# An Efficient Kernel for Multilingual Generation in Speech–to–Speech Dialogue Translation

**Tilman Becker** and **Wolfgang Finkler** and **Anne Kilger** and **Peter Poller**
German Research Center for Artificial Intelligence (DFKI GmbH)
Stuhlsatzenhausweg 3
D–66123 Saarbrücken
Germany
becker@dfki.de, finkler@dfki.de, kilger@dfki.de, poller@dfki.de

## Abstract

We present core aspects of a fully implemented generation component in a multilingual speech-to-speech dialogue translation system. Its design was particularly influenced by the necessity of real-time processing and usability for multiple languages and domains. We developed a general kernel system comprising a microplanning and a syntactic realizer module. The microplanner performs lexical and syntactic choice, based on constraint-satisfaction techniques. The syntactic realizer processes HPSG grammars reflecting the latest developments of the underlying linguistic theory, utilizing their pre-processing into the TAG formalism. The declarative nature of the knowledge bases, i.e., the microplanning constraints and the HPSG grammars allowed an easy adaption to new domains and languages. The successful integration of our component into the translation system Verbmobil proved the fulfillment of the specific real-time constraints.

## 1 Introduction

In this paper we present core aspects of the multilingual natural language generation component VM-GECO[1] that has been integrated into the research prototype of Verbmobil (Wahlster, 1993; Bub et al., 1997), a system for spontaneous speech–to–speech dialog translation.

In order to achieve multilinguality as elegantly as possible we found that a clear modular separation between a language–independent general kernel generator and language–specific parts which consist of syntactic and lexical knowledge sources was a very promising approach. Accordingly, our generation component consists of one kernel generator and language–specific knowledge sources for the languages used in Verbmobil: German and English with current work on Japanese.

Additionally, the kernel generator itself can be modularized furthermore into two separate components. The task of the so–called *microplanning* component is to plan an utterance on a phrase– or sentence–level (Hovy, 1996) including word–choice (section 2). It generates an annotated dependency structure which is used by the *syntactic generation* component to realize an appropriate surface string for it (section 3). The main goal of this further modularization is a stepwise constraining of the search–space of alternative linguistic realizations, using abstracted views on different choice criteria.

Multilingual generation in dialog translation imposes strong requirements on the generation module. A very prominent problem is the non–wellformedness (incorrectness, irrelevance, and inconsistency) of spontaneous input. It forces the realization of *robust* generation to be able to cope with erroneous and incomplete input data so that the quality of the generated output may vary between syntactically correct sentences and semantically understandable utterances. On the level of knowledge sources this is achieved by using a highly declarative HPSG grammar which very closely reflects the latest developments of the underlying linguistic theory (Pollard and Sag, 1994) and covers phenomena of spoken language. This HPSG is compiled into a TAG grammar in an offline pre-processing step (Kasper et al., 1995) which keeps the declarative nature of the grammar intact (section 3).

Maybe the most important requirement on the generation module of a speech–to–speech translation system is real–time processing. The

[1] VM–GECO is an acronym for "VerbMobil GEneration COmponents."

above mentioned features of VM-GECO contribute to the efficiency of the generation component. The TAG-formalism is well known for the existence of efficient syntactic generation algorithms (Kilger and Finkler, 1995).

In general, all knowledge sources of all modules are declarative. The main advantage is that this allows for an easier adaptation of the generation component to other domains, languages and semantic representation languages besides the easier extendability of the current system. The feasibility of the language adaptation was proved in the Verbmobil project itself where the (originally English) generator was recently extended to cover German and is currently adapted for Japanese. The adaptation to another domain and also to another specification language for intermediate structures was shown in another translation project which uses in contrast to Verbmobil an interlingua based approach (section 4.1).

## 2 The Microplanner

A generation system for target language utterances in an approach to speech-to-speech translation has to work on input elements representing intermediate results of recognition, analysis, and transfer components. In that setting, several of the tasks of a complete natural language generation system such as selection and organization of the contents to be expressed are outside of the control of our generator. They have been decided by the human user of the translation system or they have been negotiated and computed by a transfer component. Nevertheless, there remain a number of different but highly interrelated subtasks of the generation process where decisions have to be made in order to determine and realize the translation result to be sent to a speech synthesis component. The diverse subtasks — often collectively denoted as microplanning (cf. (Levelt, 1989; Hovy, 1996)) — comprise the planning of a rough structure of the target language utterance, the determination of sentence borders, sentence type, topicalization, theme-rheme organization of sentential units, focus control, utilization of nominalized, or infinitival style, as well as triggering the generation of anaphora and lexical choice. In addition, they have to address the problem of expressibility of the selected contents in a text realization component, i.e., bridging the generation gap (see (Meteer, 1990)).

