GENERATION OF SYNTHESIS PROGRAMS IN *ROBRA (ARIANE)* FROM STRING-TREE CORRESPONDENCE GRAMMARS (or a Strategy for Synthesis in Machine Translation)

Zaharin Yusoff
Projek Terjemahan Melalui Komputer
PPS Matematik & S Komputer
Universiti Sains Malaysia
11800 PENANG

Introduction

Specialised Languages for Linguistic Programming, or SLLPs (like ROBRA, Q-systems, Augmented Transition Networks, etc.), in Machine Translation (MT) systems may be considerably efficient in terms of processing power, but its procedural nature makes it quite difficult for linguists to describe natural languages in a declarative and natural way. Furthermore, the effort can be quite wasteful in the sense that different grammars will have to be written for analysis and for generation, as well as for different MT systems. On the other hand, purely linguistic formalisms (like those for Government and Binding, Lexical Functional Grammars, General Phrase Structure Grammars, etc.) may prove to be adequate for natural language description, but it is not quite clear how they can be adapted for the purposes of MT in a natural way. Besides, MT-specific problems, like appositions, ambiguities, etc., have yet to find their place in linguistics.

Nevertheless, linguistics has its role in MT, and thus some formalism will have to be found that is friendly to linguists and yet be general enough to support data structures for problems which are not terribly 'interesting' to linguists but are essential to MT. Such a formalism must not only be adequate for language description, but must also serve as a specification language for MT programs written in SLLPS.

A formalism designed specifically for this purpose,is the Static Grammar (SG) [Vauquois&Chappuy85], which was further refined into the String-Tree Correspondence Grammar (STCG) [Zaharin87a]. As in most grammar formalisms, it is very difficult to argue that the STCG is adequate for language description, but its declarative nature does provide the possibility of writing a single grammar that can be interpreted for both analysis and generation. The formalism also supports data structures for the 'non-linguistic' MT problems, and it is general enough to 'express' different linguistic theories, or a combination of them. In short, the STCG can serve as a specification language for applications in various MT systems, as is illustrated in figure 1.

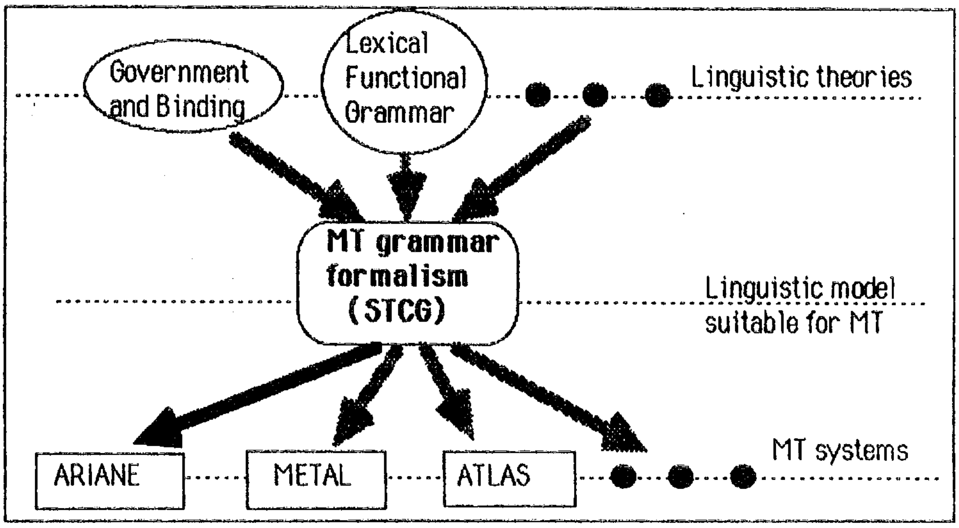
Government and Binding
Lexical Functional Grammar
Linguistic theories
MT grammar formalism (STCG)
Linguistic model suitable for MT
ARIANE
METAL
ATLAS
MT systems

Figure 1 - The STCG as a specification language for MT.

This paper reports on the work done in building a generator of ROBRA programs for synthesis in MT from grammars written in STCG (as indicated in bold in figure 1). Such an effort necessitates a proposal for a general strategy for synthesis, which we shall discuss now.

