Designing a parser for unrestricted text[1]

Fred Karlsson

(Published as Chapter 1 in Fred Karlsson, Atro Voutilainen, Juha Heikkilä and Arto Anttila, eds., *Constraint Grammar: A Language-Independent Framework for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin / New York 1995, 1-40)

1. Design considerations

By parsing is here meant the automatic assignment of morphological and syntactic structure (but not semantic interpretation) to written input texts of any length and complexity.

We shall advocate an approach called Constraint Grammar, abbreviated CG.[2] Constraint Grammar is a language-independent formalism for surface-oriented, morphology-based parsing of unrestricted text. Implementations exist for English, Swedish, and Finnish. Constraint Grammar parsing is based on disambiguation of morphological readings and syntactic labels. All relevant structure is assigned directly via lexicon, morphology, and simple mappings from morphology to syntax. The constraints discard as many alternatives as possible, the optimum being a fully disambiguated sentence with one syntactic label for each word, with the proviso that no genuine ambiguities should be obliterated.

Designing, implementing and testing a parsing formalism for unrestricted text is a complex and time-consuming task. This chapter presents in detail the goals, background assumptions, grammar design, computational tools and concrete experiences accumulated in the course of the enterprise. The details of the Constraint Grammar formalism are presented in Chapter 2.

The task of using the formalism for the construction of a full-scale Constraint Grammar is also a demanding one. The later chapters, written by Atro Voutilainen, Juha Heikkilä, and Arto Anttila, provide detailed examples and critical evaluations of how adequate the formalism turned out to be, in view of the daunting descriptive problems posed by the full-scale grammatical and textual variation of English.

The chronological order of events was not, and could not even have been, such that all of the Constraint Grammar formalism was constructed ”first” and ”then” applied to English. Rather, formalism development and its implementation, grammar construction, and the testing of all of these against a variety of texts, constitute an incremental process where all components feed each other in delicate and crucial ways.

1.1. General design goals

Here, we shall present 24 central goals, numbered (G-1) etc., that constitute the basic ”philosophy” of the Constraint Grammar approach.

(G-1) The goal of the parser and the grammar used in connection with it is to yield perfect analyses.

Every word-form token and larger structure should be assigned the appropriate morphological and syntactic analyses (categories, tags, codes, features). Appropriateness is judged in relation to the context at hand. For word-forms that are morphologically and syntactically unambiguous in context, there is only one appropriate analysis. In this situation, output is not perfect if several analyses are delivered even if the correct analysis is among the proposed ones. On the other hand,

if a word-form is morphologically and/or syntactically ambiguous (or, perhaps, vague) in context, its appropriate analyses are all the contextually viable ones. In principle, it counts as an error to enforce a unique interpretation on word-forms or larger structures that are truly ambiguous in context, i.e. the interpretations of which are not possible to resolve by structural means alone.

Perfect analysis thus means perfect recall in conjunction with perfect precision. Every instance of a missing, contextually appropriate analysis detracts from perfect recall. Every instance of a contextually inappropriate analysis, present in addition to the appropriate one(s), detracts from perfect precision.

It is methodologically important to postulate perfect analysis, in the sense just explicated, as the ultimate goal. Of course, in practice, a perfect analysis might be impossible to achieve in the domain of unrestricted text. One consequence of adopting (G-1) is that various types of heuristics are invoked only when no viable safe modes of analysis remain.

(G-2) The parser should assign some structure to every input.

This is another way of phrasing the requirement that the parser be capable of analysing unrestricted text, or that it be robust. No input should be entirely rejected. If (G-2) is taken literally, means should be available for parsing not just well-formed system sentences but also headlines and other non-sentential fragments, telegraphic communication, sublanguages of various types, dialects and colloquial forms, false starts, hesitations, misspellings, etc., all of which may occur in written language. In practice, compromises are mandatory. One reasonable approach is to establish a Master Lexicon, or Core Lexicon, catering for the central written vocabulary. This may be supplemented with additional text-type specific lexicons, depending upon special needs.

There will be no ultimate basis of normative judgment telling apart grammatical and ungrammatical sentences, the grammar accounting only for the former ones. It should also be possible to account for less than perfectly correct input.

