STRING-TREE CORRESPONDENCE GRAMMAR: A DECLARATIVE GRAMMAR FORMALISM FOR DEFINING THE CORRESPONDENCE BETWEEN STRINGS OF TERMS AND TREE STRUCTURES

YUSOFF ZAHARIN
Groupe d'Etudes pour la Traduction Automatique
B.P. n° 68
Université de Grenoble
38402 SAINT-MARTIN-D'HERES
FRANCE

ABSTRACT

The paper introduces a grammar formalism for defining the set of sentences in a language, a set of labeled trees (not the derivation trees of the grammar) for the representation of the interpretation of the sentences, and the (possibly non-projective) correspondence between subtrees of each tree and substrings of the related sentence. The grammar formalism is motivated by the linguistic approach (adopted at GETA) where a multilevel interpretative structure is associated to a sentence. The topology of the multilevel structure is 'meaning' motivated, and hence its substructures may not correspond projectively to the substrings of the related sentence.

Grammar formalisms have been developed for various purposes. Generative-Transformational Grammars, General Phrase Structure Grammars, Lexical Functional Grammar, etc. were designed to be explanatory models for human language performance, while others like the Definite Clause Grammars were more geared towards direct interpretability by machines. In this paper, we introduce a declarative grammar formalism for the task of establishing the relation between on one hand a set of strings of terms and on the other a set of structural representations - a structural representation being in a form amenable to processing (say for translation into another language), where all and only the relevant contents or 'meaning' (in some sense adequate for the purpose) of the related string are exhibited. The grammar can also be interpreted to perform analysis (given a string of terms, to produce a structural representation capturing the 'meaning' of the string) or to perform generation (given a structural representation, to produce a string of terms whose meaning is captured by the said structural representation).

It must be emphasised here that the grammar writer is at liberty (within certain constraints) to design the structural representation for a given string of terms (because its topology is independent of the derivation tree of the grammar), as well as the nature of the correspondence between the two (for example, according to certain linguistic criteria). The grammar formalism is only a tool for expressing the structural representation, the related string, and the correspondence.
The formalism is motivated by the linguistic approach (adopted at GETA) where a multilevel interpretative structure is associated to a sentence. The multilevel structure is 'meaning' motivated, and hence its substructures may not correspond projectively to the substrings of the related sentence. The characteristic of the linguistic approach is the design of the multilevel structures, while the grammar formalism is the tool (notation) for expressing these multilevel structures, their related sentences, and the nature of the correspondence between the two. In this paper, we present only the grammar formalism ; a discussion on the linguistic approach can be found in [Vauquois 78] and [Zaharin 87].

For this grammar formalism, a structural representation is given in the form of a labeled tree, and the relation between a string of terms and a structural representation is defined as a mapping between elements of the set of substrings of the string and elements of the set of subtrees of the tree : such a relation is called a string-tree correspondence. An example of a string-tree correspondence is given in fig. 1.

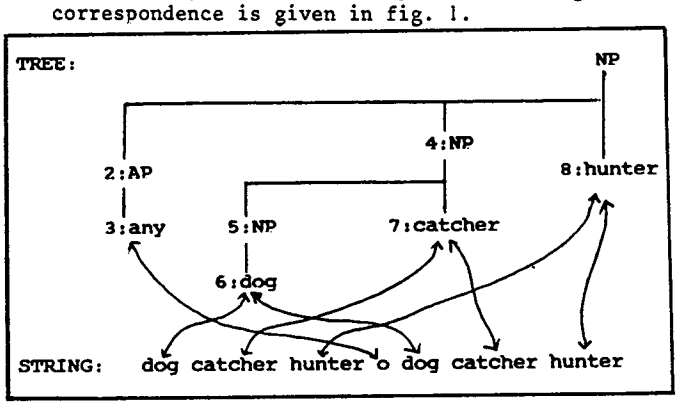


Fig.1 - A string-tree correspondence.

The example is taken from [Pullum 84] where he called for a 'simple' grammar which can analyse/generate the non-context free sublanguage of the African language Bambara given by :

$$L = \{\underline{X} \circ \underline{X} \mid \underline{X} \text{ in } N^* \text{ for some set of nouns } N, |N| \geq 1\}$$

and at the same time the grammar must produce a 'linguistically motivated' structural representation for the corresponding string of words. For instance, the noun phrase "dog catcher hunter o dog catcher hunter" means "any dog catcher hunter" and so the structural representation should describe precisely that.

