[5]

Handling Crossed Dependencies with the STCG

Tang Enya Kong and Zaharin Yusoff
Computer-Aided Translation Unit
Pusat Pengajian Sains Komputer
Universiti Sains Malaysia
11800 PENANG
{enyakong,zarin@cs.usm.MY}

Abstract

The String-Tree Correspondence Grammar (STCG) is a declarative bidirectional grammar formalism for writing grammars to be used as specifications for linguistic programs (lingware) in machine translation systems. One of its strengths is the 'natural' way of describing various non-'standard' linguistic phenomena such as crossed serial dependencies. This paper presents an Earley-based parser for the STCG formalism.

1 Introduction

The String-Tree Correspondence Grammar (STCG) [Zaharin87] is a grammar formalism for defining:

- a set of strings
  (a language),
- a set of trees
  (valid representation/interpretation structures),
- the mapping between the two
  (to be interpreted for analysis & generation).

The formalism is argued to be a totally declarative grammar formalism that can associate, to strings in a language, arbitrary tree structures as desired by the grammar writer to be the linguistic representation structures of the strings. More importantly is the facility to specify the correspondence between the string and the associated tree in a very natural manner. These features are very much desired in grammar writing, in particular for the treatment of certain linguistic phenomena which are 'non-standard', e.g. crossed dependencies. Furthermore, a grammar written in this way naturally inherits the desired property of bi-directionality (in fact non-directionality) such that the same grammar can be interpreted for both analysis and generation.

In this paper, we investigate the properties of the STCG for interpretation towards analysis (as is understood within the context of Machine Translation (MT)). Other than using STCG grammars as specifications for the automatic generation of analysis programs in the Specialised Languages for Linguistic Programming (SLLPs) of MT systems (a study reported in [Zaharin&Tang 90] and [Zaharin&Tang 91]), the work centres around the specification of a general analyser/parser for the STCG. The proposed STCG analyser is capable of mimicking some very useful features in various context-free parsing techniques. One such feature is the use of charts in tabular parsing algorithms, as exemplified in Earley's Algorithm [Earley 70], which is very helpful in avoiding redundancies that may otherwise result in a combinatorial explosion. Another is the compact way of representing possible parse trees for ambiguous sentences, such as the one seen in the GLR parser [Tomita 87] (a study reported in [Tang 94]). Here, we shall concentrate only on defining a natural way for handling the kind of awkward phenomena mentioned above (namely crossed dependencies) while at the same time retaining much of the efficiency of standard context-free parsing algorithms.

2 The STCG Formalism

The String-Tree Correspondence Grammar is a declarative grammar formalism that can be used to describe the correspondence between strings of terms and trees. In particular, linguistic rules are written with utterances as the string of terms (henceforth STRING) and the corresponding representative linguistic structures as the tree (henceforth TREE). Figure 1 gives an indication of a full STCG rule. The structure of the TREE is totally specified by the linguist and is not constrained by any application of rules (as in the case for the parse tree in the classical context free grammar). In a rule, the main correspondence is first declared: in the example, the STRING #NP1.#VK.#NP2.up (with #NP1, #VK, and #NP2 being *string variables*, ie. variables which are instantiable to strings of terms) is set to correspond to the TREE with root node S (where $A, $B, and $C are *forest variables*, ie. variables that can be instantiated to lists of subtrees), and the rule name is kept in it's *reference* - the string defined after label "with :". The main-corr(espondence) is followed by a declaration of subcorrespondences (on the right hand side) between substrings of the STRING and subtrees of the TREE, each of which possibly having a list of references (rule names). For example, the sub-corr(espondence) between the substring concatenation #VK.up and the subtree rooted at the node VK refers to the rule RVK, the latter being another rule in the grammar. This *reference* is a mechanism by means of which the string and forest variables mentioned earlier are fully instantiated via an operation called

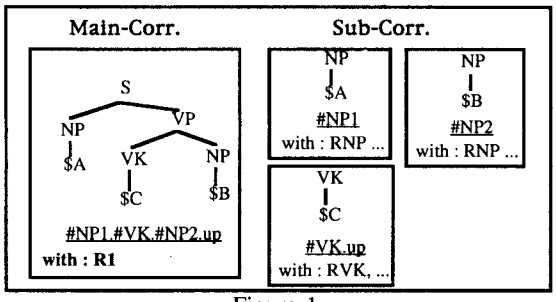


Figure 1.