On the other hand, it is clear that the distinction between ordinary, correct, frequently occurring sentences types and more or less esoteric specimens of language use must not be obliterated. If everything were to be allowed all the time on an equal footing, little could actually be done in order to achieve (G-1). There is a clear need for regimented methods of constraint relaxation or graceful degradation (e.g. Bobrow - Bates 1982; Marcus 1982; Granger 1982; Kroch - Hindle 1982; Hindle 1983; Jensen - Heidorn 1983). However, such considerations should be invoked only when all safer methods fail. In section 1.2 of this chapter, we shall argue that the proper way of doing this is to unite the grammar-based and heuristic approaches to parsing.

(G-3) The grammar formalism should be language-independent. It should also be clearly separated from the program code.

The formalism should have no bias in favour of some particular language type and it should, in a demonstrable way and without ad hoc changes to the formalism or the program code, be applicable to several languages belonging to different language families. If the grammar formalism is not clearly separated from the program code, language-independence is impossible to achieve.

(G-4) The grammar and lexicon used by the parser should be adjustable to text type.

Text type variation is considerable (Biber 1988), both in grammar and lexis. Furthermore there are hundreds of thousands of words in real texts that are not part even of the largest machine-readable lexicon, just consider all proper names. In view of this fact and of the goals (G-1, G-2), it seems reasonable to allow for an optional stage of text type adjustment or ”tuning” of the lexicon and of the grammar of the parser before a major run.

This might enhance the quality of parser output considerably, especially if the parser is to be applied to a longer homogeneous text. For example, imperatives and subjunctives cause a

lot of morphological ambiguities in English but these categories are practically non-existent in many genres. The parsing of such texts could benefit if all imperative and subjunctive readings were automatically discarded and if the corresponding disambiguation rules could be ”switched off”.

As for the lexicon, much text-type variation can be catered for by maintaining a Master Lexicon for the core vocabulary, and text-type specific lexicons for specialized domains. These can then be conjoined at run-time depending upon user needs. If a domain-specific lexicon is not available when a particular text is to be parsed, the parser should provide heuristics for the consistent morphological analysis of unknown words.

(G-5) Preprocessing of texts is an important step prior to parsing.

Even if it is important that the parser is robust, all minute text preprocessing details do not have to be catered for either by the grammar or by the parsing engine. Parsing is clearer overall if spell checking, paragraph boundary marking, case conversion, idiom recognition, mark-up of text segments, etc. are treated by separate preprocessing. Such matters are often language-dependent and somewhat idiosyncratic.

(G-6) The grammar formalism should be able to use typographical features such as punctuation and case of letters, and mark-up of input texts.

Active use should be made of every piece of information that might simplify parsing. Such useful information is conveyed by punctuation marks (also cf. Nunberg 1990), and upper/lower case of letters. For example, a sentence terminated by a question mark is likely to display certain word order properties. These could be good contextual clues for constraints that disambiguate English verb-forms. Words with an initial upper-case letter occurring in non-first position of a sentence are likely to be proper names. They might be heuristically labelled as such if they happen to be unknown to the morphological analyzer.

Real texts are higly variable. Far from all text segments consist of body text in prototypical sentential form. Standardization, e.g. the use of Standard Generalized Markup Language (SGML), makes available increasing amounts of text where text subsegments are tagged according to type, e.g. headings, captions, and footnotes. Such codes convey important information, e.g. headings normally lack finite verbs. It should be possible to use SGML-type codes in the context conditions of the parsing rules.

(G-7) Output should be readable and not flooded by an unmanageable wealth of alternatives.

This is especially relevant in regard to the representation of ambiguous output. If hundreds of theoretically possible readings are generated and printed for a single input sentence, there is no practical possibility of examining the output, at least not if the alternatives are not presented in some kind of preference order. Of course, (G-7) is not meant to imply that genuine ambiguities should be obliterated.

1.2. Grammar formalism design

(G-8) The kernel of the grammar statements are rule-like constraints but optional probabilistic features are available if the grammar statements fail.

