rules must be applied in the order: (1) Passive, (2) Dative Movement. As both rules are optional, we have four possibilities for generating the sentences given in (10). The four possibilities are:

	(i)	(ii)	(iii)	(iv)
(11) Passive	Applies	Applies	Doesn't	Doesn't
Dative	Applies	Doesn't	Applies	Doesn't

Clearly, (iv) yields (10a); our basic sentence results when neither rule is applied.

Option (iii), where only Dative Movement applies, yields sentence (10b).

Option (ii), where only Passive applies, yields sentence (10c).

Sentence (10d) is left to derive, with only the option left of applying both rules.

I. Apply Passive first, then Dative:

Mary	past	give	a book	to the man
NP	Aux	V	NP	

Passive: A book past be en give to the man by Mary

Dative: Does not apply, since there is no NP between "give" and "to".

= (10c) A book was given to the man by Mary

II. Apply Dative first, then Passive:

Mary	past	give	a book	to	the man
		V	NP	Prep	NP

Dative:	Mary	past	give	the man	a book
	NP	Aux	V	NP	

Passive: The man past be en give by Mary a book
 Affix Hopping: The man be-past give-en by Mary a book
 Extraposition: The man be-past give-en a book by Mary.
 = (10d) The man was given a book by Mary.

Thus, all the sentence of (10) can be derived by applying the rules (optionally) in the order: (1) Dative Movement, (2) Passive. Note: For Part II, additional transformations were required to obtain the final surface structure.

Formalization of Transformations

In a desire to allow the application of mathematical techniques to transformations, Peters and Ritchie, in their paper, "*On the Generative Power of Transformational Grammars*", provided general definitions modeling grammatical transformations as mappings on trees. These trees are described in their paper as *labeled bracketings*.

The notation used may appear a bit formidable to some readers; however, formalization of the notions presented earlier in this paper is the basis for all further work in the field and cannot be omitted. Care is taken to provide a basic overview without obscuring the concept. To that end, the formalism described in Peters and Ritchie is presented again here with considerably less detail and with explanations added where appropriate. Definitions are repeated from Peters and Ritchie (1973) with little or no modification where it is essential to be precise.

Let V_T and V_N be fixed, disjoint vocabularies of terminals and nonterminals, respectively. Let $L = \{ [A \mid A \in V_N] \}$ and $R = \{]A \mid A \in V_N \}$. Labeled bracketings are finite strings of symbols from $V_T \cup V_N \cup L \cup R$; terminal labeled bracketings from $V_T \cup L \cup R$. A well-formed labeled bracketing is one in which the brackets occur in (nested) matched pairs.

Definition 1. A string ϕ is a well-formed labeled bracketing if

- i) $\phi \in V_T \cup V_N$
- ii) $\phi = \phi\omega$, or
- iii) $\phi = [A \phi]A, A \in V_N$

where ϕ, ω are well-formed labeled bracketings.

Peters and Ritchie also define a debracketing function mapping labeled bracketings into strings of terminals and nonterminals as follows:

Definition 2. The debracketing function d on labeled bracketings is the mapping defined by setting

$$\begin{aligned} d(\alpha) &= \alpha \quad \text{if } \alpha \in V_T \cup V_N \\ &= \varepsilon \quad \text{if } \alpha \in L \cup R \end{aligned}$$

This is a simple homomorphism, since $d(\phi\psi) = d(\phi)d(\psi)$ for labeled bracketings ϕ, ψ .

A few examples are in order:

Examples (1)-(4). Take $V_N = \{S, NP, N, VP, V, ADJ\}$

$V_T = \{\text{Bruno, beer, had, a, Pilots, crashing, planes, are}\}$

then (1)-(3) are well-formed labeled bracketings, and (4) is not:

(1) Bruno had a beer

(2) [N [NP [N Bruno]N]NP]N

(3) [S [NP [N Pilots]N]NP [VP are [NP [ADJ crashing]ADJ
[N planes]N]NP]VP]S

(4) [V are]N

The debracketization of (2) is the terminal "Bruno". Although (2) is well-formed, it says twice that "Bruno" is a noun. Definition 3 eliminates this sort of redundancy.