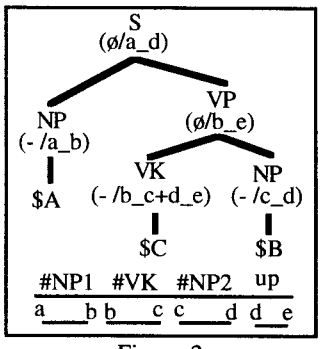


Figure 2.

identification ([Zaharin 86], [Lepage 89]) resulting in a correspondence between explicit strings of terms and trees, both without variables. In actual fact, the main-corr as well as the sub-corr specified in the rule are formally recorded in terms of a Structured String Tree Correspondence(SSTC) transparent to the linguist [Boitet&Zaharin 88] as illustrated in figure 2, where a given correspondence may be non-projective (eg. with discontinuous constituents) as is the case for the VK in the example.

Alternatively, rule R1 can be viewed as the specification of a general SSTC which is made up of a set of finer and explicit SSTCs, this being the case after all the sub-correspondences (or sub-SSTCs) defined by the referred rules are fully instantiated. We illustrate this specification in figure 3.

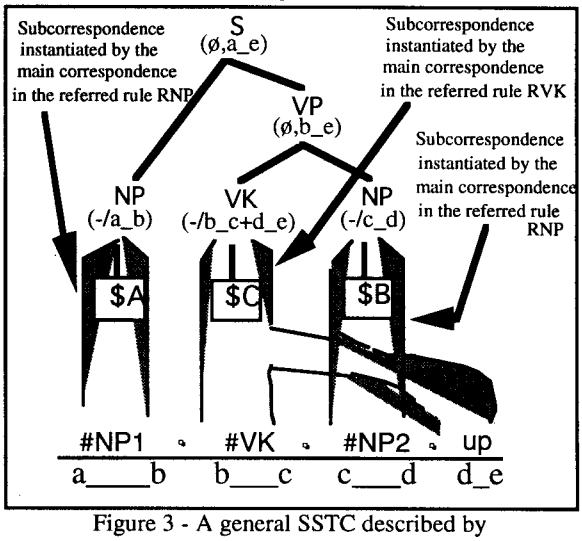


Figure 3 - A general SSTC described by rule R1 in figure 1.

In very simple terms, a string to tree correspondence in the STCG can be viewed as analogous to the mathematical definition of a relation between real numbers as in the example given on the right. Here, a relation (in this case a function) f is defined in terms of finer subrelations according to the subdomains.

$$f(x) = \begin{cases} x^2 - 3 & x \leq 3 \\ x^3 + 5 & 3 < x < 5 \\ x & 5 \leq X \end{cases}$$

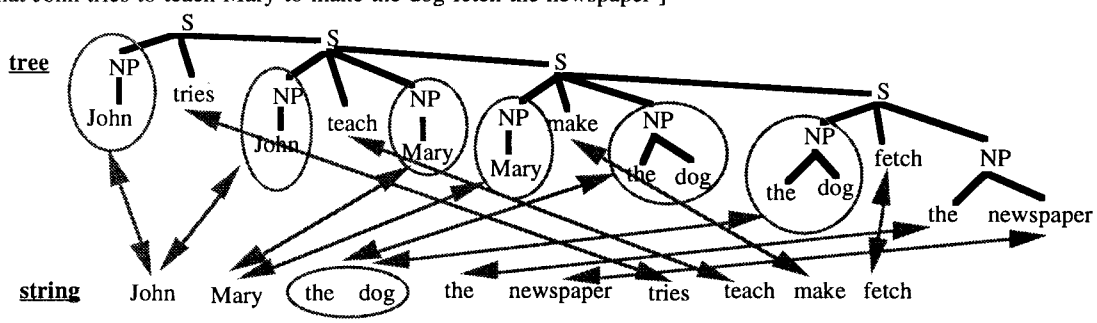
A set of STCG rules form a grammar, some of which are axiom rules (ie. start rules or rules containing axiom trees, as in the axiom or the start symbol S in the classical context free grammar). With the semantics of the rules being as indicated above, a grammar thus defines a language of strings, a language of representation trees, and the correspondence between elements of the two languages/sets.