Grammars which are used in parsers are often directly imported from autonomous grammar theory and descriptive practice that were not exercised for the explicit purpose of parsing. Parsers, or parser fragments, have been designed for English based on e.g. the Extended Standard

theory (PARSIFAL, cf. Marcus 1980; also cf. the Marcus-type parser Fidditch developed by Hindle 1983, 1989), Government and Binding Theory (Berwick - Weinberg 1984; also cf. Berwick - Abney - Tenny 1991, eds.), Generalized Phrase Structure Grammar (in a slightly modified form, as part of the Alvey Natural Language Tools (ANLT), cf. Briscoe et al. 1987; Phillips - Thompson 1987), and Tree Adjoining Grammars (Joshi 1985), to mention but a few. Such approaches to parsing are grammar-based and non-probabilistic. They normally have a strong linguistic-theoretical, especially syntactic, inclination. Often they contain grammar fragments rather than full-scale grammars, they are not extensively tested on authentic corpora, and they are not intended, or at least not fully implemented, to be applicable to unrestricted text. A notable exception is the GPSG parser of the ALVEY project which expressly is a general-purpose tool with wide coverage.

A different example of grammar-based parsing is offered by the PEG (PLNLP English Grammar) parser (Jensen 1986). Using Heidorn's specially tailored linguistic programming language PLNLP, PEG is not so theoretically committed as many other grammar-based approaches. Rather, PLNLP offers the linguist considerable latitude in postulating rules. PEG has been successfully applied to a large amount and a considerable variety of English text. Heuristics may be used in connection with or on top of PEG (Jensen - Binot 1987).

Hindle's (1983, 1989) parser Fidditch combines phrase structure grammar rules with a genuine concern for disambiguation and corpus application. Constraint Grammar resembles Fidditch in these respects.

Yet another strand of grammar-based parsing is exemplified by the Nijmegen TOSCA project (TOols for Syntactic Corpus Analysis, e.g. Aarts - van den Heuvel 1985; Oostdijk 1991). This approach emphasizes the need for basing the grammatical description on extensive corpus studies. It also stresses the importance of being able to run the parser on more or less unrestricted text. Constraint Grammar shares these concerns. The awareness of the importance of corpus studies for parser construction is rapidly growing. Hindle (1989) uses Fidditch in connection with pretagged corpora for inferring (new) disambiguation rules.

The Core Language Engine, CLE, is also an instance of grammar-based parsing, cf. the papers in Alshawi (1992, ed.). CLE incorporates a wide-coverage unification grammar for English syntax and semantics.

Probabilistic approaches to parsing, especially part of speech tagging, have recently been gaining ground. Notable advances were made in the 1980's by the CLAWS[3] system of the UCREL[4] group (Garside – Leech - Sampson 1987, eds.) and by Church (1988) with his PARTS program, both reporting a recall (success, correctness level) of roughly 96% in the task of supplying unrestricted text with unique part of speech labels. The basic method used for grammatical tagging is statistical optimization of the lexical and transitional probabilities of part of speech sequences. Sampson (e.g. 1987a) has been especially outspoken in his criticism of the potential of grammar-based (especially generative) approaches for analyzing unrestricted text. His claims have however been disputed by Briscoe (1990) and Oostdijk (1991).

Similar success levels, around 96%, have been reached by other researchers as well. DeRose's (1988) Volsunga algorithm, using transitional probabilities, is capable (in linear time) of disambiguating 96% of the part-of-speech ambiguities in running English text; the figure derives from an application to the Brown Corpus. Also cf. DeRose (1991), where alternative refinements of the method as well as an application to Greek are discussed.

A method based on a Markov-type probabilistic positional grammar has been developed by Bloemberg - Kesselheim (1988). The purpose of their model is to aid in selecting an optimal set of syntactic categories for describing and morphosyntactically disambiguating a given corpus. The model is based on pairs and triplets of successive codes. During the training phase, German and Dutch texts containing respectively 80,000 and 100,000 word-forms were used. However, Bloemberg and Kesselheim report no precise figures on the success of their model.

A special type of probabilistic approach to part of speech tagging is the neural network described by Nakamura et al. (1990). The network was trained on a corpus containing 1,024 test sentences, using both bigram and trigram modelling. The network is used as a

supplement for improving the performance of an English word recognition system. Precise data on the success level of part of speech tagging are not reported.

De Marcken (1990) reports on an improved version of DeRose's algorithm achieved by allowing the algorithm to return several categories for a word if the transitional probability is below an adjustable threshold. The algorithm has been extensively trained on and applied to the LancasterBOslo/Bergen Corpus of British English (LOB). An error rate of less than 0.1% (1 error per 1,340 words) is reached if the words are allowed to contain, on an average, 1.27 output readings, i.e. if roughly every fourth word is allowed to be two-ways ambiguous. The higher recall (less errors), as compared to CLAWS, PARTS, and VOLSUNGA, is traded for a substantial lowering of precision. More ambiguities are left in the output, precision decreasing to 73%.

