GPSG-BASED PARSING AND GENERATION

James Kilbury

1. Introduction and motivation

Generalized Phrase Structure Grammar (GPSG) is a major linguistic theory developed in the past few years by Gerald Gazdar and other linguists (cf Gazdar, Pullum 1982; Gazdar, Klein, Pullum, and Sag 1984). The theory encompases syntax, semantics, and the lexicon and is especially noteworthy for its strong theoretical orientation toward parsing (cf Sampson 1983).

Existing natural language (NL) systems for parsing and generation typically show one or more of the following inadequacies:

(1) The system lacks a theoretical linguistic foundation. Due to its conception the system is in principle unable to cope with certain linguistic phenomena, or the underlying analyses of the phenomena do not achieve explanatory adequacy.
(2) The system is poorly modularized in the following respects:
   a) The parser and grammar are mixed, so that the parser is difficult to adapt to a different natural language.
   b) Separate grammars with duplicate information are needed for parsing and generation due to (a) and because the grammatical knowledge is represented procedurally.
   c) Semantics is mixed with syntax in certain ways that hamper the adaptation of the system to a new representation language.

The goal of the work described in this paper is to overcome the particular deficiencies described above in a NL system for parsing and generation in German that is based on GPSG. The system will constitute the basis for a general NL interface for man-machine dialog. Although world knowledge and pragmatics will not be dealt with directly, their integration in the system will be provided for.

The aim of the system is to translate from NL into the special semantic representation language SRL and vice versa. Since GPSG matches each syntactic rule with a corresponding semantic rule, syntactic and semantic analysis take place simultaneously during parsing. This facilitates the integration of inferential processes involving world knowledge and pragmatics mentioned above.

Although SRL has been chosen as representation language, the grammar is independent of SRL, just as the parser and grammar are independent of each

other. The system can be adapted to different representation languages in either of two ways:

(1) The semantic parameters (see below) of the grammar rules and the lexicon can be modified.

(2) For any parser and generator a bidirectional transducer SRL(1) <--> SRL(2) can be constructed as an interface, which is the simpler solution.

While SRL is well-suited as the representation language for a NL system, it must undergo considerable further development before such a system will actually be operational. Part of this work, especially that involving compositional semantics (i.e. the semantic rules of the grammar that match with syntactic rules) is closely linked with the treatment of syntax. The ultimate adequacy of the parser output depends on that of the lexical semantic representations, however, which will derive mainly from artificial intelligence (AI) investigations of conceptual knowledge.

The novelty of GPSG lies principally in its treatment of syntax, and the parser (in the AI sense of a component producing semantic representations) proposed here is syntactically rather than semantically oriented, since it carries out a full syntactic analysis of the NL input. It is generally acknowledged that some syntactic analysis is essential even in a semantically-oriented parser (cf Eimermacher in this volume), and the desire to use a single grammar as a common knowledge base for parsing and generation makes a full treatment of syntax inevitable. Furthermore, the syntactic orientation allows the use of and comparison with parsing algorithms from the syntactic analysis of formal languages. This in turn helps to establish the formal properties of the NL parser. The parser of this paper is nevertheless semantic in the sense that it also carries out a full semantic analysis.

GPSG, like most of the other linguistic work on syntax in the past quarter century, is an outgrowth of the theory of generative-transformational grammar introduced by Noam Chomsky. It rejects, however, the claim - presented in Chomsky (1957) and thereafter largely left unquestioned - that context-free grammars are in principle incapable of capturing the generalizations necessary for adequate descriptions of natural languages. Transformational rules were introduced to deal with the phenomena for which context-free rules were allegedly inadequate, and Augmented Transition Networks (ATNs; cf Woods 1970) were in turn developed in part to cope with the difficulty of parsing with transformations. After Peters and Ritchie (1969) it became apparent that the formalism of Chomsky (1965) was too powerful, since it was equivalent to an

unrestricted rewrite system or universal Turing machine. Subsequent work in transformational grammar has largely been aimed at reducing the power of its formalism, but the success of these efforts is questionable (cf Pullum 1983).