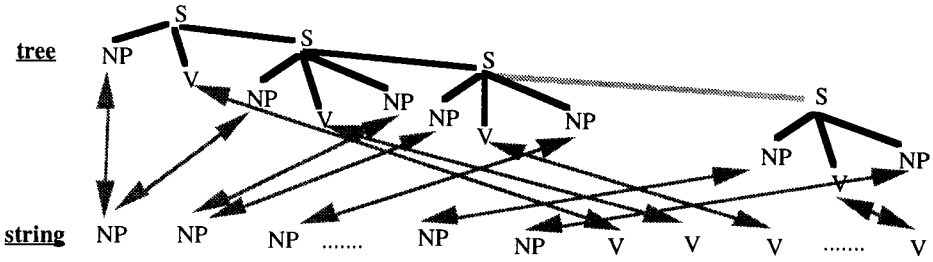
It is this set of string-tree correspondences that can be interpreted for both analysis and generation.

3 Crossed Dependencies using STCG

Crossed dependencies structures is quite a common phenomenon in certain natural languages. A very acute case can be seen in the Dutch relative clause below:

... dat Jan Marie de hond de krant probeert te leren laten halen
... that John Mary the dog the newspaper tries to teach let fetch
[= ... that John tries to teach Mary to make the dog fetch the newspaper]





An appropriate representation of the phrase structure for this string (ignoring the relative *dat*) would be the one given in the figure at the bottom of the last page (a simplified form of a structure based on [Seuren 89]). Note that in this chosen representation, the subcorrespondence between substrings in the string and subtrees in the tree are repeated in the manner given above.

The dependencies between verbs and NPs exhibit a crossed serial pattern : verbs further to the right in the string of verbs take, as their subjects and objects, NPs further to the right in the string of NPs in an overlapping manner.

In most formalisms, the problem is usually solved by means of transformations, and in others they are circumnavigated or simply ignored (especially if tree structures are not used). In augmented context-free grammars, such a situation is normally handled by the introduction of a recursive feature *stack* [1] (similar to the use of indexes in the Indexed Grammar) to ensure that the repetitive dependencies of discontinuous constituents are appropriately established in the string. However, a closer look at that technique may reveal that in some cases the specified correspondence may not satisfy the desirable property of bidirectionality due to the use of irreversible operations. Furthermore, this process may also have to resort to certain linguistically unmotivated intermediate structures.

In STCG, apart from the usual description for noun phrases, only three other descriptions are required, namely figure 4 for the description of S with a main verb taking a single noun phrase and a subordinate sentence as its arguments, figure 5 for the description of S with a main verb taking two noun phrases and a subordinate sentence as its arguments, and figure 6 for the basic description of S.

Note that the dependencies between verbs and NPs exhibit a crossed serial pattern : verbs further to the right in the string of verbs take as their subjects and objects NPs further to the right in the string of NPs, i.e. in an overlapping manner. In the STCG, we have the possibility of describing crossed dependencies by picking out disconnected substrings

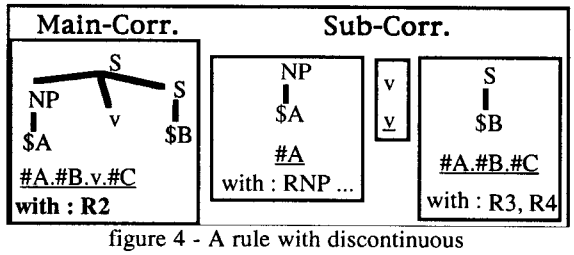


figure 4 - A rule with discontinuous 2-arguments (an NP and an S)