Synthesis

As discussed in [Zaharin89], synthesis in MT is not the same as in generation of natural languages. Generation is the process of generating all grammatical sentences in a language from a given axiom (here, an axiom may be structurally more complex than a single symbol, depending on the linguistic model and grammar formalism adopted), whereas synthesis is the process of producing a grammatical sentence from some input, which, in the case of translation, is the structure obtained from the analysis of the source text (in the case of an interlingual approach) or some structure derived from it (in the case of a transfer approach). This input to synthesis may vary from system to system, application to application, and strategy to strategy, but it is certainly not equivalent to an axiom as understood in natural language generation. Perhaps, one may view synthesis as part of a path (or a subset of paths) in the set of all possible paths in generation, where the decision on the input structure determines the point of entry along the said path (or paths). Figure 2 gives an illustration.

The aim of synthesis in MT is two-fold: one is to generate a sentence in the target language which has the same 'meaning' as the source sentence analysed, and the other is to ensure that this sentence is grammatical with respect to the target language. In most, if not all current MT systems, the first objective may have been achieved, but there is little *guarantee* that the target sentence is grammatical. Naturally guaranteeing such a result must be with respect to some grammar for the language, which is presumably written to be interpretable at least for generation (whether this grammar adequately defines the natural language in question is beside the point). However, as synthesis is quite different from generation, this grammar cannot be written as a SLLP program but is used only as a guide to write it.

Failures in synthesis (at least in the case of grammaticality) can be attributed to missing all valid paths in generation. As it is difficult to test whether one is currently on a valid path, one way of providing this guarantee is to make certain that the synthesis process does pass through the axiom point of generation. Naturally it is too much to ask of the transfer phase to output an axiom, because then the transfer phase would include some of the monolingual processes, which altogether defeats the purpose of the transfer architecture. However, it is possible to arrange it in such a way that the first part of synthesis (Synthesis1) builds an axiom from the output of transfer, and then the target sentence is generated from this axiom (Synthesis2). This way, Synthesis2 is effectively a restriction of generation, which is thus obtainable directly from the grammar specified, hence guaranteeing that the target sentence is grammatical. Synthesis1 can be based on a comparative study between the output structures of transfer and the axioms of the target language.

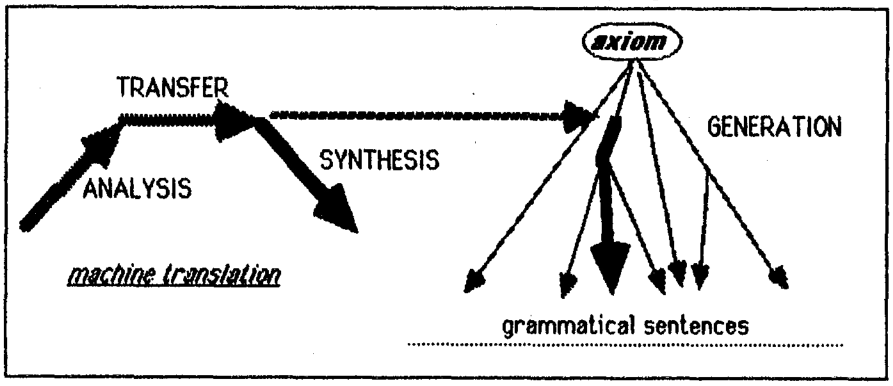

Figure 2 - Synthesis as opposed to generation.

The proposed strategy has an added advantage, which is the possibility of building multilingual systems based on the transfer approach, namely because the axiom point is exactly the output of analysis had the target text been used for analysis (as suggested in [Boitet88]). This is certainly a surer way of building multilingual systems while waiting for a genuine interlingua to be designed. Such an architecture is illustrated in figure 3.

Design

At GETA in Grenoble as well as in our project, multilevel structures or m-structures [Zaharin87b] are used as representation structures for sentences, hence the axioms. The m-structures contain four levels of interpretation, corresponding to morpho-syntactic decomposition, syntactic functions, logical relations and semantic relations. These actually give rise to four different structures, but are combined into a single structure (in the manner illustrated in figure 4), to show the inter-relation between the various levels of representation as well as to facilitate processing.

The set of all valid m-structures in a language, as well the respective mappings or correspondences [Boitet&Zaharin88] between these structures and the sentences they represent, are described by means of the formalism of the STCG. The result is a grammar for the language which can be interpreted for both analysis and generation.

For the purposes of MT, the logical and semantic levels of interpretation are considered as (almost) universal to all languages, while the other two are language dependent. Thus these two levels are used as the pivot for translation. However, certain surface features (pertaining to the morpho-syntactic decomposition and syntactic functions, as well as some other features analysable from the source text) are also transferred as an aid to translation. More precisely, they are considered as heuristics which can guide the synthesis, in particular to find the necessary path in generation. [They are also used in fail-soft measures in case the analysis fails, but that is beside the point.]