De Marcken has also built a phrase structure parser, including a part of speech disambiguator, on top of the probabilistic system, but he gives no precise figures of the quality of this part of the system, when applied to unedited text (de Marcken 1990: 248).

Without belittling the impressive results achieved by probabilistic approaches, three critical remarks are called for. First, it is not obvious that stochastic algorithms could qualify as genuine language-independent formalisms. This seems to be true especially if categories of a particular language, such as English part of speech tags, or Finnish surface-syntactic functional tags, are part of the program code.

Second, the error rates of the probabilistic approaches seem to remain fairly high even in the domain of part of speech tagging, some 4% if substantial precision lowering is not allowed, thus detracting considerably from the perfection requirement (G-1). de Marcken (1990) gains higher recall by trading precision. Furthermore, there seem to be no easily accessible ways of diagnosing errors and trying to improve the performance of a large completed probabilistic system.

Finally, it has not yet been conclusively shown how successfully a primarily probabilistic approach would carry over into full-scale syntactic analysis of unedited text, including grammatical labelling. The original UCREL syntax module was somewhat tentative (Garside - F. Leech 1987). This line of research has later been directed towards a shallow notion of Skeleton Parsing, partly interleaved with manual intervention by the grammarian (Leech - Garside 1991: 16; Beale 1988).

Skeleton Parsing uses a combination of context-free phrase-structure rules and probabilistic estimation during the parsing process. Frequency considerations play an important role as source of information when the grammar rules are designed. This line of research has also been promoted in collaboration between the UCREL group and the Continuous Speech Recognition Group of the IBM T. J. Watson Research Center (cf. Black – Garside - Leech 1993, eds.). Promising results have been achieved especially in the domain of broad, ”skeletal” syntactic bracketing. Also cf. the experimental syntax of the Realistic Annealing Parser (Souter - O'Donoghue 1991: 44).

Another kind of probabilism is the notion ”preference”, suggested by Wilks (e.g. 1985) especially in regard to the resolution of attachment ambiguities. Preferences are normally semantic in nature and not based on n-gram analysis.

Is it possible to design a grammar-based approach which would, in the vein of (G-1), clearly raise the correctness rate (recall) as compared to probabilistic part of speech tagging? Could such an approach also extend successfully to surface syntactic analysis?

Judging from available literature, the theoretically inclined grammar-based approaches have not fully achieved both these aims. For most of these approaches, only scanty evaluation is available on their performance in the analysis of unrestricted text.

It seems worth exploring alternatives to making mainstream formal syntactic theories the kernel of the parsers. Constraint Grammar purports to be such an alternative. It has a linguistic kernel different in several ways from more theoretical grammar-based approaches. But Constraint Grammar also offers the option of using probabilistic tools on top of and in partial interaction with the linguistic constraints, i.e. when the latter ones fail.

Thus, the question of grammar-based versus probabilistic parsing need not be either–or. Constraint Grammar explicitly takes the stand that both constraints (= "grammar rules") and more probabilistic (= heuristic) statements are needed. For the purpose of constructing a successful parser, it is necessary to extract as much information as possible from grammatical restrictions, but also from probabilities computed across partially analyzed sentences or otherwise available. The stance of Constraint Grammar is expressly that as much useful parsing information as possible should be squeezed out of rule-type statements. Furthermore, probabilism should optimally enter the parser only on top of the linguistic kernel, and in a regimented fashion.