Definition 3. A labeled bracketing ϕ is said to be *reduced* if there are no $A, \chi_1, \chi_2, \phi, \omega, \sigma, \tau$, such that either $\phi = \chi_1 [A]A \chi_2$, or

i) $\phi = \chi_1 [A \phi]A \chi_2$,

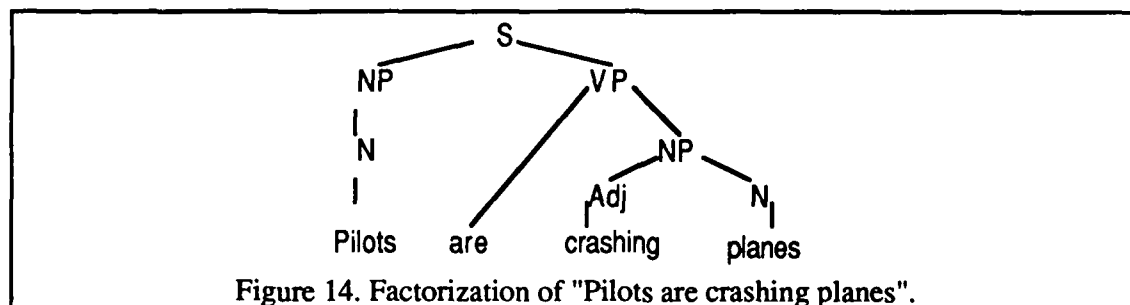
ii) $\phi = \sigma [A \omega]A \tau$, and

iii) ϕ and ω are well-formed, $\sigma \in L^*$ and $\tau \in R^*$.

For labeled bracketing (2) we have $\chi_1 = \chi_2 = \varepsilon$, $\omega = \text{Bruno}$, $\sigma = [NP$, $\tau =]NP$, and $A = N$, so a reduced labeled bracketing for (2) would be: $[NP [N \text{ Bruno}]N]NP$.

We can define the *interior* of a terminal labeled bracketing as the longest well-formed substring of a labeled bracketing retaining all its terminals. The interior of (2) is (2) itself, while the interior of (4) is "are". The left and right *exteriors* of (4) are $E_L(4) = [V$, $E_R(4) =]N$.

The corresponding notion of an interior for phrase markers can be illustrated as follows. A factorization of a phrase marker is induced by a factorization of its terminal string. Consider the phrase marker shown in Figure 14.



A factorization of the terminal string "Pilots are crashing planes" into the three substrings $X1=\text{Pilots}$, $X2X3=\text{are crashing}$, and $X4=\text{planes}$ induces the division of the phrase marker into 3 factors:

$$\left[\begin{array}{c} \text{NP} \\ | \\ \text{N} \\ | \\ \text{Pilots} \end{array} \right] \left[\begin{array}{cc} & \text{Adj} \\ \text{are} & | \\ & \text{crashing} \end{array} \right] \left[\begin{array}{c} \text{N} \\ | \\ \text{planes} \end{array} \right] \quad (12)$$

Each factor includes the highest node dominating only terminal symbols. This notion corresponds to that of an interior for labeled bracketings.

A standard factorization is defined as follows:

Definition 4. A *standard factorization* into n terms, for $n \geq 1$, of a terminal labeled bracketing ϕ is defined, if ϕ is a substring of a well-formed labeled bracketing, to be an (ordered) n -tuple $(\psi_1, \dots, \psi_n)$ of labeled bracketings such that

- (i) $\phi = \psi_1\psi_2\dots\psi_n$, and
- (ii) for each $i = 1, \dots, n$, the leftmost symbol of ψ_i is not a right bracket, nor is the rightmost symbol a left bracket. This assures us that the factors have been chosen to coincide with the phase breaks, and that each non-null factor contains terminals.

