1 A Lexicalized Tree Adjoining Grammar for the Generation of Chinese Weatherforecasts

Yu-Fang Wang, email: fangfang@coli.uni-sb.de
Dept. of Computational Linguistics, University of Saarbrücken

Abstract

This paper presents a first Tree Adjoining Grammar for a sublanguage of Mandarin Chinese. It models the style of the weather forecast bulletins currently adopted by the Shanghai Meteorological Center (SMC), at the same time exploring extensions to a broader coverage of the language. It is implemented under the incremental generator *TAG-GEN* which uses a variation of the *Tree Adjoining Grammar (TAG)* formalism. The grammar will be used in the system *Multilingual Weather Forecast Assistant (MLWFA)* and is intended to serve as a basis for further applications of the *multilingual generation project ACNLG*[1].

Keywords: Natural Language Generation, multilingual generation, Tree Adjoining Grammar, surface generation, computational grammar

1.1 Introduction

Natural Language Generation (NLG) is the field of research that investigates how computer programs can be built to produce natural language text from data provided by an application program.

The overall goal of the ACNLG project[2] [Huang et al. 96a, Huang et al. 96b] is to develop an architecture for multilingual generation that is able to handle structurally different languages such as Chinese, English and German. The approach will be tested in one or two real-world applications.The MLWFA system[3] will be ACNLG's first application. It is intended to generate weather forecasts from meteorological data in an automatic way.

ACNLG uses a widely accepted three-stage model for generation that comprises of the system components macroplanner, microplanner, and surface generator. It is realized as a pipeline architecture (s. fig. 1).

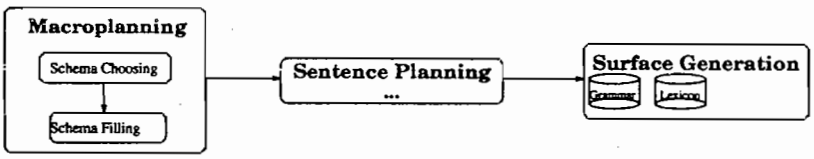

Figure 1: The architecture of ACNLG

- **Macroplanning** – ***What-to-Say***: The macroplanner (or *deep generator*) determines the content of the text by selecting and organizing the material to be included.
- **Microplanning** – ***How-to-Say***: The microplanner (or *sentence planner*) chooses appropriate *linguistic resources* (i.e. syntactic categories such as noun phrase or sentence, and words) for the pieces of information chosen, and arranges them into paragraphs and sentences.
- **Surface Generation** – ***Realization***: The surface generator realizes syntactic operations and produces a grammatical natural language utterance.

[1]The project is supported by Volkswagen Stiftung, the Shanghai Commission for Science and Technology, and the Chinese National Science Foundation.

[2]ACNLG is a cooperation between the German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany, and Shanghai Jiaotong University (SJTU)

[3]MLWFA is a joint effort between SJTU/DFKI and the meteorologcical center of Shanghai. The SMC is responsible for providing meteorological data, while the generation of natural language sentences out of these data is the task of SJTU/DFKI.

For the surface generation part, a computational grammar was designed which is implemented in the TAG formalism [Wang 97]. TAG was used because, first, it is a well-established formalism, and, second, because it provides appropriately sized chunks for the generation from the microplanner output as used in MLWFA (see section 1.4).

1.2 The Weather Forecast Language

The SMC weather forecast language is, as in general, characterized by a relatively small vocabulary and telegraphic style. The language is an interesting subset of Chinese in that it covers remarkably many phenomena. However, it contains no passives, no negation and hardly makes use of copula. Elliptic constructions are often found. Another significant feature of the weather forecast language is the extensive use of complex nominal constructions. The following phenomena were investigated and implemented:

- a set of Chinese sentence structures that are common in Chinese and are also represented in the SMC sublanguage. Apart from ordinary subject-predicate sentences that also exist in English and German, double subject sentences and NP_{SUBJ}– NP_{PRED} were considered.
- the Chinese noun phrase having the classifier phrase (ClP) as specifier, and a number of modifiers: relative clauses, measure phrases, associative phrases and attributive adjectives. Rarely do they occur in a sequence within one noun phrase.
- coordinate constructions on sentence, noun and number level,
- subordinate constructions (i.e., subclauses modifying main clauses), and
- serial verb constructions (SVC).