Generally speaking, the greater the formal power of a grammar is, the greater the complexity and the less the efficiency of the automaton that recognizes or parses the corresponding language. Of course, speed and efficiency are of major importance in any NL computer system, but it is essential to show that the grammar of a proposed syntactic theory has less than type 0 power since the language would otherwise not be decidable, i.e. it would be impossible in the general case to establish whether or not a given input is a sentence of the language. Such a grammar would thus appear to lack cognitive plausibility in an important respect. Unfortunately, many linguistic works on syntactic theory have not been formal and explicit enough to allow the power of the grammar to be determined. The NL systems of artificial intelligence have often been developed with an engineering approach that pays little attention to these questions, but answers are necessary both for practical reasons and in order to meet the increasing demand for cognitive plausibility in AI systems.

GPSG departs from the fundamental observation that "parsing is easy and quick" (Gazdar 1981: §1) for humans and argues that the transformational component must be radically constrained; this is accomplished through its total elimination. It is furthermore claimed that natural language can be adequately described by a type 2 (context-free) grammar, which would then guarantee the existence of efficient algorithms like that of Earley (1970) for NL parsing. The latter claim is very strong, but if correct, it means that GPSG constitutes an ideal basis for a NL computer system. These considerations have led to the choice of GPSG for the system described here.

2. Concepts of GPSG

The following section seeks to present a very brief and rough sketch of concepts that are central to GPSG. Interested readers will want to consult Gazdar (1981), Gazdar and Pullum (1982), and Gazdar, Klein, Pullum, and Sag (1984). It is impossible due to limitations of space to include adequate examples of the application of these notions in the actual description of natural language, but an extended example including semantics is given by Gazdar, Pullum, and Sag (1982).

A number of the concepts mentioned here (namely, (1), (2), and (7)) are common to modern syntactic theory in general, but the others are treated distinctively in GPSG.

(1) A context-free rule of syntax has the form A --> α, where A is a nonterminal symbol and α a string of nonterminal and (pre)terminal symbols; the latter are understood as nonterminal symbols corresponding to lexical categories like N(oun) and V(erb) that directly dominate terminal symbols (i.e. actual lexical forms like dog and eats). A node A has as its immediate successors (i.e. directly dominates) the members of α. Grammars containing only rules of this form belong to type 2 in the Chomsky hierarchy.

(2) X-bar syntax involves an analysis of nonterminal symbols (grammatical categories) that reveals interrelations between the categories and permits greater generalization in rules. The category of a symbol is given by X while the number of bars (or exponent) denotes the projection level (cf Radford 1981: 79ff). The familiar rules

S --> NP VP VP --> V NP

appear in X-bar notation as

$\overline{\overline{V}}$ --> $\overline{\overline{N}}$ $\overline{V}$ $\overline{V}$ --> V $\overline{\overline{N}}$.

The symbol in the right side of a rule with the same category as the left-side symbol but with a lower projection level is the head of the corresponding construction.

(3) GPSG has developed an elaborate theory of syntactic features, which appear together in a complex expression (tree) as the parameter of a nonterminal symbol. Since syntactic categories and projection levels are also treated as features, the formalism of GPSG in effect provides for parametrized rules using a single nonterminal symbol. Variable features allow rules to express agreement relations, as in a rule for subject-verb agreement like

$\overline{\overline{V}}$ --> $\overline{\overline{N}}$(number(α)) $\overline{V}$(number(α)) .

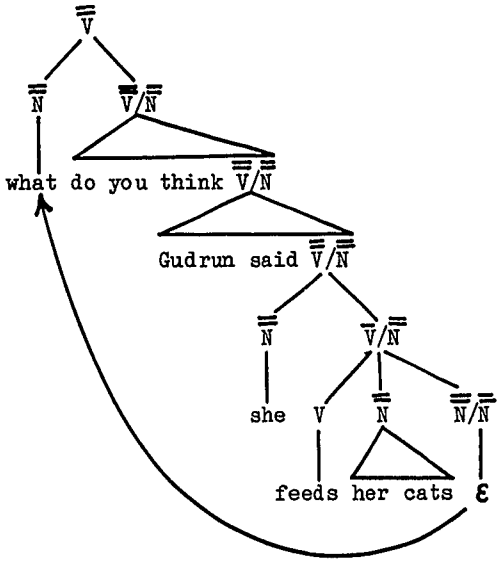
(4) Syntactic features also allow rules like