There are actually six possible types of relations between grammar-based and probabilistic approaches to parsing. All these hypothetical approaches do not seem to have been actively pursued:

A. grammar-based rules only, e.g. ALVEY Generalized Phrase Structure Grammar, Fidditch, TOSCA, The Core Language Engine;

B. probabilistic modules only (PARTS part of speech tagging, CLAWS part of speech tagging + UCREL syntax;

C. grammar-based rules strictly followed by probabilistic modules;

D. probabilistic modules strictly followed by grammar-based rules (the combination of PARTS and Fidditch; de Marcken's (1990) model for part of speech disambiguation and syntactic analysis trained on the Lancaster–Oslo/Bergen Corpus of British English);

E. grammar-based rules interleaved with probabilistic modules; grammar rules interleaved with heuristic metrics for solving ambiguities, followed by a fitting procedure for handling parsing failure: EPISTLE (Heidorn 1982; Jensen - Heidorn 1983; also cf. the papers in Jensen – Heidorn - Richardson 1993, eds.); McCord's (e.g. 1990) Slot Grammar; Skeleton Parsing (Black – Garside - Leech 1993, eds.); Realistic Annealing Parsing (Souter - O'Donoghue 1991); unification-based grammar rules supplanted with statistical information drawn from pretagged corpora (Briscoe - Carroll 1991); optimal linguistic constraints eventually followed by more heuristic constraints if the optimal constraints fail, then reapplication of the optimal constraints: Constraint Grammar;

F. probabilistic modules interleaved with grammar-based rules.

Constraint Grammar is here classified as type E because there can be (but need not necessarily be) some interleaving of constraints and more probabilistic (heuristic) mechanisms. But it must be stressed that the kernel of Constraint Grammar is meant to be linguistic rather than probabilistic in nature. Furthermore, if probabilism enters a Constraint Grammar at all, it does so in a highly linguistic fashion, i.e. as heuristic statements the outer shape of which is basically linguistic. Therefore the relative importance of probabilism is lesser in Constraint Grammar than in several of the other approaches of type E.

In actual practice, Constraint Grammar leaves the determination of the relation between purely linguistic constraints and more probabilistic statements to the discretion of the grammar writer. It is beneficial for the overall parsing success of a Constraint Grammar of a particular language if as many constraints as possible can be kept non-heuristic. On the other hand, nothing in the formalism prevents the grammarian from postulating crude constraints involving high error risks. This latitude of determining the grammar-heuristics relationship should be seen as a strength of the formalism.

There are several parsers that could be characterized as type E. The classical EPISTLE system has a similar overall setup, as does Briscoe - Carroll's (1991) generalised probabilistic LR parsing system based on the grammatical kernel of the ALVEY Toolkit. The disambiguation rules of Fidditch are related to stochastic disambiguation methods and therefore Fidditch could perhaps be classified as belonging to type E as well. Skeleton Parsing was already discussed above.

Construing the relation between grammar rules and probabilistic devices as in type E offers certain benefits. As implemented in Constraint Grammar, the step of graceful degradation is taken under controlled circumstances. Probabilism is resorted to only when the safe measures (= the optimal grammar-based constraints) fail, and only on the explicit request of the user.

Probabilistic mechanisms have better prospects of success if their domain is only the small amount of ambiguities remaining after the use of optimal grammar-based constraints. Furthermore, because the identity of the remaining unclear instances is known at application time, output can be marked in a suitable way and examined after the completion of the run. This regimentation isolates the effect of probabilistic mechanisms and makes it possible to observe them and to experiment with them. Due to this type of marking in the output, manual correction post hoc is easy, if needed.

(G-9) The foremost task of a parsing-oriented grammar is rather to aid in parsing every input than to define the notion ”grammatically correct sentence”, or to describe ”all and only the grammatical sentences”.

Jensen – Heidorn (1983: 93) have noted that ”.. trying to write a grammar to describe explicitly all and only the sentences of a natural language is about as practical as trying to find the Holy Grail”. We are in sympathy with the gist of this observation. Every grammar is bound to leak when confronted with real-world corpus variation.

The descriptive statements of the present formalism, the constraints, do not have the ordinary task of defining the notion ”correct sentence in L”. Rather, the constraints discard as many improper alternatives as possible. No input, be it in sentential form or not, will ever be fully rejected as ungrammatical. The constraints do not express strict grammaticality requirements as such. Rather, constraints capitalize on what the grammar writer knows to be grammatical, i.e. grammar rules, and especially on what can be (more or less safely) inferred from such rules.

For example, ordinary phrase structure rules for English NPs would state that a determiner is followed by certain optional premodifiers plus an obligatory noun. A consequence of this is that a determiner cannot be followed by a finite or infinitive verb form, without an intervening noun. In Constraint Grammar, a constraint could be postulated to this effect, discarding the verbal readings of the word-form bank in expressions like the bank, the old bank. Constraints thus express consequences of ordinary grammar rules and are in a kind of epiphenomenal relation to the rules. Elimination of clearly ungrammatical or otherwise improper (or improbable) structural configurations is a more modest aim than full enumeration of all and only the sentences that manifest grammatical configurations.