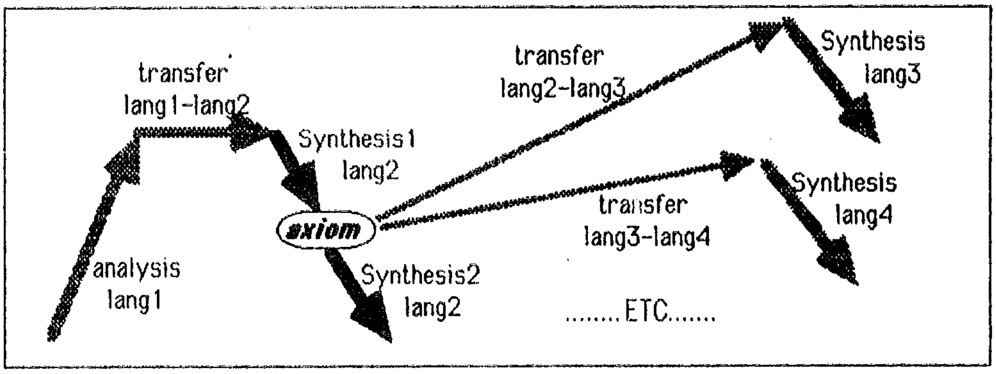

Figure 3 - A multilingual system based on the transfer approach.

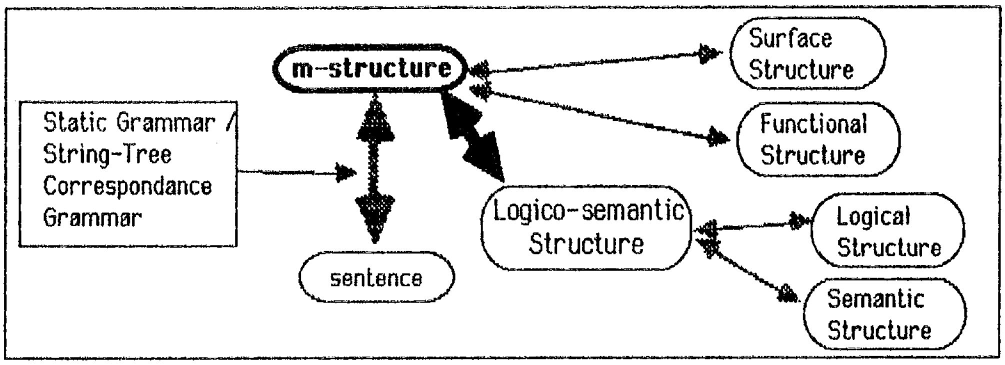

Figure 4 - The linguistic model.

With the above in mind, the input to synthesis is the combination of the logical and semantic structure, namely the logico-semantic structure, augmented with certain transferred surface features (hence expressed in terms of the target language) to be used as heuristics. Synthesis1 is then the process of building an axiom in the target language from this input, which is basically retracing the bold path in figure 4, where the process is aided by the said heuristics (which otherwise would give a large number of possibilities). Synthesis2 is the process of generating the target sentence from the resulting axiom, hence exactly the opposite of analysis. Both Synthesis1 and Synthesis2 will have to be interpreted directly from the grammar rules, which, coupled with the fact that the process passes through the axiom, ensure the grammaticality of the generated sentence.

Implementation

It would not be possible to describe the implementation in full in a short paper. Furthermore, one would need to be familiar with the STCG formalism [Zaharin87a][Zaharin90] as well as the ROBRA language in ARIANE [Boitet79]. However, we shall give a brief outline here to indicate the general strategy.

STCG is a set of rules defining the correspondence between a text and its chosen representation structure (in our case a m-structure). Simplified to the utmost (with tree structures as well as complex feature lists being eliminated, which incidentally decreases its capability of treating discontinuities in a natural way), its rules resemble context free rules with references which function to restrict the possible references to other rules (a form of subscripts). Figure 5 shows an example of a set of context free rules being rewritten in this simplified form of STCG rules.

On the other hand, ROBRA contains a set of tree transformational rules whose application is dictated by a control graph, where each node contains a set of rules to be considered for application and each arc has conditions on the output of the last node.