In the string-tree correspondence in fig. 1, there are three concepts involved : the TREE which is a labeled tree taking the role of the structural representation, the STRING which is a string of terms, and finally the correspondence which is a mapping (given by the arrows ⟷) defined between substrings of STRING and subtrees of TREE (a more formal notation using indices would be less readable for demonstrational purposes). In the TREE, a node is given by an identifier and a label (eg. 1:NP). To avoid a very messy diagram, in fig. 1 we have omitted the other subcorrespondence between substrings and subtrees, for example between the whole TREE and the whole STRING (trivial), between the subtree 4(5(6),7) and the two occurrences of the substring "dog catcher" (non-trivial), etc. We shall do the same in the rest of this paper. (Then again, this is the string-tree correspondence we wish to express for our examples - recall the remark earlier saying that the grammar writer is at liberty to define the nature of the string-tree correspondence he or she desires, and this is done in the rules, see later). We also note that the nodes in the TREE are simply concepts in the structural representation and thus the interpretation is independent of any grammar that defines the correspondence (in fact, we have yet to speak of a grammar) ; for instance, the TREE in fig. 1 does not necessitate the presence of a rule of the form "AP NP hunter → NP" to be in the grammar.

A more complex string-tree correspondence is given in fig. 2 where we choose to define a structural representation of a particular form for each string in the language $a^n b^n c^n$. Here, the case for n=3 is given. The problem is akin to the 'respectively' problem, where for a sentence like "Peter, Paul and Mary gave a book, a pen and a pencil to Jane, Elisabeth and John respectively", we wish to associate a structural representation giving the 'meaning' "Peter gave a book to Jane, Paul gave a pen to Elisabeth, and Mary gave a pencil to John".

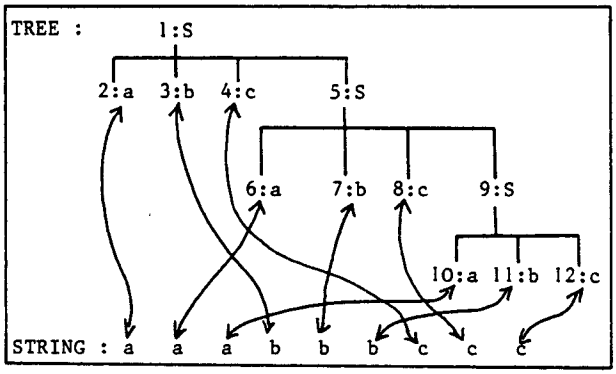


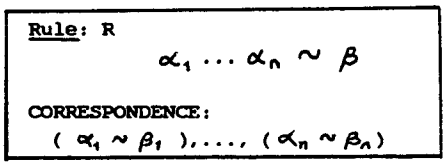
Fig. 2 - A non-projective string-tree correspondence for $a^n b^n c^n$

At this point, again we repeat our earlier statement that the choice of such structural representations and the need for such string-tree correspondence are not the topics of discussion in this paper.

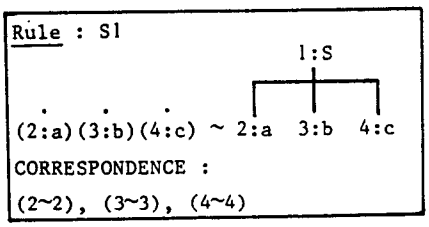
The aim of this paper is to introduce the tool, in the form of a grammar formalism, which can define such string-tree correspondence as well as be interpretable for analysis and for generation between strings of terms and structural representations.