A standard factorization of Example (3) is

(13) $(\psi_1, \psi_2, \psi_3, \psi_4)$ where

$$\psi_1 = [S \ [NP \ [N \ \text{Pilots}]N \]NP$$

$\psi_2 = [\text{VP are,}$
 $\psi_3 = [\text{NP [Adj crashing]Adj, and}$
 $\psi_4 = [\text{NP [N planes]N]NP]VP]S.$

The standard factorization of (12) is $(\psi_1, \psi_2\psi_3, \psi_4)$.

The *contents* $C(\phi)$ of a terminal labeled bracketing (defined iff ϕ is a substring of a well-formed labeled bracketing) is the concatenation of the interiors of the terms of a unique factorization where each factor has an interior. Without repeating the formal definition here, a few examples will serve to illustrate the general notion. The contents of $\psi_2\psi_3$ is "are [Adj crashing]Adj", and of $\psi_2\psi_3\psi_4$ is $\psi_2\psi_3[\text{NP [N planes]N]NP , VP}$. Basically, we are identifying the longest well-formed substring of each individual factor, then we concatenate the result; we extract everything that is well-formed. We also use $R(\phi)$ to refer to the string of brackets remaining after $C(\phi)$ has been removed, for example, $R(\psi_2\psi_3)$ is $[\text{VP[NP, and } R(\psi_2\psi_3\psi_4)$ is $]S$.

We are now in a position to define the structural condition and transformational mapping for labeled bracketing notation. The corresponding notions for phrase markers are structural description and structural change. We begin by describing the three kinds of elementary transformations which are allowed in a transformational mapping: deletion of a certain factor, substitution of a certain factor by a sequence of other factors, and adjoining a sequence of factors to a given factor.

Definition 5. The *deletion elementary* is the function T_d from substrings of well-formed labeled bracketings to labeled bracketings defined by $T_d(\phi) = R(\phi)$.

The *substitution elementary* is the function T_s from pairs (ϕ, ψ) of substrings of well-formed labeled bracketings to labeled bracketings defined if and only if ϕ has an interior by setting $T_s(\phi, \psi) = E_l(\phi) C(\psi) E_r(\phi)$.

The *left-adjunction elementary* is the function T_l from pairs (ϕ, ψ) of substrings of well-formed labeled bracketings to labeled bracketings defined iff ϕ has an interior by setting

$$T_l(\phi, \psi) = E_l(\phi) [A_1 \dots [A_m C(\psi) I(\phi)] A_m \dots] A_1 E_r(\phi),$$

where $A_1, \dots, A_m$ is the longest sequence of nonterminals such that there is a well-formed labeled bracketing ω for which $I(\phi) = [A_1 \dots [A_m \omega] A_m \dots] A_1$,

allowing the case $m=0$ in which there are no such brackets.
 The *right-adjunction elementary* is defined parallel to left-adjunction.

A structural condition on a factorization is a Boolean combination of three kinds of predicates, where the factorization has n terms:

$A^n_{i \rightarrow j}$, $h \rightarrow i \equiv^n j \rightarrow k$, $i \rightarrow j \equiv^n x$,
 where $i \rightarrow j$ indicates a sequence of i th- j th factors,
 and x is a terminal string.

As an example, take the predicate $NP^9_{2 \rightarrow 2}$. This means that the second term of a factorization (which has 9 terms) must be a noun phrase in order for the structural condition to apply.

Without repeating the formal definition of structural condition given by Peters and Ritchie (1973), it will become clear how a transformation works using this new notation by carefully going through an example, slightly modified from their presentation. Consider the sentence: 'By whom had the call been put through to Chicago before John left?' The deep structure is:

Q wh+ Δ_a past have+en put through the call to Chicago by be+en before John left.
 where " Δ_a " is a dummy noun form indicating someone.