1.3 Tree Adjoining Grammar (TAG)

The Tree Adjoining Grammar (TAG) formalism is a powerful tool to design natural language grammars (s. [Joshi et al. 75]). TAG in its standard formulation has more power than context-free grammars, but only mildly so ([Joshi 85]).

The most significant difference from context-free based grammar formalisms is that TAG operates on trees as elementary data structures rather than on strings. Therefore, TAG is a tree-generating system. Larger, more complex trees are derived from a *finite set of elementary trees* by means of adjunction, the only combining operation admitted in the standard version. *Initial trees* and *auxiliary trees* constitute the set of elementary trees. An initial tree is a phrase-structure like, complete unit, an auxiliary tree is a recursive structure which can be inserted into the body of a (possibly modified) initial tree.

There are two basic ideas underlying TAG. First, elementary trees are minimal linguistic structures in which grammatical contraints can be localized, in this way providing a domain for stating dependencies, such as subcategorization, filler-gap relations and agreement phenomena between elements of a tree.

The second basic idea is that adjunction factors recursion into a finite set of elementary trees, in this way deriving complex structures. An auxiliary tree β, say with root and foot node labelled X, is inserted into an initial tree α as follows[4]: The subtree α dominated by X, say X-tree is removed, leaving a copy of X behind. Then, β is attached at X and X-tree is reattached at the foot node of β, resulting in the newly derived tree γ (again, s. fig. 2).

Numerous TAG variations have developed from the standard version. In the following, the ones which are relevant for the TAG-GEN syntactic generator will be introduced.

[4] α refers to the source tree, β to the tree that adjoins into α, and γ to the derived tree.

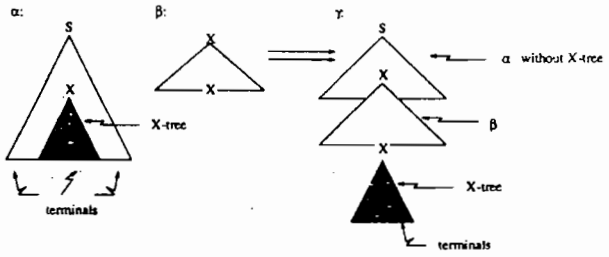

Figure 2: Adjunction of β into α

TAG has been extended by a second combining operation which has context-free power. For **TAG with Substitution**, there may be non-terminal leaves in the trees of $\mathcal{I}$ and $\mathcal{A}$ called 'substitution nodes'. They are marked by a down arrow ($\downarrow$) in order to differentiate them from foot nodes in auxiliary structures.

Substitution operates on two initial trees, α, and β. β has a root node labelled X, α has a non-terminal node also labelled X at its frontier. During substitution, β attaches at the node X in α, yielding γ (fig. 3).

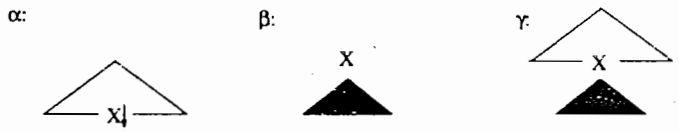

Figure 3: Substitution of β into α

Each elementary tree in a **Lexicalized TAG (LTAG)** specifies a lexical item as head element which is marked by '*' (or $\Diamond$) [Joshi&Schabes 91]. It serves as *anchor* in the lexicon and makes it possible to select the tree when its anchor is specified in the input to the syntactic generator. Substitution nodes are used as placeholders for the syntactic realization of the head's complements (which are realized in separate trees).

Another deviation from standard TAG is that root-node categories other than S are admitted, e.g. NP or PP.

The standard formalism can be extended by *associating feature structures at the nodes* to allow compact representation of complex syntactic dependencies.

Feature TAG (FTAG) makes use of an alternative view on nodes of a TAG tree [Vijay-Shanker 92]: Each internal node can reflect two relations, one to its parent, and the other to its children. This observation results in splitting each feature structure associated with a node into a top and a bottom part, representing the relation to the father and the son node, respectively, as is illustrated by the VP node of tree α in figure 4, ("t" stands for top, "b" for bottom).