V(rule(25)) --> geben, ... ('give' in German)

$\overline{V}$ --> $\overline{\overline{N}}$(case(dat)) $\overline{\overline{N}}$(case(acc)) V(rule(25))

which express lexical subcategorization without introducing context-sensitivity in the formalism. Conversely, lexical syntactic features can be used to stipulate that a nonterminal symbol be expanded to a particular lexeme.

(5) Slash categories - or in the present form of GPSG, slash features - have been introduced to capture the grammaticl relations in sentences like



containing unbounded dependencies (here, between feeds and what). The notation α/β denotes an α in which a β is missing, while α/α is like a trace (cf Radford 1981: 191ff) having no lexical realization.

(6) A context-free rule S --> A B C simultaneously expresses information about immediate dominance (that S immediately dominates the successors A, B, and C) and about linear precedence (the order of the successors). In the ID/LP formalism this information is expressed in separate rules, so that generalization about partial ordering becomes possible. The ID/LP rules

S --> A, B, C

B < C

correspond to the context-free rules

S --> A B C

S --> B A C

S --> B C A,

which fail to capture the generalization B < C, however.

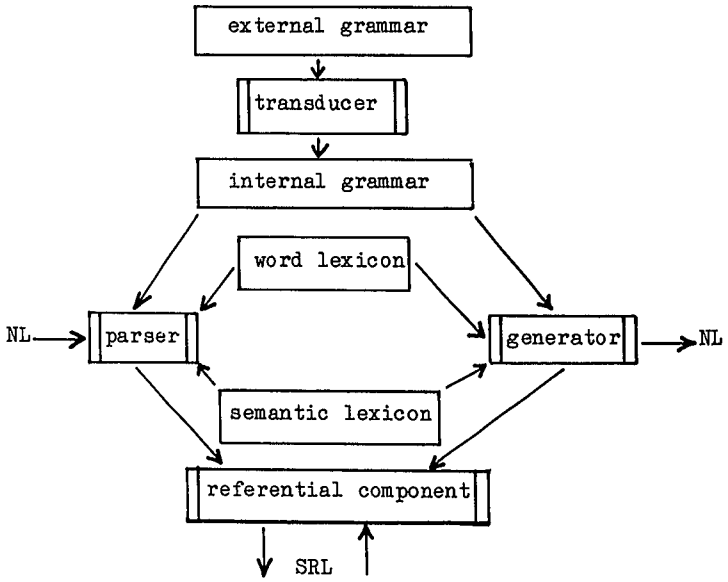
(7) GPSG adopts the rule-to-rule hypothesis according to which each ID rule is matched by a schema that indicates how the representation of the corresponding phrase is built up from the representations of its immediate constituents (cf Gazdar, Pullum, and Sag 1982: 593 and below). This hypothesis establishes a strict parallelism between syntax and semantics.

(8) Metarules of the form α ==>β, where α and β are ID-rule schemata, specify that for every ID rule of the form α there is a corresponding rule of the form β. This device together with the slash categories or features is used

to account for many of the phenomena for which Chomsky proposed transformations.

3. System architecture

The system under development shows the following architecture:



The external grammar written in normal GPSG notation is transduced into the PROLOG-oriented internal form, which then serves as a grammatical knowledge base for the parser and generator. This distinction between the internal and external grammar is important for the modularization of the system; it facilitates the work of linguists with minimal programming knowledge and allows a simple adjustment of the system to other natural languages. Furthermore, the transduction process not only transforms the notation of the grammar but also constructs tables which are crucial for efficient parsing (see below).

Note that the grammar, parser, and generator are modularized and that the latter accept an arbitrary grammar conforming to the GPSG formalism.