We can apply the Passive transformation which matches against the deep structure as follows:

W	NP	Aux	V	X	NP	Y	Passive	Z
1	2	3	4	5	6	7	8	9
1	6	3+8	4	5	0	7	2	9

- (1) Q
- (2) wh+ Δ
- (3) past have+en
- (4) put
- (5) through
- (6) the call
- (7) to Chicago by

(8) be+en

(9) before John left

The factorization of the deep structure is as follows (the interiors of each term have been underlined for clarity):

(1) [S [Pre Q]Pre

(2) [NP wh [NΔ]N]NP

(3) [PP [Aux [tense Past]tense [aspect [perf Have en] perf] aspect] Aux

(4) [VP [V Put]V

(5) [Prt through]Prt

(6) [NP [Det the]Det [N call]N]NP

(7) [Dir to Chicago]Dir [manner [agent [prep-p [prep by]prep

(8) [passive be en]passive]prep-p]agent]manner]VP

(9) [time before John left]time]PP]S.

The transformation Passive consists of a structural condition

$NP^9_{2 \rightarrow 2}$, $Aux^9_{3 \rightarrow 3}$, $V^9_{4 \rightarrow 4}$, $NP^9_{6 \rightarrow 6}$, $Passive^9_{8 \rightarrow 8}$, and

and a transformational mapping:

{ [T_s , (2,2),(6,6)], [T_r (3,3),(8,8)], [T_d (6,6)], [T_s (8,8),(2,2)] }

where T_s is the substitution elementary,

T_r is the right-adjunction elementary, and

T_d is the deletion elementary (refer to Definition 5).

Applying this transformation to the factorization of the deep structure yields:

(1) same as factor 1 above

(2) [NP [det the]det [N call]N]NP from applying [T_s (2,2),(6,6)]

(i.e. replace contents of term 6 with interior of term 2) See Definition 4.

(3) [PP [Aux [Aux ...]Aux] [passive be en]passive]Aux,

where $A_1 \dots A_m = \text{Aux}$,

[PP is the left exterior of factor 3,

[passive be en]passive is the contents of factor 8,

and [Aux...]Aux represents the interior of factor 3. See Definition 5.

- (4) same
- (5) same
- (6) the contents of factor 6 is 6 itself $R(\phi) = \epsilon$
- (7) same
- (8) [NP wh [NΔ]N]NP]prep-p]agent]manner]VP
- (9) same

The seventh factor gives us an opportunity to make a distinction between the interior and contents of a labeled bracketing. The interior of factor 7 does not exist since there is no well-formed substring of factor 7 containing all its terminals; whereas, the contents of factor 7 does exist. The contents of factor 7 is

[Dir to Chicago]Dir [prep by]prep,

extracting all well-formed substrings and concatenating them. Note also,

$R(7) = [\text{manner} [\text{agent} [\text{prep-p}.$

Peters and Ritchie go on to prove a very important theorem which is restated as Theorem 1. This theorem identifies the equivalence between transformational grammars and r.e. languages.

Theorem 1. Every recursively enumerable language is generated by some context-sensitive based transformational grammar, and conversely.

Areas for Further Research

A number of modifications and extensions have been proposed to the Standard Theory of Transformational Grammar. Some of these will be explored below.

Extended Standard Theory

The Standard Theory relies on the Katz-Postal Hypothesis (1964) which states that transformations were meaning preserving.

(14) Katz-Postal Hypothesis

Transformations are meaning-preserving, in the following sense: if two surface structures derive from exactly the same underlying structure and if their derivations differ only in that an optional transformation has applied in one but not the other, then they must have the same meaning.

Critics of the Standard Theory, however, were quick to point out that meaning was affected by the application of some transformations. Two apparent counterexamples to the hypothesis will suffice. (from [Akmajian and Henry; 1975])

(15a) John didn't leave the room, did he?

(15b) John left the room, didn't he?

(15c) Q - not - John - past - leave the room

(16a) Few people have read three of Hemingway's novels.