(G-10)Morphological and lexical analysis is the basic step of parsing.

Parsing is traditionally regarded as basically syntactic in nature. Early computational linguistics did not pay much attention to morphology. But successful parsing of unrestricted text is not possible without an appropriate machine-readable dictionary containing the core vocabulary and sufficiently rich descriptors for the individual lexemes. Furthermore, a comprehensive morphological analyser is needed that works in tandem with the lexicon. The analyser should have full lexical coverage, i.e. capability of properly analysing all inflectional phenomena and of doing base-form reduction. It should also have a precise mechanism for analysis of productive compounds not listed in the lexicon.

We use morphological analyzers designed according to Koskenniemi's (1983) two-

level model, TWOL. Such a computational morphology has proven theoretically sound, generalizable across languages, robust, and fast. Large lexicons containing at least tens of thousands of lexical entries are easy to implement using the two-level model. Of course, ”ease of implementation” does not mean that the laborious process of analyzing individual lexical entries, supplying them with the right descriptors, etc., would be simpler than before. The classical problems of lexicography remain.

The lexicon, in the sense of Master Lexicon, should be large enough, containing at least 30,000 lexemes, one per lemma (lexical entry) under the assumption that the morphological rules account for all non-lexicalized inflected, derived, and compound forms. The number 30,000 lexemes is a proper starting point for treating unrestricted text because the core vocabulary seems to be of roughly this size. For example, the intersection of Longman Dictionary of Contemporary English (LDOCE, ASCOT version) and the Oxford Advanced Learner's Dictionary contains some 30,000 entries (CELEX News 4/1988: 8).

When common names and some central text-type specific vocabulary are added, core vocabulary size can be roughly estimated at 50,000 lexemes. Presently, the ENGTWOL Master Lexicon contains 56,000 entries (Heikkilä, Chapter 4), the 1991 version of the SWETWOL Master Lexicon 48,000 entries (Karlsson 1992). Creating, maintaining, and updating such a lexicon, and interfacing it with morphology and syntax, is of course a major task.

A 50,000-entry core lexicon with additional full inflectional capability compares reasonably well to what is documented in the literature. The 60,000-entry Longman Dictionary of Contemporary English (LDOCE) has served as a platform or aid for several lexicons, e.g. Akkerman et al. (1988; the ASCOT project), and the ALVEY morphological analyzer (Pulman et al. 1988; Carroll – Grover 1989; Boguraev – Briscoe 1987). The early EPISTLE project used a 130,000 word dictionary (Jensen – Heidorn 1983). Its successor CRITIQUE has a lexicon in excess of 100,000 entries (Richardson – Braden – Harder 1988). Later, the Lexical Systems project at IBM has built on this work by using the UDICT module (63,000 lemmas) and amending it with a word-form list containing 250,000 items (Byrd et al. 1986). McCord's (e.g. 1990) Slot Grammar also uses the UDICT lexicon. The TOSCA project uses a word-form list containing 70,000 items (Oostdijk 1991: 154), Hindle's (1989: 119) Fidditch 100,000 word-forms. The CLAWS lexicon is smaller. Originally, it contained 8,000 items but has later been extended to 26,500 entries (Beale 1988).

(G-11) The central task of the parser is to resolve ambiguities.

There is a growing awareness that ambiguity is one of the crucial problems of NLP. This has been emphatically stated e.g. by Gazdar – Mellish (1989: 7) according to whom this is the single most important NLP problem. The basic idea of Constraint Grammar is to bring the description of ambiguities to the fore. Constraint Grammar basically is a formalism for writing disambiguation rules (cf. section 2 below and Chapter 2).

(G-12) Break up the parsing grammar, and the whole problem of parsing, into three modules: morphological disambiguation, assignment of intrasentential clause boundaries, and assignment of surface-syntactic labels (syntactic analysis proper).

Viewing the grammar of the parser in turn from one or the other of these three angles has clarified many intricacies and has also made each module take manageable proportions.