The grammar formalism for such a purpose is called the String-Tree Correspondence Grammar (STCG). The STCG is a more formal version of the Static Grammar developed by [Chappuy 83] [Vauquois & Chappuy 85]. The Static Grammar (shortly later renamed the Structural Correspondence Specification Grammar), was designed to be a declarative grammar formalism for defining linguistic structures and their correspondence with strings of utterances in natural languages. It has been extensively used for specification and documentation, as well as a (manual) reference for writing the linguistic programs (analysers and generators) in the machine translation system ARIANE-78 [Boitet-et-al 82]. Relatively large scale Static Grammars have been written for French in the French national machine translation project [Boitet 86] translating French into English, and for Malay in the Malaysian national project [Tong 86] translating English to Malay ; the two projects share a common Static Grammar for English (naturally). The STCG derives its formal properties from the Static Grammar, but with more formal definitions of the properties. In the passage from the Static Grammar to the STCG, the form as well as some other characteristics have undergone certain changes, and hence the change to a more appropriate name. The STCG first appeared in [Zaharin 86], where the formal definitions of the grammar are given (but under the name of the Tree Correspondence Grammar).

A STCG contains a set of correspondence rules, each of which defines a correspondence between a structural representation (or rather a set or family of) and a string of terms (similarly a set or family of). Each rule is of the form :



The simplest form of such a rule is when $\alpha_1, \ldots \alpha_n$ are terms and β is a tree. The rule then states that the string of terms $\alpha_1, \ldots, \alpha_n$ corresponds ($\sim$) to the tree β, while the entry CORRESPONDENCE gives the substring-subtree correspondence between the terms $\alpha_1, \ldots, \alpha_n$ and the subtrees $\beta_1, \ldots, \beta_n$ of β. An example is given by rule S1 below which defines the string-tree correspondence in fig. 3.



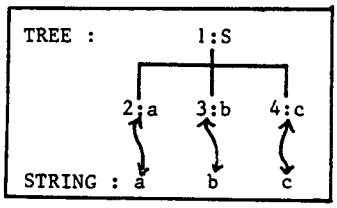


Fig. 3 - Correspondence defined by S1

Although in the example in fig. 3 above, the leaves of the TREE are labeled and ordered exactly as the terms in the STRING, this is not obligatory. For example, it is indeed possible to change the label of node 2 to something else, or to move the node to the right of node 4, or even to exclude the node altogether. In short, the string-tree correspondence defined by a rule need not be projective.

Such elementary rules $\alpha_1 \ldots \alpha_n \sim \beta$ (with $\alpha_1, .., \alpha_n$ terms) can be generalised to a form where each α_i $(i=1,..,n)$ represents a string of terms, say $\underline{A}_i$. Here, generalities can be captured if α_i specifies the name of a rule which defines a string-tree correspondence $\underline{A}_i \sim T_i$ (for some tree T_i given in the said rule, but it is of little significance here), in which case the interpretation of the string-tree correspondence defined by $\alpha_1 .. \alpha_n \sim \beta$ is taken to be $\underline{A}_1 .. \underline{A}_n \sim \beta$ (here $\underline{A}_1 .. \underline{A}_n$ means the concatenation of the strings $\underline{A}_1, .., \underline{A}_n$). The substring-subtree correspondence will still be given by the entry CORRESPONDENCE. Fig. 4 illustrates this.

The alternative to the above is to give each α_i in terms of a tree (ie. without reference to any rule), but then there is no guarantee that this tree will correspond to some string of terms. Even if it does, one cannot be certain that it would be the string of terms one wishes to include in the rule - after all, two entirely different strings of terms may correspond to the same tree (a paraphrase) by means of two different rules.

We shall discard the alternative and adopt the first approach. The generalised rule $\alpha_1, .. \alpha_n \sim \beta$ (with each α_i being the name of a rule) can be extended further by letting α_i be a list of rule names, where this is interpreted as a choice for the string-tree correspondence $\underline{A}_i \sim T_i$ to be referred to, and hence the choice for the string of terms $\underline{A}_i$ represented by α_i. In such a situation, it may also be possible that we wish the topology of the tree β to vary according to the choice of $\underline{A}_i$, and this variation to be in terms of the subtrees of the tree T_i. For these reasons, we specify each α_i as a pair (REFERENCE, STRUCTURE) where REFERENCE is the said list of rule names and STRUCTURE is a tree schema containing variables, such that the structure represents the tree found on the right hand side of the "~" in each rule referred to in the list REFERENCE. This way, the tree β can be defined in terms of T_i by means of the variables (for example those appearing simultaneously in both α_i and β). See the example later in fig. 5 for an illustration.