(16b) Three of Hemingway's novels have been read by few people.

The first pair of sentences both derive from the same underlying structure, namely (15c). They have undergone the same series of transformational rules, (i.e. Tag Formation, Negative Placement, Contraction, Question, etc.) yet do not have the same meaning. The only difference is that Negative Placement has placed not in the main clause of (15a) and in the tag of (15b). Sentence (15a) supposes John has not left the room and expects the answer "no"; whereas, sentence (15b) supposes John has left the room, and the answer is expected to be "yes". The second counterexample concerns derivations where the Passive transformation has been applied. Examining sentence (16b), it is clear that its meaning is slightly different than that of sentence (16a).

Examples such as these have led to the Extended Standard Theory. In this extended model, the semantic interpretation rules operate on the entire set of trees used in the derivation instead of extracting meaning only from the deep structure.

Generative Semantics

Proponents of this model made sweeping changes to the Standard Theory to handle the problems described above and other issues as well. The basic idea is that there is no separate semantic interpretation component; rather, both the semantic and syntactic representations are imbedded in phrase markers. The base structures, now, were not merely syntactic but were logical representations. The terminal nodes of the base structure were no longer words but semantically interpretable terms, similar to symbolic logic terms. Sentences (15) and (16) could now be handled by the fact that they would not derive from a single underlying structure but from two different logical forms.

Montague Grammar

The difference between Montague Grammar and the Standard Theory is in the treatment of the semantic structure. Rather than using the deep structure as input to a semantic component to produce meaning, Montague Grammar associates a semantic rule with each syntactic rule. Whenever a syntactic rule is applied to the syntactic structure, the semantic rule is applied to the corresponding logical structure. The formalism is extremely complex involving intensional logic and will not be described here.

Trace Theory

A modification of Extended Standard Theory, Trace Theory proposed that both the phonological and semantic components operate only on the surface structures, but the surface structure would now contain traces to capture the relevant information about meaning from the deep structure. As an example (from [Winograd; 1983]) consider the sentences (17).

- (17a) This is the oscilloscope Tom used to fix.
- (17b) This is the oscilloscope Tom used to fix the radio.
- (17c) Tom used the oscilloscope to fix the radio.

Here we see how Trace Theory can explain a phonological process. The same idea extends to semantic interpretation of similar surface structures for sentences with different meaning. In (17a), a reader contracts the phrase "used to" to "used"; whereas, the same contraction does not apply to sentence (17b). Why? It just so happens that contraction is blocked in precisely those cases where we might say "Something was deleted between the two words we are trying to contract." Referring to the underlying structure in (17c), we see that the phrase "the oscilloscope" occurs between "used" and "to". So, in Trace Theory, whenever we move a phrase, a *trace* is left behind. The surface structure retains this trace, and the contraction rule can be easily restated to block contraction across a trace marker.

Generalized Phrase Structure Grammar (GPSG)

Gerald Gazdar, of the University of Sussex, offered an extended interpretation of PS grammars, adding the notions of rule schemata and meta-rules which greatly reduce the work done by transformations. Rule schemata are patterns of rules. They present sets of rules, which have some common property, as a single statement. [Walter; 1986]

For example, the rule:

(18) $* \rightarrow * \text{ "and" } *$, where $*$ is any category,

represents

(19) $NP \rightarrow VP \text{ "and" } NP$

$VP \rightarrow VP \text{ "and" } VP$

$N \rightarrow N \text{ "and" } N$

A metarule creates new rules from rules which already exist. If a grammar contains the productions

(20) $VP \rightarrow V \ NP \ ___$

$VP \rightarrow V \ NP \ PP$

$VP \rightarrow V \ NP \ NP$

$VP \rightarrow V \ NP \ VP$

then the metarule

(21) $VP \rightarrow V \ NP \ \underline{W} \Rightarrow VP [pas] \rightarrow V \ \underline{W} \ PP$

creates the following new rules:

(22) $VP [pas] \rightarrow V \ __ \ PP$

$\rightarrow V \ \underline{PP} \ PP$

$\rightarrow V \ \underline{NP} \ PP$

$\rightarrow V \ \underline{VP} \ PP$

As in Montague Grammar, each syntactic rule has associated with it a semantic rule (which operates in parallel) to create a semantic representation of a sentence.