By separating vertical and horizontal information in a tree, the format of **LD/LP TAG** does away with the implicit coupling of local dominance (LD) and linear precedence (LP) relations of nodes within a tree [Joshi 87]. The elementary structures are no longer trees but simply dominance structures. Linear precedence is defined *separately* on them. The decision to choose the LD/LP format taken by the TAG-GEN developers was enforced by the requirements of incremental generation. Figure 5 shows the initial tree α of figure 4 in TAG-GEN notation, reformulated in terms of LD/LP TAG (omitting feature structures):

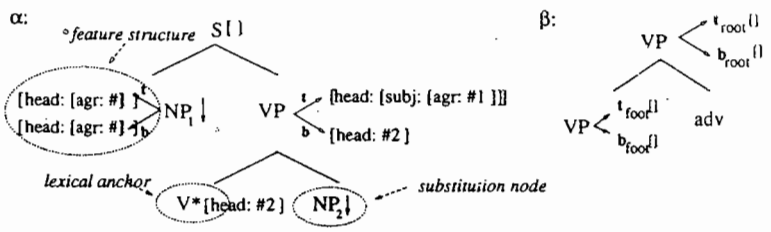

Figure 4: TAG rules with lexical anchor, substitutions nodes and features

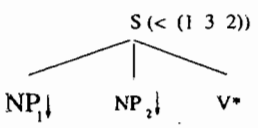

Figure 5: A TAG-GEN rule

1.4 TAG-GEN

The syntactic generator TAG-GEN was designed to simulate spontaneous speech[5] [Kilger&Finkler 95a]. Therefore, the mode of processing is *incremental*, which means the following:

Each given increment triggers its immediate processing such that

- processing starts before the input is complete,
- output increments can be produced before processing is complete and, if possible,
- even before the input is complete.

This means that the system is able to deal with a generator input that does not come in as a whole but in a piece-by-piece fashion, and an output will be generated as soon as there is sufficient information, that is, even *while* the input continues and *before* it is complete. To fulfil the definition above, TAG-GEN proceeds in 4 stages, realizing an object-oriented approach. As underlying grammar formalism, TAG is used, extended by substitution, lexicalization, separation of local dominance and linear precedence information, and feature structures. The starting point for the generator is the output dynamically submitted by the microplanner component (alias *generator input*). The input consists of content words, the 'entities' and the semantic relations between them. For each of these input increments, one object is created. Each object passes through the following processing stages:

1. The Input Interface consumes incremental input. For each object, an adequate syntactic rule, i.e. a TAG grammar rule, is chosen on the basis of the object's input information. At the end of input, the whole of the increments constitute a *global hierarchical net.* Its nodes correspond to the given lexical items, its edges to the semantic relations specified between these items.
2. In the Phrase Formulator, the relation objects combine the local structures of the entity objects into larger structures. They use adjunction or substitution for exchanging relevant data in order to build the global syntactic tree representing the utterance. The attachment to the global structure is necessary for each object to be supplied with contextual information which is used for linearization and inflection.

[5]Note that the grammar used TAG-GEN only as a testbed for the first version of MLWFA. In fact, MLWFA uses the FB-LTAG generator which is not incremental while the microplanner output for MLWFA will be very similar to TAG-GEN's. It can be expected, however, that the grammar can be modified at reasonable costs.

For adjunction and substitution, the gap between substitution nodes and other lexicalized trees that substitute those nodes must be bridged. For this, the relations between the entities are mapped onto so-called *intermediate structures* that realize the syntactic relations between two lexemes (s. figure 7) below.

3. At the Linearization Level, the goal is to serialize the terminals into the order given by the linearization rule which is annotated at the respective grammar rule. In TAG-GEN, the linearization component also evaluates information about inflection specified in the generator input.

4. The Output Interface produces incremental output by synchronizing the output activities of all objects.

1.5 A Generation Example

In this section, the generation of the target string

(1) "Da4-feng cong2 bei3-fang yan2-shen zhi Jiang-nan2"
wind from north extend to Jiang-Nan
'There will be winds from the north down to Jiang-Nan.'

will be traced, starting from the generator input as is specified in figure 6.