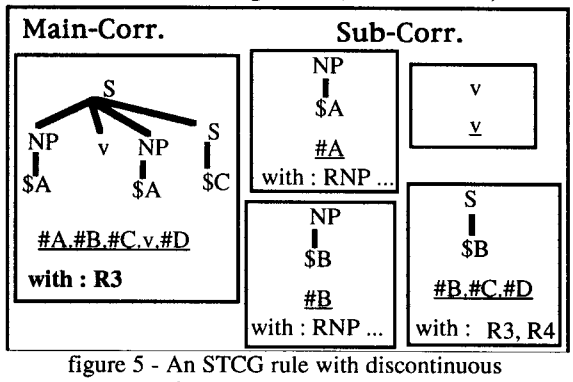


figure 5 - An STCG rule with discontinuous 3-arguments (two NPs and an S)

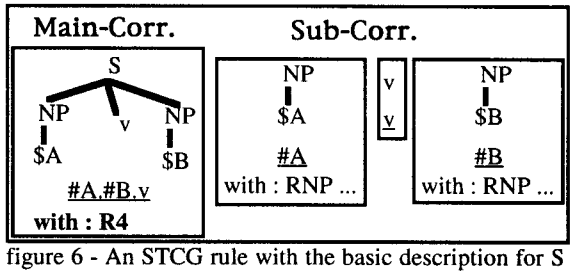


figure 6 - An STCG rule with the basic description for S

in the STRING of the main correspondence to form connected substrings in the STRING of the subcorrespondences. This will result in subcorrespondences between substructures in the TREE and disjoint substrings in the STRING.

4 Analysis of Crossed Dependencies Based on the STCG

In the following, we show that the Dutch relative clauses example given above can be treated with an Earley based algorithm for the STCG. We look at the analysis of the transliterated Dutch relative clause "John Mary the dog the newspaper tries teach let fetch." with respect to a grammar which consists

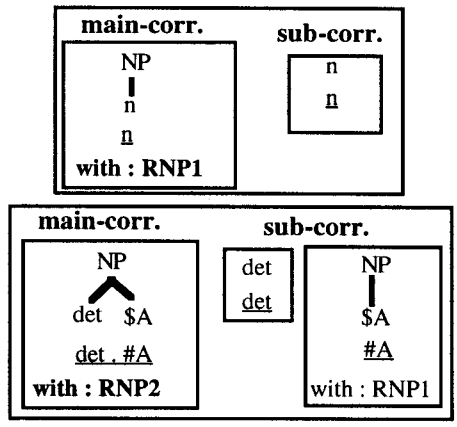
[1] The introduction of a recursive feature stack may invoke a situation where there are infinitely many categories - a situation which induces non-termination in parsing with feature-theoretic grammar formalisms like PATR-II. (see [Shieber 89] and [Harrison&Ellison 92]).

of rules R2,R3,R4 as given above and two NP rules on the right, namely RNP1 and RNP2, with R2 being the axiom rule.

As we have seen above, a STCG grammar actually defines a set of SSTCs in a way quite similar to the definition of a mathematical function. In evaluating a mathematical function, if the function is defined in terms of other sub-functions then it can only be completely evaluated after all its sub-functions have been evaluated and return with the appropriate values. We can view the STCG analysis process in the same manner where, by taking the input string/sentence as their STRING, the set of explicit SSTCs defined by the axiom rules of a grammar are constructed based on the resultant sub-SSTCs defined by the reference rules of these axiom rules. Since the reference rules of the axiom rules may in turn refer to other rules, they may also return the completed SSTCs only after their respective reference rules have been completed. This reference process will terminate when all remaining sub-SSTCs evaluated are defined by subcorrespondences which do not refer to any other rule, namely the 'lexical-SSTCs', which must match with the input words (the non-lexical SSTCs are called 'phrasal-SSTCs'). An example analysis of the Dutch relative clause based on the given set of rules is illustrated by the SSTC derivation tree in figure 7.