Lexical Functional Grammar (LFG)

An LFG consists of a context-free grammar and a dictionary. Equations are associated with each production in the grammar and with each entry in the dictionary. The derivation process works in three phases.

Phase 1. A phrase marker is generated. Then leaf nodes are assigned words from the dictionary. Next, all nodes (except those which are assigned words) are marked with unique variables.

Phase 2. Recall that each word and each production have an associated equation. In the second phase, these equations are instantiated and a functional description is produced, which is another set of equations.

Phase 3. Solving the set of equations produces a functional structure. One solution indicates a grammatical sentence. Two solutions indicate an ambiguous sentence. No solutions indicate an ungrammatical sentence (for details see [Winograd; 1983]).

Where do we go from here?

Robert Berwick, in his paper *Strong Generative Capacity/Weak Generative Capacity and Modern Linguistic Theories* (1984), gives a good review of the current state of research in Transformational Grammar theory. One use of mathematical analysis has been to diagnose grammatical formalisms as too powerful (allowing too many grammars). For instance, Peters and Ritchie's demonstration that the theory of TG could specify any recursively enumerable set was thought by some to indicate that TG were too powerful. Berwick states, "A theory that is too powerful could generate either unnatural tree structures (and so be too powerful in terms of strong generative capacity) or it could generate unnatural sentences (and be too powerful in terms of weak generative capacity)."

We want our theory to describe all and only the natural languages. A more restricted formalism is desired if we are to learn about how humans actually derive language (rule systems underlying linguistic behavior) or how different languages interact. We cannot infer too much from a grammar which allows us to specify an arbitrary Turing Machine computation (see Figure 15), since it is generally accepted that Turing Machines are able to specify any language.

At the other end of the complexity hierarchy (see Figure 15), it has been shown that many reasonable questions about regular languages are solvable. To name a few, membership, inclusion, equivalence, infiniteness, and emptiness have all been shown to be solvable for regular languages. Many similar questions about the other classes of languages are either unsettled or unsolvable. Language theory concepts such as nondeterminism and the complexity hierarchy depicted in Figure 15 allow us to prove lower bounds on the inherent complexity of certain practical problems [Hopcroft and Ullman;1979]. We can also use Automata Theory to show that certain problems are unsolvable, by showing that

the problem is equivalent to another one which has already been shown to be Turing Machine unsolvable.

Berwick demonstrates that the excess power in Chomsky's Aspects Theory comes from unbounded deletion. In fact, he claims that all the proofs demonstrating the power of the Aspects Theory use this erasing power to delete strings of arbitrary length. Berwick shows that proofs given independently by Peters and Ritchie (1973), Kimball (1967), and Salomaa (1971) all use unbounded deletion in order to demonstrate that Transformational Grammars can generate any r.e. set. Berwick contrasts this property of the older theories with modern Government Binding theory, stating that current theories allow only a linear amount of erasing.

The trend, then, is toward a more restrictive formalism. Research remains to be done in terms of both strong and weak generative capacity of Transformational Grammars. If a particular theory generates too many languages, something can be learned from finding out what the source of its excess capacity is.