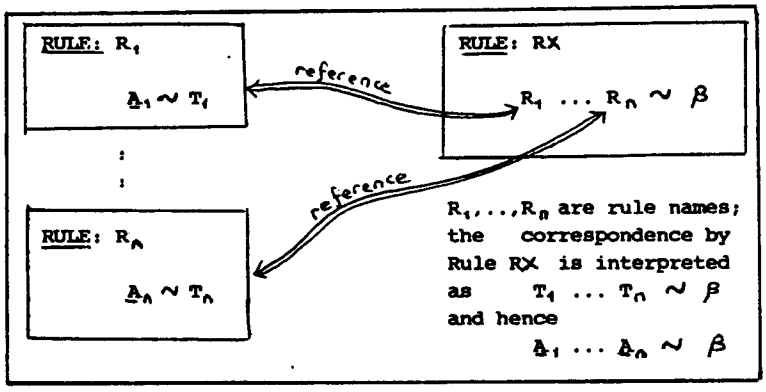
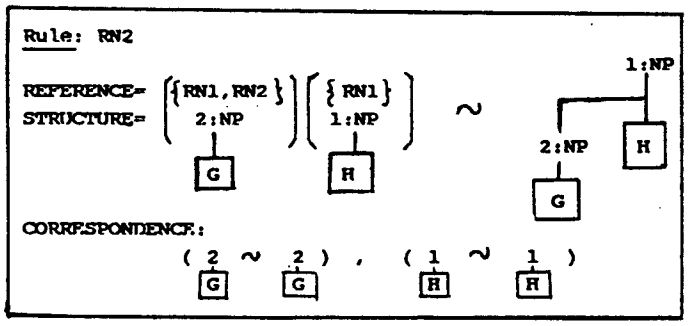
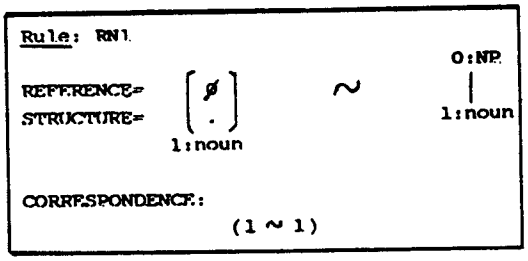


Fig.4 - String-tree correspondence with reference to other rules

Rules RN1 and RN2 below are examples of STCG rules in the form discussed above, where RN2 refers to RN1 and itself. Variables in the entry STRUCTURE are given in boxes, eg. [F], where each variable can be instantiated to a linear ordered sequence of trees. For a given element (REFERENCE, STRUCTURE), the instantiations of the variables in STRUCTURE can be obtained only by identifying (an operation intuitively similar to the standard notion of unification - again, see later in fig. 5) the STRUCTURE with the right hand side of a rule given in the entry REFERENCE.

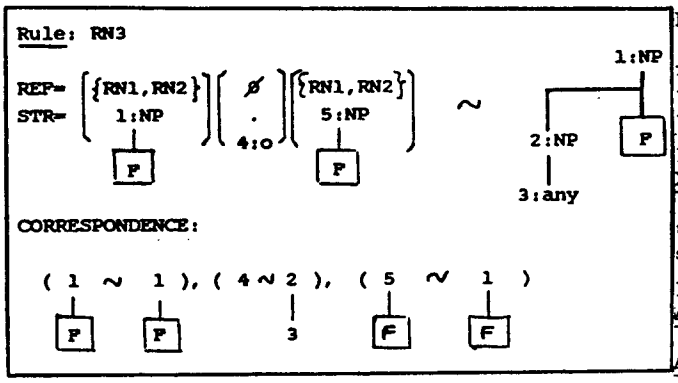




As an immediate consequence to the above, an STCG rule thus defines a correspondence between a set of strings of terms on one hand and a set of trees on the other (by means of a linear sequence of sets of trees). The rule RN1 describes a correspondence between a single term and a tree containing a node NP dominating a single leaf (for example, it gives the respective structural representations for "dog", "catcher", etc.). The rule RN2 describes a correspondence between two or more terms and a single tree - note the recursive REFERENCE in the first element of RN2 (for example, it gives the structural representation for "catcher hunter" as well as for "dog catcher hunter", see later in fig. 5).