The input to our microplanning component consists of semantic representations encoded in a minimal recursive structure following a variant of UDRT. Each individual indicated by some input utterance is formally represented by a discourse referent. Information about the individual is encoded within the DRS-conditions. Relations between descriptions of different discourse referents lead to a hierarchical semantic structure (see Figure 1 for a graphical representation of fragments of an example input to the generator). Discourse referents are depicted as boxes headed by individual names $i_n$; conditions are illustrated within those boxes.

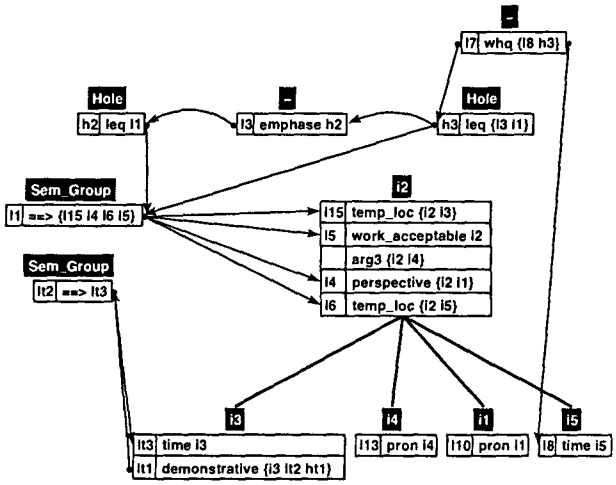


Figure 1: Example Input to the Generator

Besides these input terms from the transfer component, the generator may access knowledge about the dialogue act, the dialogue history as well as some prosodic information of the user's utterance.

The output of the microplanner is a sentence plan that serves as input for the syntactic realization component. It describes a dependency tree over lexical items annotated with syntactic, semantic, and pragmatic information which is relevant to produce an acceptable utterance and guide the speech synthesis component.

### 2.1 Design of the Microplanning Kernel

An important design principle of our generator is the demand to cope with multidirectional dependencies among decisions of the diverse subtasks of microplanning without preferring one

order of decisions over others. E.g., the choice of an interrogative sentence requires an (at least elliptical) verbal phrase as a major constituent of the sentence; nominalization or the choice of passive voice depends on the result of word choice, etc. Therefore, we conceived microplanning as a constraint-satisfaction problem (Kumar, 1992) representing undirected relations between variables. Thereby, variables are created for elements in the input to the generator. They are connected by means of weighted constraints. The domains of the variables correspond to abstractions of possible alternatives for syntactic realizations of the semantic elements including sets of specifications of lexical items and syntactic features. A solution of the constraint system is a globally consistent instantiation of the variables and is guaranteed to be a valid input for the syntactic generation module. Since there might be locally optimal mappings that lead to contradiction on a global level, the microplanner generally uses these weighted constraints to direct a backtracking or propagation process.

One the one hand, the advantages of utilizing a constraint system lie in the declarativity of the knowledge sources allowing for an easier adaptation of the system to other domains and languages. We benefited from this design decision and realized microplanning for English and German by means of merely establishing new rule sets for lexical and syntactic choice. The core engine for constraint processing was reused without modification. On the other hand, having defined a suitable representation of the problem to be solved, a constraint–based approach also establishes a testbed for examining the pros and cons of different evaluation methods, including backtracking, constraint propagation, heuristics for the order of the instantiation of variable values, to name a few means of dealing with competition among alternatives and to find a solution.

The microplanner makes use of the minimal recursive structure of its semantic input term (see Fig. 1) by triggering activities by bundles of conditions, discourse referents, and holes representing underspecified scope relations in the input. These three input categories are reflected by different microplanning rule sets that are applied conjointly during the process of microplanning. The rules are represented as pattern–condition–action triples. A pattern is to be matched with part of the input, a condition describes additional context-dependent requirements to be fulfilled by the input, and the action part describes a bundle of syntactic features realizing lexical entities and their relations to complements and modifiers.