In our implementation, STCG rules are translated to ROBRA rules (possibly a few ROBRA rules to one STCG rule) while the references are interpreted to provide the control graph. In Synthesis1, only the tree part (RHS) is considered, where the ordered tree structure given in the STCG rule together with its features pertaining to the logical and semantic interpretation are used as conditions for testing and then ordering the nodes and subtrees in the object tree. The assignment of the rest of the features (also in the tree part) is based on the heuristics obtained from the transfer phase, which actually form part of the rest of the features in the STCG rule, but are in this case used as conditions of assignment. The output of Synthesis1 is an axiom in the target language. Synthesis2 is exactly the opposite of analysis, where the tree part is used as conditions and the string part as assignments. Figure 6 indicates the computation from the various parts of a STCG rule to the various parts of a ROBRA rule, in this case for Synthesis2.

context free rules	*simplified STCG rules*
r1: S → NP(all) VP	r1: S → NP VP ref(all) ref(all)
r2: NP1 → n	r2: NP → n
r3: NP2 → det NP1	r3: NP → det NP ref(r1)
r4: NP3 → adj NP1/ adj NP2	r4: NP → adj NP ref(r1,r2)
r5: VP → v NP(all)	r5: VP → v NP ref(all)

Figure 5 - An example showing the function of references in STCG rules.

As for the control graph in ROBRA, Synthesis1 and Synthesis2 have similar control graphs, which we indicate in figure 7. The main transformational system is standard except for the part indicated, which together with the subtransformational systems (one for each syntactic class K) are computed from the references in the rules.

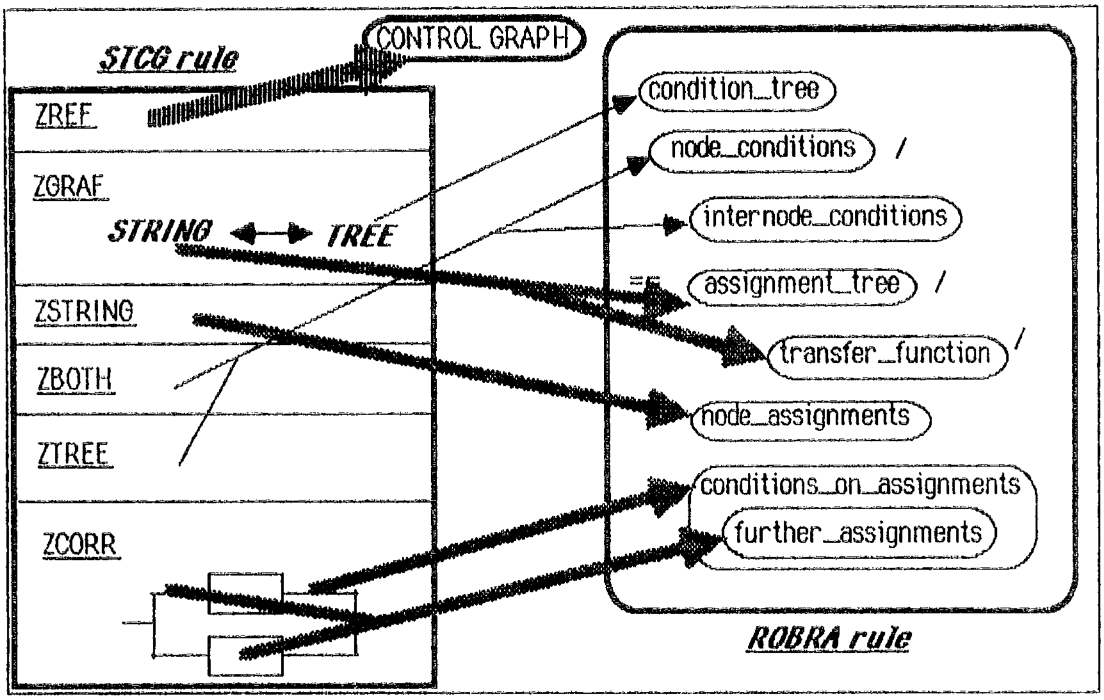

Figure 6 - The computation from STCG rules to ROBRA rules for Synthesis2.