The entry STRUCTURE of an element may also act as a constraint by making explicit certain nodes in the STRUCTURE instead of just a node dominating a forest (we have no examples for this in this paper, but one can easily visualise the idea). This means that the entry STRUCTURE of an element α_i = (REFERENCE, STRUCTURE) in a rule $\alpha_1..\alpha_n \sim \beta$ is also a constraint on the trees in T_i, and hence on the strings in $\underline{A}_i$ (as $\underline{A}_i$ and T_i are now sets), in a correspondence $\underline{A}_i \sim T_i$ defined by a rule referred to by α_i in its entry REFERENCE. Whenever it is made use of, such a constraint ensures that only certain subsets of T_i, and hence of $\underline{A}_i$, are referred to and used in the correspondence described by $\alpha_1..\alpha_n \sim \beta$.

The string-tree correspondence in fig. 1 is defined by rule RN3 below, which refers to rules RN1 and RN2. We show how this is done in fig. 5. Note that if two variables in a single rule have the same label, then their instantiations must be identical. The concept of derivation as well as the derivation tree have been defined for the STCG [Zaharin 86], but it would be too long to explain them here. Instead, we shall use a diagram like the one in fig. 5, which should be quite self-explanatory.



Going back to fig. 2 where the string-tree correspondence for $a^3b^3c^3$ is given, each substructure below a node S in the TREE corresponds to a substring "abc", but the terms in this substring are distributed over the whole STRING. In general, in a string-tree correspondence $\underline{A}_1..\underline{A}_n \sim \beta$ defined by a rule $\alpha_1..\alpha_n \sim \beta$, it is possible that we wish to define a substring-subtree correspondence of the form $\underline{A}_{j_1}..\underline{A}_{j_m} \sim \beta_k$, where $\underline{A}_{j_1},..,\underline{A}_{j_m}$ are disjoint substrings of the string $\underline{A}_1...\underline{A}_n$ and β_k is a subtree of β, and that β_k cannot be expressed in terms of the respective structural representations (if any) of $\underline{A}_{j_1},..,\underline{A}_{j_m}$. Such a correspondence cannot be handled by a rule of the form discussed so far because a structural representation (STRUCTURE) found on the left hand side can correspond only to a unit (connected) substring.

We can overcome this problem by allowing a rule to define a subcorrespondence between a substructure in the TREE (in the RHS) and a disjoint substring in the STRING (in the LHS), where this subcorrespondence is described in another rule (ie. using a reference - SUBREFERENCE - for a substructure in the TREE, rather than uniquely for the elements in the LHS). One also allows elements in the LHS to be given in terms of variables which can be instantiated to substrings. Rule S2 (after fig. 5) gives an example of such a rule where $\underline{X},\underline{Y},\underline{Z}$ are variables.

The rule S2 is of the following general type. (Recall that we wish to define a substring-subtree correspondence of the form $\underline{A}_{j_1}...\underline{A}_{j_m} \sim \beta_k$, where $\underline{A}_{j_1},..,\underline{A}_{j_m}$ are disjoint substrings of the string $\underline{A}_1..\underline{A}_n$ and β_k is a subtree of β, and that β_k cannot be expressed in terms of the respective structural representations (if any) of $\underline{A}_{j_1},..,\underline{A}_{j_m}$). In the rule $\alpha_1..\alpha_n \sim \beta$, the elements $\alpha_1,..,\alpha_n$ are to be as before except for those representing the substrings $\underline{A}_{j_1},...,\underline{A}_{j_m}$ which are to be left as unknowns, written say $\underline{X}_{j_1},..,\underline{X}_{j_m}$ respectively. The correspondence $\underline{A}_{j_1}..\underline{A}_{j_m} \sim \beta_k$ is to be written in the entry CORRESPONDENCE as $\underline{X}_{j_1}..\underline{X}_{j_m} \sim \beta_k$, and this is given a reference in an entry SUBREFERENCE as a means to define the correspondence elsewhere in another rule. In this SUBREFERENCE, if a rule $\alpha'_1..\alpha'_p \sim \beta$ is a possibility, the identification between the sequences $\underline{X}_{j_1}..\underline{X}_{j_m}$ and $\alpha'_1..\alpha'_p$ must be given. The interpretation of the rule is that the SUBREFERENCE gives a string-tree correspondence $\underline{A}'_1..\underline{A}'_p \sim \beta'$ which precisely defines the string-tree correspondence $\underline{A}_{j_1}..\underline{A}_{j_m} \sim \beta_k$, where β_k is identified with β' and $\underline{A}_{j_1}..\underline{A}_{j_m}$ is identified with $\underline{A}'_1..\underline{A}'_p$ with the separation points being obtained from the predetermined identification between $\underline{X}_{j_1}..\underline{X}_{j_m}$ and $\alpha'_1..\alpha'_p$ mentioned above.