The word lexicon contains lexemes of the natural language together with morphological and syntactic information and pointers to entries in the semantic lexicon. The latter contains abstract sememes (i.e. elementary

semantic units) together with the semantic patterns in which they appear and pointers to the corresponding lexemes. This use of distinct lexica minimizes redundancy in the representation of lexical knowledge. It allows the efficient retrieval of semantic representations for lexemes in the parsing process and, conversely, of lexemes for sememes in generation. In the general case the relation between lexemes and sememes is one-to-many in both directions.

The construction of semantic representations in the parsing process follows the method of Knuth (1968) for programming languages. Nonterminal symbols have the general form Cat(σ,ψ) in the external grammar and Cat(θ,σ,ψ) in the internal grammar, where θ,σ, and ψ are variable or partially instantiated syntactic derivation trees, syntactic feature trees, and semantic representations, respectively. Given an ID/LP rule of the form k:A -->α where α = ... A' ... and A' is the head of A (see above), A and A' have a common representation schema ψ, which is a function of the representation schemata of the successors in α (cf rule-to-rule hypothesis above). When a particular syntactic phrase corresponding to A is parsed, ψ is instantiated through unification with the representations of terminal symbols (lexemes). In cases where ψ cannot be constructed solely through unification of congruent representations, semantic procedures may be called using the rule identifier k.

These mechanisms suffice to resolve many instances of syntactic ambiguity, as in

(1) the apple tree beside the path bearing fruit

(2) the apple tree beside the path leading to the train station

(examples from Camilla Schwind). They also provide for idioms like Hans kicked the bucket. which have a literal reading corresponding to a representation constructed compositionally as well as an idiomatic reading, which can be lexically specified.

At present the referential component is a catch-all intended, among other things, to resolve certain anaphors and ellipses. Since there is common linguistic knowledge that is used both in parsing and generation, it would be desirable to introduce a modularization of the component parallel to the grammar/parser/generator division. The architecture must ultimately be extended to allow full interaction with inferential processes, which are essential to resolve ambiguities as in The dog is our best friend. (generic or nongeneric the) or John saw the Rocky Mountains flying to Los Angeles. (which requires knowledge about the world).

Work on the generator has only begun, but it appears possible to use the same grammar as a knowledge base. The algorithm of the generator itself must be constructed so as to fulfill criteria of cognitive plausibility; here it may be possible to incorporate other work on generation such as that of Kempen and Hoenkamp (1982). The central problem appears to be that of lexical selection as reflected in the following German examples:

(1a) Fritz ißt ein Steak. /Fritz is eating a steak./
(1b) Die Katze frißt einen Fisch. /The cat is eating a fish./
(2a) Hans fährt nach Hamburg. /Hans is going to Hamburg./
(2b) Hans fährt zu Erika. /Hans is going to visit Erika./

In both sentence pairs contextual restrictions determine different lexical realizations of a single sememe. Such cases can be handled effectively with the semantic lexicon using devices already developed for SRL (cf SEMNET 1981: 36, 130).

4. Implementation

The system is being implemented in Waterloo PROLOG, Version 1.4, and runs on an ITEL AS/5-7031 (IBM 370) with VM/SP operating system at the Technical University of Berlin.

A parser with a corresponding grammar transducer for direct parsing with the ID/LP formalism (i.e. without expansion of the ID/LP grammar into a corresponding context-free grammar) has already been implemented and is described briefly in Kilbury (1984a) and in detail in Kilbury (1984b). Trial implementations based on indirect parsing were made first. For indirect parsing the external grammar in the GPSG formalism was transduced into an internal grammar containing only context-free rules; the latter was then implemented as a Definite Clause Grammar as described in Pereira and Warren (1980). GPSG was thus reduced to the status of a notational device allowing the linguist to state generalizations when writing the external grammar. It soon became clear that such an implementation would lead to enormous size and redundancy in the internal grammar and to inacceptable inefficiency in a practical NL system. Furthermore, the DCG implementation, while useful for certain purposes amounts to a top-down, left-right, depth-first parser with the built-in backtracking of PROLOG. The algorithm of Earley (1970) for context-free languages appears to be more efficient and has been modified by Shieber (1983) for direct parsing with ID/LP grammars. The DCG implementation

was therefore abandoned for this system, but the recent work of Pereira and Warren (1983) leaves it as an open question.