As in Earley's algorithm, in general, the STCG analyser scans the input string $X_1,...,X_n$ from left to right, and as each word X_i is scanned, a set of items for state S_i is constructed with each item representing the conditions on the analysis process at that point in the scan. The operations on a state S_i are as follows : process each item in the state in order, performing one of the three operations of prediction, scanning and completion, depending on the form of the item. These operations may add more items to S_i and/or put items in a new state S_{i+1}. Each item in a state consists of the following :

(1) a STCG production (with all its STRING assigned with interval variables), containing a marked position in the STRING of the main-corr (henceforth main-STRING) to show that the portion of the input string scanned till now is identifiable with the substring to the left of the marker and so is derivable from the corresponding component sub-SSTC,

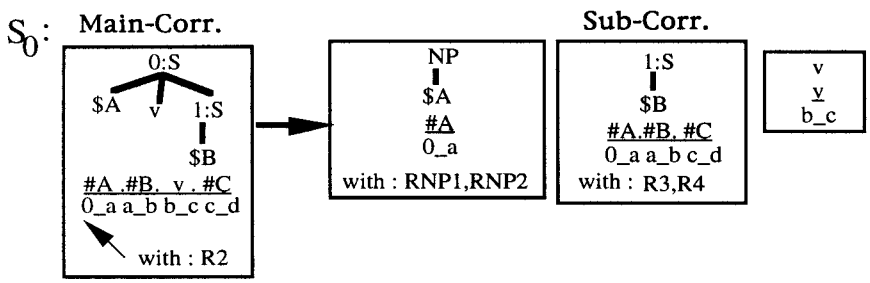


(2) a point in the said production functioning as the marker described,
(3) a list of state pointers pointing back to the positions in the input string (or the states) at which point the item was updated to begin looking for the next sub-SSTC.

This 3-tuple is represented as a production with a dot in the main-STRING, followed by a list of integers functioning as the list of state pointers. Below is an example item corresponding to rule R2 generated at state S_0 since none of the input words have been recognised at that point in the scan (we have dropped the NP node to save space)

The prediction operation is applicable to an item when there is a phrasal SSTC in the sub-corr(espondence) containing a STRING with the beginning index of its interval equal to the dot position - which corresponds to the next sub-SSTG that needs to be constructed. The example item given above is applicable for the prediction of two phrasal-SSTCs in the sub-corr as their beginning indices are similar to the dot position.

The scanning operation is applicable in cases when there is a lexical SSTC in the sub-corr(espondence) containing a STRING with the beginning index equal to the dot position. The operation tries to match the lexical-SSTC of the item with the next word X_i in the input string, and if they match, it adds the item to the next state S_{i+1}, with the dot moved over the terminal symbol in the main-STRING of the item to indicate that the terminal symbol has been scanned. Observe that when an index interval variable in an item is intantiated to a



(0)