A microplanning rule for the combination of the semantic predicates WORK_ACCEPTABLE, ARG3, and PERSPECTIVE which get realized as a finite verb, i.e., representing a 3:1 mapping of semantic predicates to a syntactic specification is shown in Figure 2.

```
;; standard finite verb with 2 complements
((WORK_ACCEPTABLE (L I) ARG3 (L I I2) ;; pattern
  PERSPECTIVE (L1 I I3))
 ($not ($sem-match NOM (L I)))      ;; condition
 (WORK_ACCEPTABLE (CAT V)           ;; action
  (HEAD (OR SUIT_V1 SUIT_V2)) (FORM ordinary)
  (TENSE $get-tense I) (VOICE $get-voice I))
 (I2 (GENDER (NOT MAS FEM)))
 (REGENT-DEP-FUNC WORK_ACCEPTABLE I2 AGENT)
 (REGENT-DEP-FUNC WORK_ACCEPTABLE I3 PATIENT)
 (KEY KEY-V))
;; nominalized form ...
```

Figure 2: Example Microplanning Content Rule

In the condition part of the verbal mapping the existence of a NOM-condition within the semantic input information is tested. It would forbid the verbal form by demanding a nominalized form. The action part describes the result of lexical selection (the lemma "suit") plus generic functions for computing relevant syntactic features like tense and voice. I2 which stands for the ARG3 of WORK_ACCEPTABLE, defined by a database of linking–information as the semantic agent is characterized as neither allowing gender masc(uline) nor fem(inine) for preventing *"he suits"* in the sense of *"he is okay"*. Entries starting with KEY define identifiers used for computing the preference value of a microplanning rule with respect to the given situation. In an additional database, KEYs are associated with weights for predefined situation characteristics such as time pressure, or register. The microplanning content rules are not directly entered by a rule writer but are compiled off–line from several knowledge sources for lexical choice rules, rules for syntactic decisions and linking rules, thereby filtering out contradictory combinations without requiring on–line runtime.

Regarding the sets of alternatives that result

from the application of the microplanning rules, the most direct way of realizing a constraint net seems to be the definition of one variable for each condition, discourse referent, and hole, leading to a variable net as shown in Figure 3.

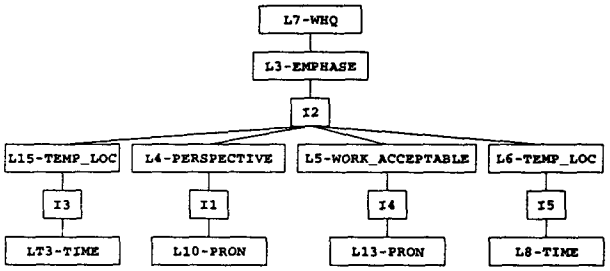


Figure 3: Variable Net for Microplanning

For our task, it is not enough to define binary matching constraints between each pair of variables that purely test the compatibility of the described syntactic features. Some syntactic specifications may contain identifications of further entities, e.g., discourse referents and syntactic identifiers which influence the result of the compatibility test between a pair of variables referring to these identifiers. Thus, the constraint net is not easily subdivided into subnets that can be efficiently evaluated. The large number of combinations of alternative values is handled by known means for CSP such as uniting variables with 1–value domains and applying matching mechanisms to their values, computation of 2–consistency by matching value pairs and filtering out inconsistent ones, storing and reusing knowledge about binary incompatibility and performing intelligent backtracking.

The result of the constraint solving process for the input shown in Fig. 1 is given in Fig. 4.

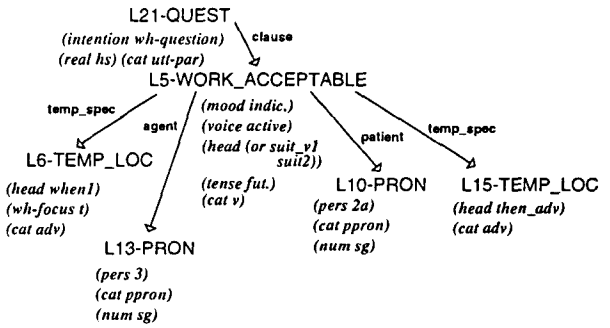


Figure 4: Microplanning Result for the Example

## 3 The Realizer

The syntactic realizer[2] proceeds from the microplanning result as shown in Figure 5. It produces a derived phrase structure from which the output string is read off. The realizer is based on a fully lexicalized grammar in the sense that every lexical item selects for a finite set of possible phrase structures (called elementary trees). In particular, we use a Feature-Based Lexicalized Tree-Adjoining Grammar (FB-LTAG, see (Vijay-Shanker and Joshi, 1988; Schabes et al., 1988)) that is derived from an HPSG grammar (see section 4 for some more details). The elementary trees (see Figure 9) can be seen as maximal partial projections. A derivation of an utterance is constructed by combining appropriate elementary trees with the two elementary TAG operations of adjunction and substitution.