A STCG containing the rules S1 and S2 defines the language $a^nb^nc^n$, and associates a structural representation like the one in fig.2 to every string in the language. Fig.6 illustrates how this grammar defines the string-tree correspondence in fig.2.

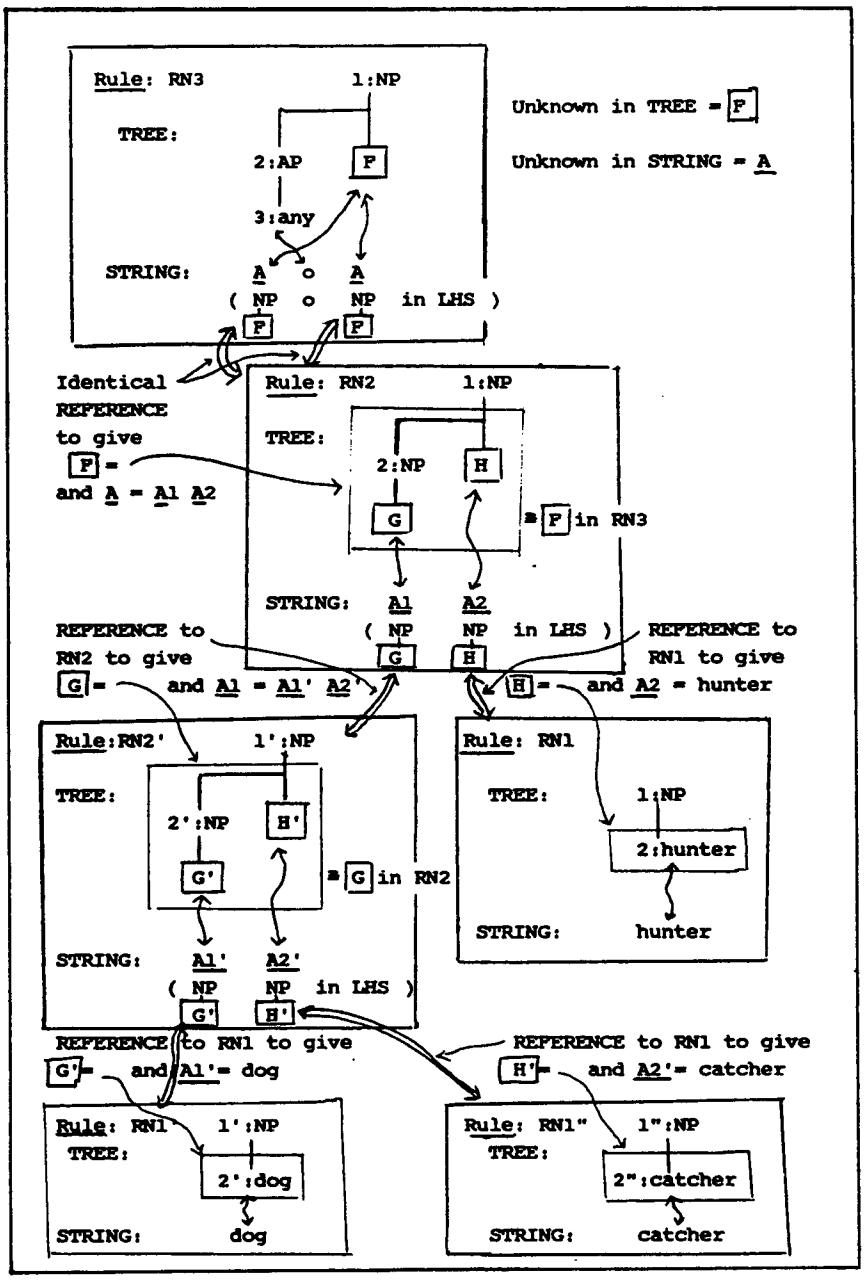
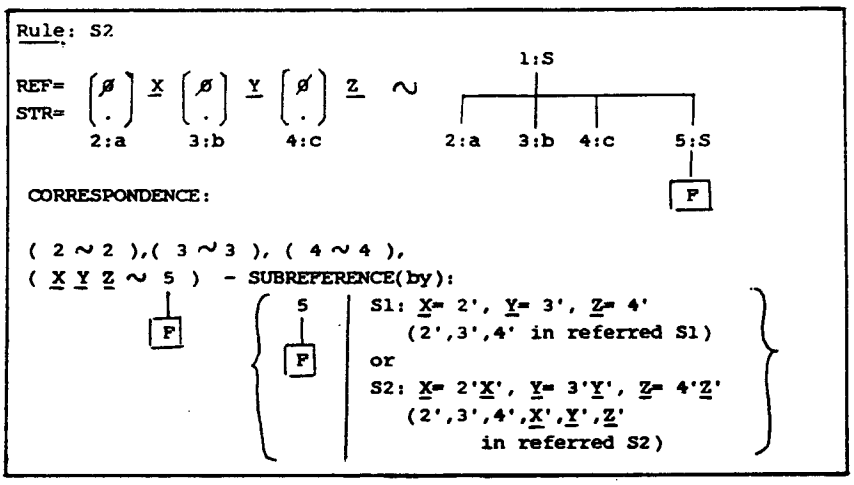