Stage I: The Interface

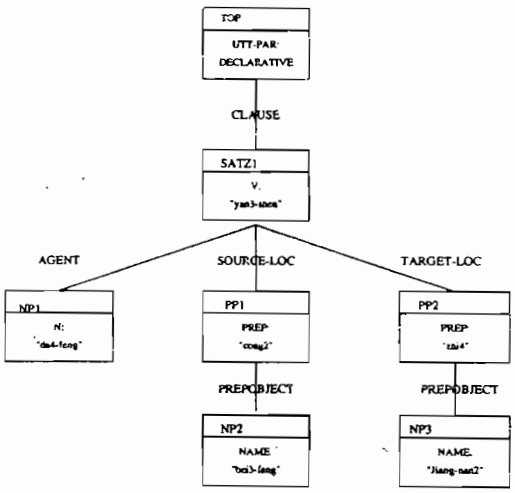

Figure 6: A Generator Input

The *input entities* are mapped onto corresponding lexical entries which themselves select appropriate grammar trees. *Relation entities* are mapped onto intermediate structures as shown in figure 7[6].

Generation always starts from the top entity UTT-PAR (see same figure) by means of which several kinds of maximal projections such as sentences, noun phrases or prepositional phrases can be generated. Derivation is made possible by an intermediate tree rooted in $S_{UTT-PAR}$.

To ease the graphical representation, the step of an entity of the generator input being mapped onto the lexical entry will be skipped. Instead, I show the direct mapping from the input to the corresponding grammar rules. "yan3-shen" is a transitive verb, taking a noun phrase subject

[6]Mapping of entities to basic structures is marked by a dashed line, mapping onto intermediate structures by a dotted line.

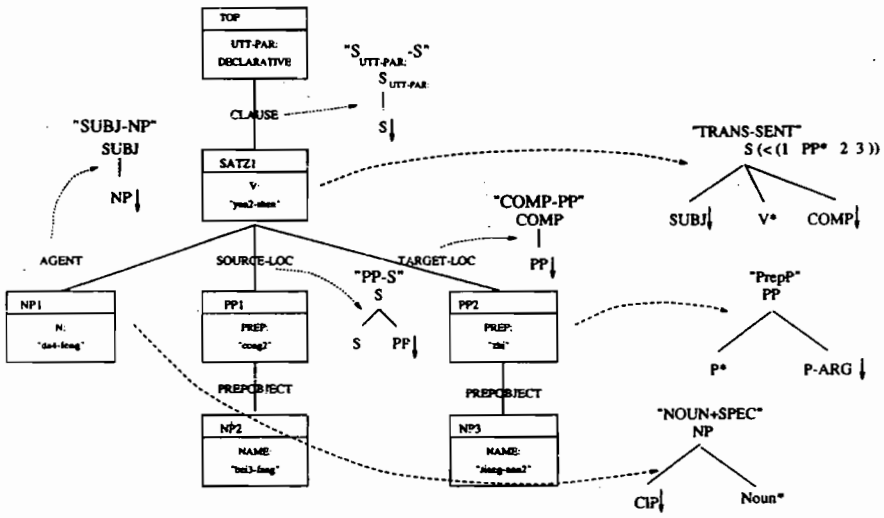

Figure 7: Entities select appropriate grammar rules

and a prepositional phrase object as complements. The PP is connected to "yan3-shen" via the semantic relation named TARGET-LOC (s. fig. 6).

"yan3-shen" is mapped onto the tree named "TRANS-SENT", "da4-feng" onto the tree "NOUN+SPEC and, finally, "cong2 bei3-fang" and "zhi Jiang-nan2" are mapped onto the tree "PrepP" (s. fig. 7).

Stage 2: The Phrase Formulator Relation nodes are mapped onto intermediate structures[7]. For example, they connect the substitution node SUBJ↓ of the tree "yan2-shen" with "da4-feng" 's realization as a noun phrase'. The same is done for the substitution node COMP↓. It is filled by the PP "zhi Jiang-nan2" via the relation TARGET-LOC, substituting COMP↓ by PP. Last, for "cong2 bei3-fang", which is linked to "yan2-shen" via the relation SOURCE-LOC (s. figure 6 above), no substitution node in the source tree "TRANS-SENT" is provided (since it is not an argument of the verbal head). Therefore, SOURCE-LOC is mapped onto an *intermediate auxiliary tree*, "PP-S". At the end of this stage, the Phrase Formulator has derived the structure shown in figure 8.