For each node (i.e., lexical item) in the dependency tree, the *tree selection* phase determines the set of relevant TAG trees. A first tree retrieval step maps every object of the dependency tree into a set of applicable elementary TAG trees. The main tree selection phase uses information from the microplanner output to further refine the set of retrieved trees. The *combination* phase finds a successful combination of trees to build a (derived) phrase structure tree. The final *inflection* phase uses the information in the feature structures of the leaves (i.e., the words) to apply appropriate morphological functions. An initial preprocessing phase is needed to accommodate the handling of auxiliaries which are not determined in microplanning. They are derived from the tense, aspect and sentence mood information as supplied by microplanning.

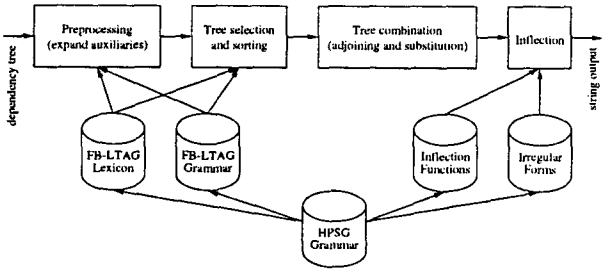


Figure 5: Steps of the syntactic generator.

The two core phases are the tree selection and

[2]A more detailed description is contained in (Becker, 1998).

the combination phase. The tree selection is driven by the HPSG instance or word class that is supplied by the microplanner. It is mapped to a lexical type by a lexicon that is automatically compiled from the HPSG grammar. The lexical types are then mapped to a tree family, i.e., a set of elementary TAG trees representing all possible minimally complete phrase structures that can be build from the instance. The additional information in the dependency tree is then used to add further feature values to the trees. This additional information acts as a filter for selecting appropriate trees in two stages:

Some values are incompatible with values already present in the trees. These trees can therefore be filtered immediately from the set. E.g., a syntactic structure for an imperative clause is marked as such by a feature and can be discarded if a declarative sentence is to be generated. Additional features can prevent the combination with other trees during the combination phase. This is the case, e.g., with agreement features.

The combination phase completely belongs to the core machinery. It can be exchanged with more efficient algorithms without change of the grammar or lexicon. It explores the search space of all possible combinations of trees from the candidate sets for each lexical item (instance). Since there is sufficient information available from the microplanner result and from the trees, a well-guided best-first search strategy can be employed in the current system.

As part of the tree selection phase, based on the rich annotation of the input structure, the tree sets are sorted locally such that preferred trees are tested first. Then a modified backtracking algorithm traverses the dependency tree in a bottom-up fashion[3]. At each node and for each subtree in the dependency tree, a candidate for the phrase structure of the subtree is constructed. Then all possible adjunction or substitution sites are computed, possibly sorted (e.g., allowing for preferences in word order) and the best candidate for a combined phrase structure is returned. Since the combination of two partial phrase structures by adjunction or substitution might fail due to incompatible feature structures, a backtracking algorithm must be used. A partial phrase structure for a subtree of the dependency is finally checked for completeness. These tests include the unifiability of all top and bottom feature structures and the satisfaction of all other constraints (e.g., obligatory adjunctions or open substitution nodes) since no further adjunctions or substitutions will occur in this subtree.

The necessity of a spoken dialog translation system to robustly produce output calls for some relaxations in these tests. E.g., 'obligatory' arguments may be missing in the utterance. This can be caused by ellipsis in sentences such as *"Ok, we postpone."* or by false segmentations in the analysis such as segmenting *"Wir sollten (we should) das Treffen verschieben (the meeting postpone)."* into two segments *"Wir sollten"* and *"das Treffen verschieben"*. In order to generate *"postpone the meeting"* for the second segment, the tests in the syntactic generator must accept a phrase with a missing subject if no other complete phrase can be generated.