Fig.5 - Rules RN1,RN2,RN3 to define the correspondence in fig.1



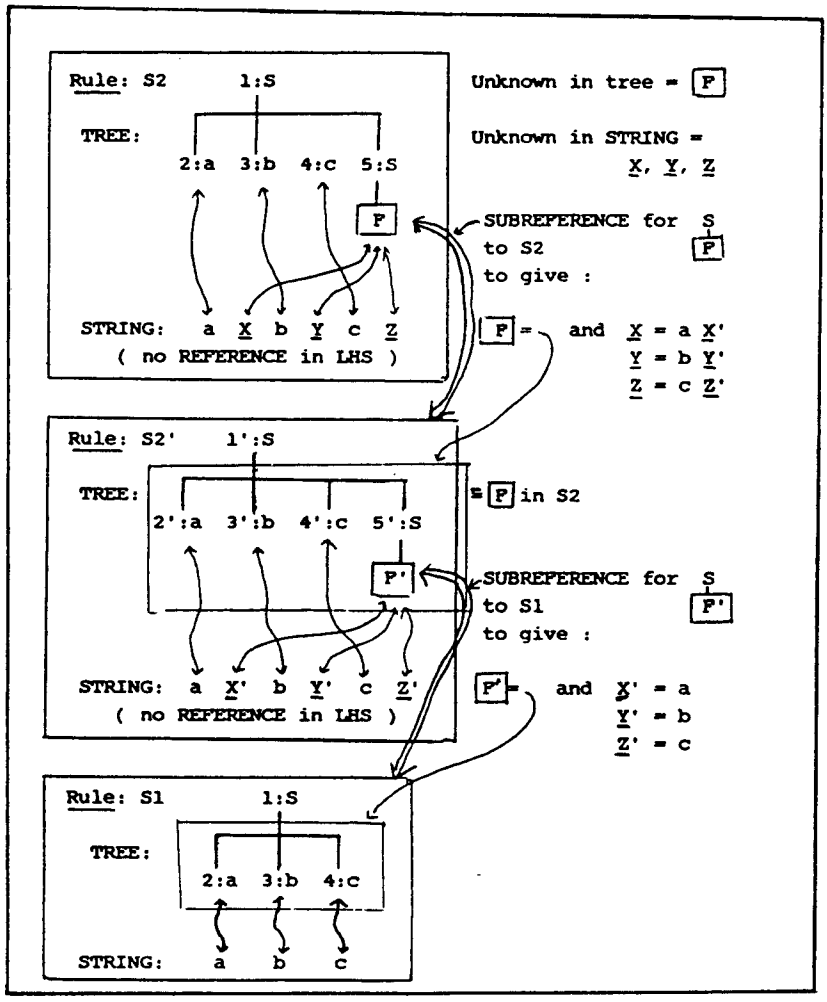


Fig.6 - Rules S1,S2 to define the correspondence in fig.2

The informal discussion in this paper gives the motivation and some idea of the formal definition of the String-Tree Correspondence Grammar. The grammar stresses not only the fact that one can express string-tree correspondence like the ones we have discussed, but also that it can be done in a 'natural' way using the formalism - meaning the structures and correspondence are explicit in the rule, and not implicit and dependent on the combination of grammar rules applied (as in derivation trees). The inclusion of the substring-subtree correspondence is also another characteristic of the grammar formalism. One also sees that the grammar is declarative in nature, and thus it is interpretable both for analysis and for generation (for example, by interpreting the rules as tree rewriting rules with variables).

In an effort to demonstrate the principal properties of the formalism, the STCG presented in this paper is in a simple form, ie. treating trees with each node having a single label. In its general form, the STCG deals with labels having considerable internal structure (lists of features, etc.). Furthermore, one can also express constraints on the features in the nodes - on individual nodes or between different nodes.
As mentioned, the concepts of direct derivation (=>) and derivation ($\overset{*}{\Rightarrow}$), as well as the derivation tree are also defined for the STCG. (Note that the rules with properties similar to the rule S2 entail a definition of direct derivation which is more complex than the classical definition). The set of rules in a grammar forms a formal grammar, ie. it defines a language, in fact two languages, one of strings and the other of trees.

At the moment, there is no large applications of the STCG, but as the STCG derives its formal properties from the Static Grammar, it would be quite a simple process to transfer applications in the Static Grammar into STCG applications. Like the Static Grammar, the STCG is basically a formalism for specification, but given its formal nature, one also aims for direct interpretability by a machine. Though still incomplete, work has begun to build such an interpreter [Zajac 86].