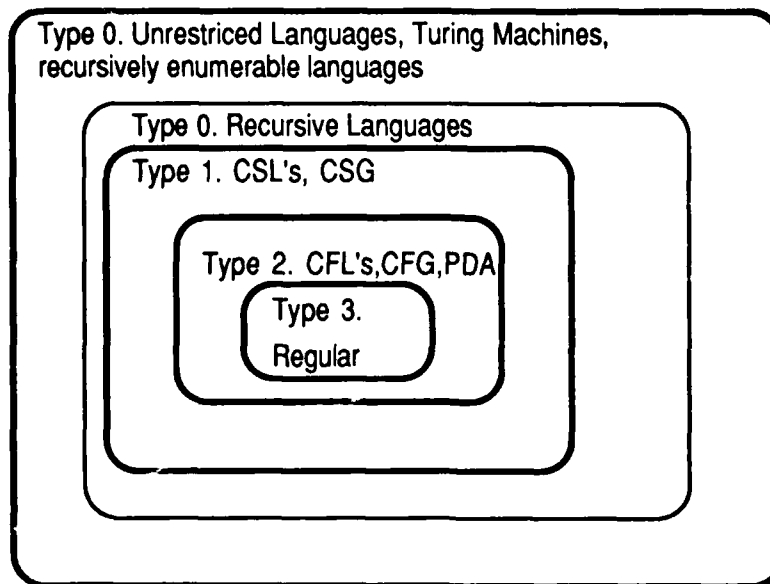


Figure 15. The Languages under Σ^* .

Note: A language accepted by a TM is called a recursively enumerable set if it halts on certain inputs and may not halt on others. A recursive set is accepted/generated by at least one TM that halts on all of its inputs.

Appendix A: List of Ordered Rules

The following is the ordered list of transformations from [Akmajian and Heny; 1975]. A line connecting two rules indicates that an explicit argument for their displayed ordering exists. Although there is not an argument for every pair, the combination of all the pairwise orderings leads to a total ordering.

1. Dative Movement	(Optional)	
2. Equi NP Deletion (Equi)	(Obligatory)	
3. Raising to Object	(Obligatory)	
4. Raising to Subject	(Obligatory)	
5. <u>For</u> Deletion	(Obligatory)	
6. Passive	(Optional)	
7. Agent Deletion	(Optional)	
8. Reflexivization	(Obligatory)	
9. Extraposition	(Optional)	
10. <u>It</u> Deletion	(Obligatory)	
11. Number Agreement	(Obligatory)	
12. <u>There</u> Insertion	(Optional)	
13. Tag Formation	(Optional)	
14. Negative Placement	(Obligatory)	
15. Contraction	(Optional)	
16. Subject-Auxiliary Inversion	(Obligatory)	
17. WH Fronting	(Obligatory)	
18. Affix Hopping	(Obligatory)	
19. <u>Do</u> Support	(Obligatory)	

References

- Akmajian, Adrian, Richard A. Demers and Robert M. Harnish, Linguistics: An Introduction to Language and Communication, Cambridge, MA: MIT Press, 1984.
- Akmajian, Adrian and Frank W. Heny, An Introduction to the Principles of Transformational Syntax, Cambridge, MA: MIT Press, 1975.
- Barr, Avron and Edward Feigenbaum (Eds.), The Handbook of AI: Volume 1, Los Altos, CA: William Kaufmann, Inc., 1981.
- Hopcroft, John and Ullman, Jeffrey, Introduction to Automata Theory, Languages, and Computation, Reading, MA: Addison-Wesley, 1979.
- Kimball, J. 1967 Predicates Definable by Transformational Derivations by Intersection with the Regular Languages. *Info and Control* 11:177-195
- Peters, P.S. and Ritchie, R.W. 1971 On Restricting the Base Component of Transformational Grammars. *Info and Control* 18:483-501.
- Peters, S. and Ritchie, R. 1973 On the Generative Power of Transformation Grammars. *Information Sciences* 6:49-83.
- Salomaa, A. 1971 The Generative Capacity of Transformation Grammars of Ginsburg and Partee. *Info and Control* 18:227-232.
- Walter, Sharon. 1986 Natural Language Processing: A Tutorial. RADC-TR-86-110 In-House Report. Rome Air Development Center, Griffiss AFB, NY 13441.
- Winograd, Terry, Language as a Cognitive Process, Reading, MA: Addison-Wesley, 1983.