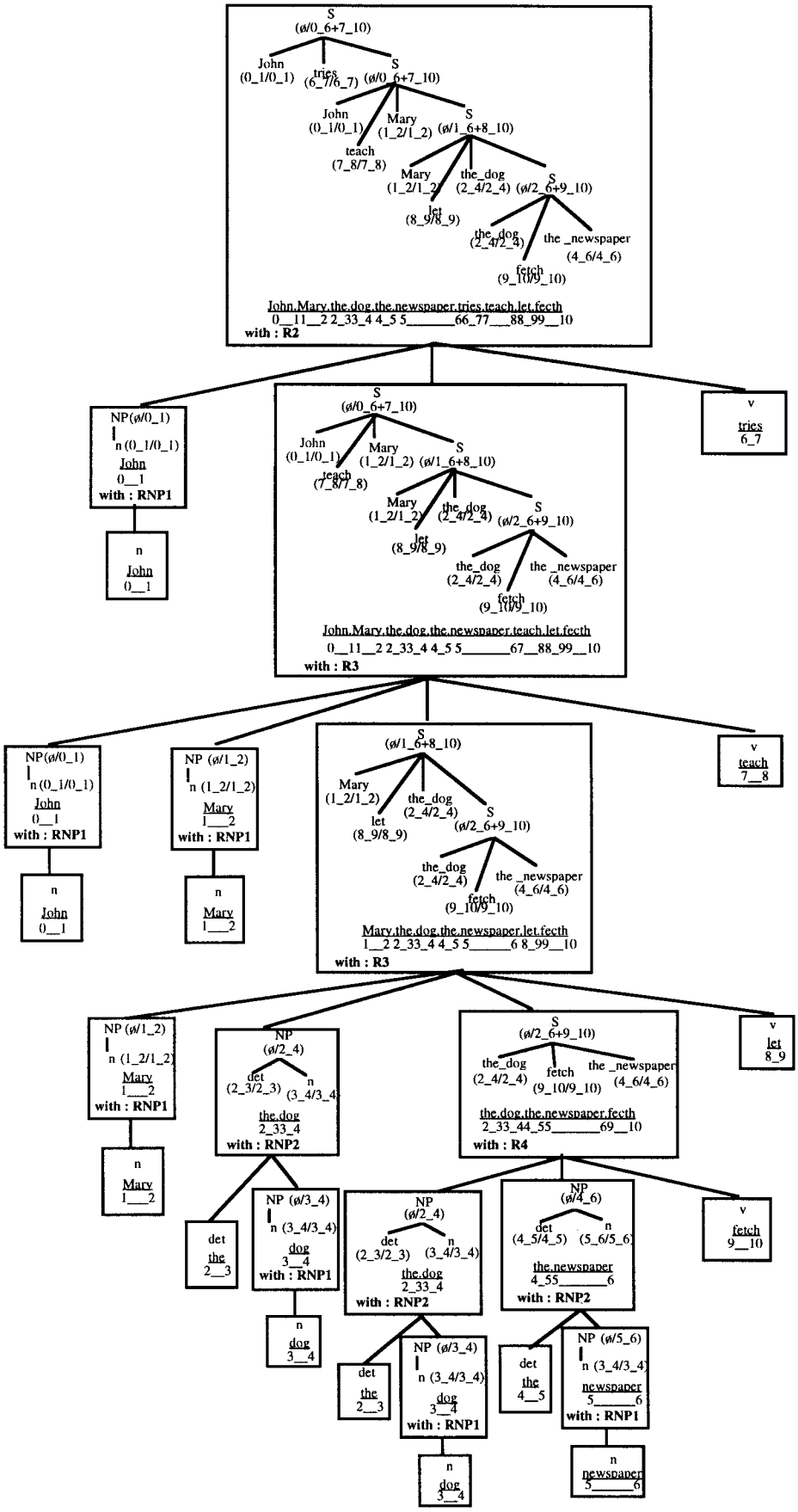


Figure 7 : An example SSTC derivation tree for the analysis of the Dutch relative clause.

constant index during scanning, indicating that the interval i_i+1 covers only a single word of the input string, namely X_i, all items in the states (except the current state) pointed to by the list of state pointers and containing the interval d_*a* (d is the dot position and *a* is the index variable instantiated to i+1) must also be updated accordingly. As can be seen clearly from the derivation tree in Figure 7, this updating process provides a natural way of treating crossed dependencies in discontinuous constituents.

The third operation, completion, is applicable to an item if the dot has moved to the end index of the main-STRING and that all its sub-corrs have been successfully combined with the resultant phrasal-SSTCs of the referred rules or the input words. The completion operation first identifies (as defined in [Lepage&Zaharin 91]) the string of TREE defined in all sub-SSTC of the item with the corresponding string of TREEs consisting of the TREEs of the combined SSTCs or the input words. If the identification is successful then it produces a complete SSTC as defined by the main-corr of the item and the resultant SSTC is then combined to those item which are waiting for the completion of this item in order to proceed further. We refer the reader to [Tang94] for further details on the parser.