The present parser is based on the Earley-Shieber algorithm but shows a basic modification of the latter. In the transduction process a first-relation (table) F is constructed that specifies the numbers of rules in which a given nonterminal symbol can be introduced as left-most successor. Whereas the Earley algorithm has the basic cycle predict with grammar - new symbol - complete, the modified algorithm has the sequence new symbol - complete - predict with F. This modification results in a radical reduction in the number of items that must be constructed during parsing and thus overcomes a fundamental practical deficiency in the Earley algorithm. The parser finds all derivations for syntactically ambiguous input and handles the lexical homonymy that the sentence John saw the saw. illustrates.

Current work is focused on the extension of the parser to allow direct parsing with metarules. Just as with the ID/LP formalism, the first implementations of metarules expanded an external grammar into an internal form containing only context-free rules (cf Thompson 1982). This approach has now generally been rejected because of the impractical size of the expanded internal grammar. Investigations on the application of metarules have been closely connected with the search for constraints on their power (cf Gazdar and Pullum 1982: 26; Shieber, Stucky, Uszkoreit, and Robinson 1983; Stucky 1983) since expansion with recursive metarules applying to their own output would result in an infinite rule set in the internal grammar. The problem is to find algorithms (which use AI methods for problem solving, perhaps) that generate exactly those ID/LP rules actually required at parse-time.

In general it appears desirable to integrate AI methods with the algorithms from the theory of syntactic analysis of formal languages. This should not only increase the efficiency of the parser but also open the prospect of achieving cognitive plausibility (cf Pulman 1983).

Both syntactic and semantic considerations speak for an emphasis on verb-oriented parsing, which is not yet fully realized in the parser. In extensions of the parser, lexical retrieval may take place in a first pass in which all the verbs would be identified and noted; lexical subcategorization information in the verb entries could then help guide the actual parsing process. Such a lexical pass seems independently desirable in connection with components that are able to skim texts.

Once the crucial problem of metarules has been adequately dealt with, the parser will be extended to the entire GPSG formalism including that for

syntactic features; the treatment of slash categories and the various feature conventions belongs here. Additional mechanisms will be necessary for the construction of semantic representations as described above. The transducer will ultimately be modified so as to produce the internal grammar in an optimized normal form that preserves the semantics of the rules.

Preliminary work on implementation of the generator will be carried out parallel to these steps, which are, however, actually a prerequisite for the former.

5. Open linguistic problems

Considering its recency it can be no surprise that GPSG has been in a state of flux and continued development since its debut in the late 1970s. A good example of this is the integration of slash categories with the general treatment of syntactic features in Gazdar and Pullum (1982: ii, 35). A current problem of great theoretical interest is the analysis of so-called "crossed serial dependencies." Pullum (1983) argues that the latter can already be adequately accounted for in GPSG and that the formalism in fact requires further constraints, while Thompson (1983) proposes an extension of GPSG to handle the phenomenon. As the comparison of syntactic theories in Joshi (1983) shows, GPSG is one of a family of approaches using similar means and focused on essentially the same questions. Further changes in GPSG are unlikely to overturn its basic principles. While these theoretical issues are of great interest, workers in artificial intelligence cannot wait for a final codification of the theory before attempting to construct a functioning NL system based on GPSG.

A greater practical problem for the present system is the paucity of descriptive studies on German in the GPSG framework. Extremely promising treatments of German word order and other problems are given in Uszkoreit (1982, 1983) and Russell (1983). In view of the rapidly growing interest in GPSG a large number of studies on individual problems may be expected in the near future. Efforts to write a large GPSG for practical use in an AI system will necessarily make heavy use of the comprehensive formal description of German syntax in Zoeppritz (1984).

Of course, there are also many open questions in semantics. Linguistic work in this area will undoubtedly contribute to the treatment of categories like tense and case, while the representation of conceptual knowledge in artificial intelligence will provide the main basis for lexical representations.