ACKNOWLEDGEMENTS

I would like to thank Christian Boitet who had been a great help in the formulation of the ideas presented here. My gratitude also to Hans Karlgren and Eva Hajičová for their remarks and criticisms on earlier versions of the paper.

REFERENCES

[Boitet-et-al-82]
Ch. Boitet, P. Guillaume, M. Quezel-Ambrunaz
"Implementation and conversational environment of ARIANE-78.4".
Proceedings of COLING-82, Prague.

[Boitet 86]
Ch. Boitet
"The French National Project : technical organization and translation results of CALLIOPE-AERO".
IBM Conference on machine translation, Copenhagen, August 1986.

[Chappuy 83]
S. Chappuy
"Formalisation de la description des niveaux d'interprétation des langues naturelles. Etude menée en vue de l'analyse et de la génération au moyen de transducteurs".
Thèse 3ème Cycle, Juillet 1983, INPG, Grenoble.

[Pullum 84]
G.K. Pullum
"Syntactic and semantic parsability".
Proceedings of COLING-84, Stanford.

[Tong 86]
Tong L.C.
"English-Malay translation system : a laboratory prototype".
Proceedings of COLING-86, Bonn.

[Vauquois 78]
B. Vauquois
"Description de la structure intermédiaire".
Communication présentée au Colloque de Luxembourg, Avril 1978, GETA doc., Grenoble.

[Vauquois & Chappuy 85]
B. Vauquois, S. Chappuy
"Static Grammars : a formalism for the description of linguistic models".
Proceedings of the conference on theoretical and methodological issues in machine translation of natural languages, COLGATE University, New York, August 1985.

[Zaharin 86]
Zaharin Y.
"Strategies and heuristics in the analysis of a natural language in machine translation".
PhD thesis, Universiti Sains Malaysia, Penang, March 1986. (Research conducted under the GETA-USM cooperation - GETA doc., Grenoble).

[Zaharin 87]
Zaharin Y.
"The linguistic approach at GETA : a synopsis".
GETA document, January 1987, Grenoble.

[Zajac 86]
R. Zajac
"SCSL : a linguistic specification language for MT".
Proceedings of COLING-86, Bonn.