Figure 6 shows a combination of the tree retrieval and the tree selection phases. In the tree retrieval phase for `L5-WORK_ACCEPTABLE`, first the `HEAD` information is used to determine the lexical types of the possible realizations `SUIT_V1` and `SUIT_V2`, namely `MV_NP_TRANS_LE` and `MV_EXPL_PREP_TRANS_LE` respectively[4]. These types are then mapped to their respective sets of elementary trees, a total of 25 trees. In the tree selection phase, this number is reduced to six. For example, the tree MV_NP_TRANS_LE.2 in Figure 9 has a feature CL-MODE with the value IMPERATIVE. Now, the microplanner output for the root entity `LGV1` contains the information `(INTENTION WH-QUESTION)`. The `INTENTION` information is unified with all appropriate CL-MODE features, which in this case fails. Therefore the tree MV_NP_TRANS_LE.2 is discarded in the tree selection phase.

The combination phase uses the best-first bottom-up algorithm described above to determine one suitable tree for every entity and also a target node in the tree that is selected for the governing entity. For the above example, the selected trees and their combination nodes are

[3]The algorithm stores intermediate results with a memoization technique.

[4]`MV_NP_TRANS_LE` is an abbreviation for "Main Verb, NP object, TRANSitive Lexical Entry" used in sentences like *"Monday suits me."*

```
;; traverse for: L5-WORK_ACCEPTABLE
    returned MV_NP_TRANS_LE
    returned MV_EXPL_PREP_TRANS_LE
    total: 6 trees
;; traverse for: L13-PRON
    returned PERS_PRO_LE
    total: 1 tree
;; traverse for: L10-PRON
    returned PERS_PRO_LE
    total: 1 tree
;; traverse for: L6-TEMP_LOC
    returned WH_ADVERB_WORD_LE
    total: 2 trees
;; traverse for: L15-TEMP_LOC
    returned NP_ADV_WORD_LE
    total: 5 trees
;; traverse for: LGV1
    returned WILL_AUX_POS_LE
    total: 2 trees
```

Figure 6: An excerpt from the tree retrieval and selection phase.

shown in Figure 7[5].

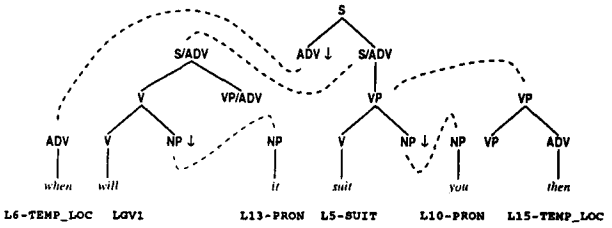


Figure 7: The trees finally selected for the entities of the example sentence.

Figure 8 shows the final phrase structure for the example. The inflection function selects the base form of "suit" according to the BSE value of the VFORM feature and correctly uses *"will."* Information about the sentence mode WH-QUESTION can be used to annotate the resulting string for the speech-synthesis module.

## 4 Results

Our approach to separate a generation module into a language–independent kernel and language–specific knowledge sources has been successfully implemented in a dialogue translation system. Furthermore, the mentioned adaptability to other generation tasks has also been proved by an adaptation of the generation module to a new application domain and also to a completely different semantic representation

[5]Note that the node labels shown in Figures 7 and 8 are only a concession to readability. The TAG requirement that in an auxiliary tree the foot node must have the same category label as the root node is fulfilled.

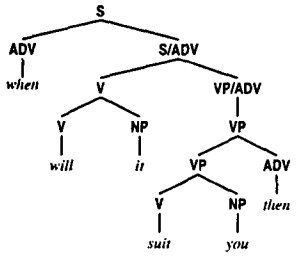


Figure 8: The final phrase structure for *"When will it suit you then?"*

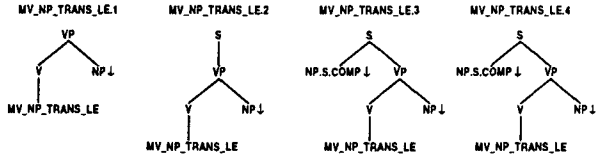


Figure 9: Some of the trees for transitive verbs. They are compiled from the corresponding lexical type MV_NP_TRANS_LE as defined in the HPSG grammar. Trees 3 and 4 differ only with respect to their feature structures which are not shown in this figure.

language by adapting the microplanning knowledge sources to the new formalism.