5 Concluding Remarks

Efficient context-free parsing methods such as the LR parser and Earley's Algorithm have recently been referred to extensively in implementing parsers for many formalisms used in the field of NLP. In an effort to retain the efficiency of standard context-free parsing algorithms, most recent declarative formalisms are typically restricted by the constraint of string concatenation in context-free grammars which allows a sentence to be systematically decomposed so that the parsing process can be indexed by the subparts of that decomposition (the substrings). However, it has also been widely recognised that the concatenation restriction of CFG can be problematic in handling phenomena like crossed dependencies. As an alternative, we propose the STCG formalism which allows for a more 'natural' way of specifiying the strings of the language being described, their corresponding linguistically motivated representation trees, and the correspondence between the two, where the correspondence need not be projective and hence appropriate for the said phenomena. Even though the standard CF parsing methods cannot be adopted directly in the analysis of an input sentence with respect to a STCG grammar (due to the STRING patterns of the STCG which need not submit to the concatenation restriction of CFG), in this paper we have been able to present an analyser for the STCG which is capable of mimicking some very useful features in various context-free parsing techniques while at the same time handling the kind of awkward phenomena, namely crossed dependencies.

References

[Boitet & Zaharin 88] Zaharin Yusoff, Christian Boitet, *Representation trees and string-tree correspondences*, proceedings of the 12th International Conference on Computational Linguistics, COLING-88, Budapest, August 1988, pp.59-64.

[Earley 70] J. Earley, *An efficient cantext-free parsing algorithm* , Communications of the ACM, Vol. 13, Num. 2, Feb 1970, pp. 94-102.

[Harrison&Ellison 92] S. P. Harrison, T.M. Ellison, *Restriction and Termination in Parsing with Feature-Theoretic Grammars*, Computaional Linguistics, Vol. 18, Num. 4, 1992.

[Lepage89] Y.Lepage, *Un systeme de grammaires correspondancielles d'identification,* these de Docteur, IMAG, Universite Joseph Fourier, Grenoble, June 1989.

[Lepage&Zaharin 91] Zaharin Y., Yves Lepage, *Identification: a unification-like operation with variables instantiating to forests,* proceedings of the International Conference on Current Issues in Computational Linguistics, Penang, June 1991.

[Seuren 89] P. Seuren, *A problem in English subject complementation* , In : D. Jaspers, W. G. Klooster, Y. Putseys, P.A.M. Seuren (eds.), *Sentential complementation and the lexicon. Studies in honour of Wim de Geest* , Foris, Dordrecht, Holland, 1989, pp.355-375.

[Shieber 89] S.M.Shieber, *Parsing and Type Inference for Natural and Computer Languages.* Ph.D. thesis, Stanford University, Stanford, California.

[Tang 94] Tang Enya Kong, *Natural languages Analysis in machine translation (MT) based on the STCG*, PhD thesis, Universiti Sains Malaysia, Penang, March 1994 (to be submitted).

[Tomita 87] Masaru Tomita, *An Efficient Augmented Context-Free Parsing Algorithm,* Computational Linguistics, 13(1-2) January-June 1987.

[Zaharin 86] Zaharin Y., *Strategies and heuristics in the analysis of natural languages in machine translation*, PhD thesis, Universiti Sains Malaysia, Penang, March 1986.

[Zaharin 87] Zaharin Y., *String-Tree Correspondence Grammar: a declarative grammar formalism for defining the correspondence between strings of terms and tree structures*, proceedings of the 3rd Conference of the European Chapter of the Association for Computational Linguistics, Copenhagen, April 1987.

[Zaharin&Tang 90] Zaharin Yusoff, Tang Enya Kong, *Generation of analysis programs in ROBRA (ARIANE) From String-Tree Correspondence Grammars (or a Strategy for Analysis in machine tranaslation),* PTMK document, Penang, 1990.

[Zaharin&Tang 91] Zaharin Yusoff, Tang Enya Kong, *String-Tree Correspondence Grammars as a base for the automatic generation of analysis programs in machine tranaslation,* proceedings of the International Conference on Current Issues in Computational Linguistics, Penang, June 1991.