Stage 3: The Linearization Component The linearization rule is applied in the following way: An object traverses its local structure from the root-node downwards and evaluates the linearization rule annotated at every node.

Application of the rule finally yields the target sentence.

1.6 Conclusion

To conclude, the following are the main achievements of this work:

- a computational account of a substantial portion of Mandarin Chinese,
- specification of a grammar for generation in a lexicalized TAG environment,
- integration of the grammar in TAG-GEN and MLWFA.

However, the most complicated tasks from a general point of view were the categorization of words and phrases and the extraction of linearization rules for modifiers. For the latter, no generally applicable solution was found.

[7]In the figure, the connection is marked by dotted lines.

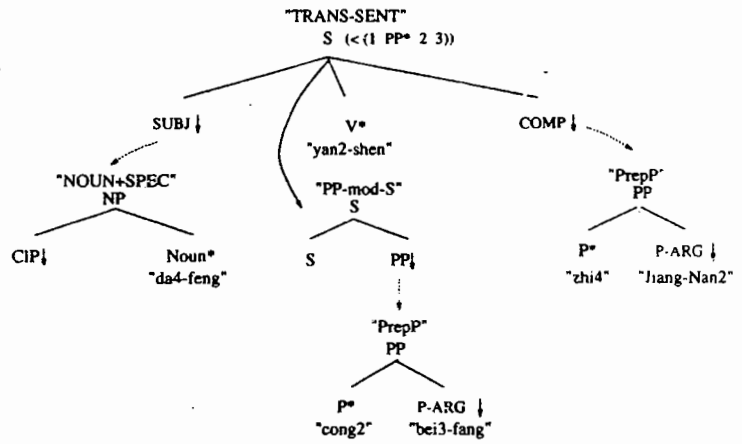

Figure 8: The Phrase Formulator combines elementary trees

I thank Anne Kilger, Xiao-Rong Huang, Tian-Fang Yao, Guo-Dong Gao and Sanjeev Mahajan for fruitful discussions on the paper.

References

[Huang et al. 96a] X.R. Huang, T.F. Yao, G.D. Gao, *Generating Chinese Weather Forecasts with Stylistic Variations*, *Proceedings of 17th International Conference on Computer Processing of Oriental Language, 1996.*

[Huang et al. 96b] X.R. Huang,*Choosing among Architectures for Applied Multilingual NLG*, In *Proceedings of the PRICAI-96 Workshop on Future Issues for Multi-lingual Text Processing*, Cairns, Australia, pp. 25-31, 1996.

[Joshi et al. 75] A.K. Joshi, L.S. Levy, M. Takahashi, *Tree Adjunct Grammars*, Journal of Computer and Systems Science, pp. 136-163, 1975.

[Joshi 85] A.K. Joshi, *An Introduction into Tree Adjoining Grammars*, Technical Report MS-CIS-86-64, LINC-LAB-31, Dept. of Computer and Information Science, Moore School, University of Pennsylvania, 1985.

[Joshi 87] A.K. Joshi, *Word Order Variation in Natural Language Generation*, AAAI '87, Seattle, U.S.A., 1987.

[Joshi&Schabes 91] A.K. Joshi, Y. Schabes, *Tree Adjoining Grammar and Lexicalized Grammars*, in: *Definability and Recognizability of Sets of Trees*, Nivat and Podelski (eds.), Elsevier, 1991.

[Kilger&Finkler 95a] A. Kilger, W. Finkler *Incremental Generation for Real-Time Applications*, DFKI Report RR-95-11, German Research Center for Artificial Intelligence (DFKI), 1995.

[Vijay-Shanker 92] K. Vijay-Shanker, *Using Descriptions of Trees in a Tree Adjoining Grammar*, Computational Linguistics, vol. 18(4), pp. 481-517, 1992.

[Wang 97] Y.-F. Wang, *A Lexicalized Tree Adjoining Grammar for the Generation of Chinese Weather Forecasts*, Diploma Thesis, Dept. of Computational Linguistics, Saarbrücken, Germany, 1997