VM–GECO is fully implemented (in Common Lisp) and integrated into the speech–to–speech translation system Verbmobil for two output languages, English and German. The adaptation to Japanese generation will be performed in the current project phase. Our experience from adding German makes us confident that this can be done straightforwardly by creating the appropriate knowledge sources without modifications of the kernel generator. To give the reader a more detailed impression of the implementation of the generation component we present some characteristic data of the English generator. The numbers for the German system, especially for lexicon and processing time, are similar.

The underlying English grammar is a lexicalized TAG which consists of 2844 trees. These trees were transformed during an offline preprocessing step from 2961 HPSG lexical entries of the linguistically well motivated English HPSG grammar written at CSLI. On the other hand the microplanner's knowledge sources consist of 2730 partially pre-processed microplanning rules which are utilized in an in-

tegrated handling of structural and lexical decisions based on constraint propagation. The microplanning rules are of course especially adapted to the underlying semantic representation formalism. Furthermore, the underlying lexicon covers the word list that has been constructed from a large corpus of the application domain of the Verbmobil system, i.e., negotiation dialogues in spontaneous speech.

The TAG grammar resulting from the compilation step allows for highly efficient lexically driven robust syntactic generation mainly consisting of tree adjoinings, substitutions, and feature unifications. The average overall generation time per sentence (up to length 24) is 0.7 seconds on a SUN ULTRA-1 machine, 68 % of the runtime are needed for the microplanning while the remaining 32 % of the runtime are needed for syntactic generation.

### 4.1 Reusing the Kernel

Beside the usability for multiple languages in Verbmobil our kernel generation component has also proven its adaptability to a very different semantic representation language (systematically and terminologically) in another still ongoing multilingual (currently 12 languages) translation project. The project utilizes an interlingua–based approach to semantic representations of utterances. The goal of this project is to overcome the international language barrier which is exemplarily realized by a large corpus improvement of the transparency of consisting of international law texts. Our part in this project is the realization and implementation of the German generation component. Because of our language–independent core generator the adaptation of the generation component to this semantic representation decreased to the adaptation of the structural and lexical knowledge bases of the microplanning component and appropriate domain–specific extensions on the lexicon of the syntactic generator. With an average sentence length of 15 words the average runtime per sentence on a SUN ULTRA-2 is less than 0.5 seconds. Currently, even the longest sentence (40 words) needs under 2 seconds runtime.

Within Verbmobil, the generation component will also be used for text generation when producing protocols as described in (Alexandersson and Poller, 1998).

## References


J. Alexandersson and P. Poller. 1998. Towards multilingual protocol generation for spontaneous speech dialogues. In *9th INLGW*, Niagara-on-the-lake, Canada.

T. Becker. 1998. Fully lexicalized head–driven syntactic generation. In *9th INLGW*, Niagara-on-the-lake, Canada.

Th. Bub, W. Wahlster, and A. Waibel. 1997. Verbmobil: The combination of deep and shallow processing for spontaneous speech translation. In *Proceedings of ICASSP '97.*

E. Hovy. 1996. An overview of automated natural language generation. In X. Huang, editor, *Proc. of the Intl. Symposium on NL Generation and the Processing of the Chinese Language, INP(C)-96*, Shanghai, China.

R. Kasper, B. Kiefer, K. Netter, and K. Vijay-Shanker. 1995. Compilation of HPSG to TAG. In *33rd ACL*, Cambridge, Mass.

A. Kilger and W. Finkler. 1995. Incremental generation for real–time applications. Research Report RR–95–11, DFKI GmbH, Saarbrücken, Germany, July.

V. Kumar. 1992. Algorithms for constraint–satisfaction problems: A survey. *AI Magazine*, 13(1):32–44.

W.J.M. Levelt. 1989. *Speaking: From Intention to Articulation.* The MIT Press, Cambridge, MA.

M.W. Meteer. 1990. *The "Generation Gap" - The Problem of Expressibility in Text Planning.* Ph.D. thesis, Amherst, MA. BBN Report No. 7347.

C. Pollard and I. A. Sag. 1994. *Head-Driven Phrase Structure Grammar.* Studies in Contemporary Linguistics. University of Chicago Press, Chicago.

Y. Schabes, A. Abeillé, and A. K. Joshi. 1988. Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars. In *COLING-88*, pages 578–583, Budapest, Hungary.

K. Vijay-Shanker and A. K. Joshi. 1988. Feature structure based tree adjoining grammars. In *COLING-88*, pages 714–719, Budapest, Hungary.

W. Wahlster. 1993. Verbmobil: Translation of face–to–face dialoges. In *MT Summit IV*, Kobe, Japan.