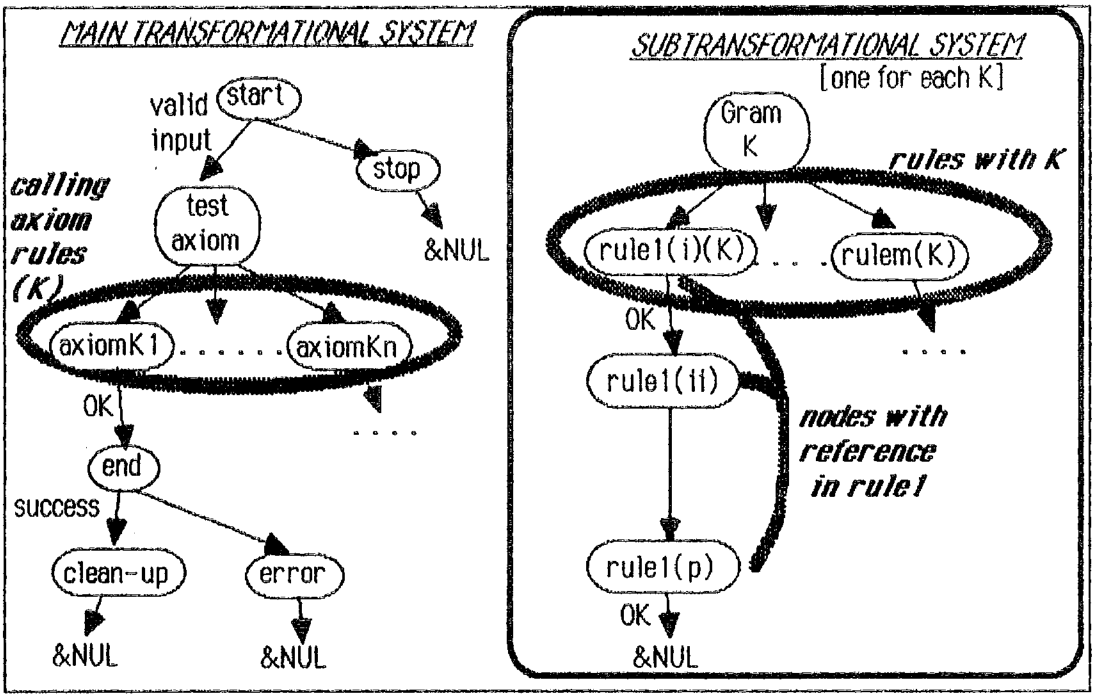

Figure 7 - The control graph generated.

Current Situation and Future Work

For the moment, only the generator for Synthesis2 has been implemented. An editor for the STCG as well as for the SG have been developed (both with syntax checkers), which generate the same internal form, from which the generator of ROBRA synthesis programs (currently only Synthesis2) produces its output. Both the editors as well as the generator are implemented on the Macintosh using TURBO PASCAL V. The output ROBRA program is a text file which can be transferred to the IBM mainframe (on which runs ARIANE) and then compiled under the ARIANE environment.

The next phase in this work will be the implementation of a generator for Synthesis1, followed by a generator for Analysis, and perhaps Transfer. The ultimate aim is to provide an environment in which a MT application can be built by means of specifying only linguistic rules in a declarative and natural way, in particular without having to write SLLP programs. However, it is still not clear how ambiguity and transfer rules can be incorporated automatically.

REFERENCES

[Boitet79] Ch.Boitet, *Automatic production of CF and CS-analyzers using a general tree transducer*, 2^ Int. Kolloquium Uber Mashinnelle Ubersetzung, Lexicographie und Analyze, Saarbrucken, November 1979.

[Boitet88] Ch.Boitet, *Hybrid pivots using m-structures for multilingual transfer-based MT systems*, Meeting of the Japanese Institute of Electronics, Information and Communication Engineers, Tokyo, June 1988.

[Boitet&Zaharin88] Ch.Boitet, Zaharin Y., *Representation trees and string-tree correspondences*, proceedings of the 12th International Conference on Computational Linguistics, COLING-88, Budapest, August 1988, pp.59-64.

[Vauquois&Chappuy85] 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.

[Zaharin87a] 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, pp.160-166.

[Zaharin87b] Zaharin Y., *The linguistic approach at GETA*, TECHNOLOGOS (langues et artefacts), printemps 1987, **no.4**, LISH-CNRS, Paris, pp.93-110.

[Zaharin89] Zaharin Y., *On formalisms and analysis, generation and synthesis in machine translatiom*, proceedings of the 4th Conference of the European Chapter of the Association for Computational Linguistics, Manchester, April 1989, pp.319-326.

[Zaharin90] Zaharin Y., *Structured string-tree correspondences and the String-Tree Correspondence Grammar*, Projek Terjemahan Melalui Komputer, Universiti Sains Malaysia, Penang, January 1990.