The modules are clearly interrelated. Thus, for optimal morphological disambiguation of e.g. English part of speech ambiguities it would be useful to know the left and right boundaries of the current clause, if it occurs in a complex construction where clause and sentence boundaries do not coincide. It would also be beneficial to know as much as possible about the syntactic structure of the clause. Everything that is known decreases the range of morphological interpretations, i.e. helps in reducing the set of morphological ambiguities.

Two examples: (i) An ambiguous word-form w, with ”finite verb” as one of its

morphological interpretations, cannot be a finite verb if there is a finite verb elsewhere in the same simplex non-coordinated clause. (ii) The end boundary of clause c must be before word-form w, if w unambiguously is a finite verb and an unambiguous finite verb occurs immediately to the left of w in c. In this configuration, w must be the first word of clause c+1.

The totality of morphological disambiguation, clause boundary assignment, and syntactic analysis constitutes a formidable problem when projected onto the variability of running text. Therefore we approach the modules one by one. This offers the additional benefit that one can start experimenting with small Constraint Grammars on real texts almost immediately, e.g. without first having to design a full syntactic description.

(G-13) Extend the principle of ambiguity resolution to syntactic analysis proper (assignment of syntactic labels), and to clause boundary assignment.

The second basic tenet of Constraint Grammar, in addition to viewing disambiguation as the primary problem of parsing, is to apply the principle of disambiguation to all three modules: morphological disambiguation, clause boundary determination, and syntactic analysis. Constraint Grammar thus is a fairly systematic and unified framework.

Principles (G-11, G-13) taken together mean that Constraint Grammar parsing is reductionistic in nature. The starting point is all possible analyses which is the worst situation. The task of the individual constraints is to discard as many alternatives as possible. A successful parse eliminates all ambiguities, making the sentence morphologically and syntactically unambiguous. An unsuccessful parse leaves the input sentence untouched, save for morphological analysis and enumeration of the syntactic functions each morphological reading of each word may have, granting at least that analysis. Optional heuristics may be invoked. A completely unsuccessful parse, a real disaster, would eliminate all proper readings and syntactic functions, leaving only improper ones. This could happen only if the constraints are very poorly formulated.

(G-14) The purpose of syntactic description is to assign to each word a label indicating its surface syntactic function. The labels also indicate basic dependency relations within clause and sentence. Syntactic output is a flat representation.

Constraint Grammar syntax is in many ways close to traditional syntax but spiced with a richer set of syntactic labels and a more explicit treatment of dependency relations.

The compilation of constraints capitalizes heavily on basic-level descriptive grammatical data. However, there are no suitable ready-made grammars on the shelf that one could take as such and start incorporating into a parser. Good descriptive grammars provide foundational facts but these have to be interpreted and hand-coded according to the grammar formalism and frequently modified in view of the results obtained from extensive corpus analysis and test runs.

1.3. Parser design

(G-15) The parser should provide good tools for testing, debugging and optimizing individual constraints as well as the whole grammar.

Toy grammars are easy to manage but when a large grammar containing hundreds or thousands of rules is applied to a variety of sentences, not to speak of running text, dependable and lucid facilities for testing and management of the grammar and its behaviour in the parsing environment become crucially important.

There are several situations where such tools are called for. When errors occur in the output and the grammar is to be corrected, it should be possible to reparse such sentences and examine which constraints were activated and which were sources of the errors. When grammars

become large, it should be possible to find out, by various types of statistics, which constraints are in frequent use and which not. When a large grammar is compiled and tested, it frequently happens that some early constraints can be eliminated because they are subsumed by more general constraints that were included in the grammar at a later point in time. Such situations can only be uncovered by appropriate statistics.

In general, constraint interaction is a major problem in the compilation of a full-scale grammar. The larger the grammar grows, the more important becomes the proper management of constraint interactions. This is especially true when perfect analysis is strived for. In limited domains, NLP systems may occasionally reach success levels of 95B97%. But of course the remaining errors or open questions are the hardest ones to solve. It is precisely in this situation, when the grammar is large and a number of difficult problems remain, that the need for testing and debugging tools is greatest. This is also the situation where the need for the grammar writer's control over the grammar and the parsing environment is the greatest. This is the foremost reason why the Constraint Grammar approach to parsing tries to defer the use of heuristics as long as possible, and to retain as much control as possible over both constraints and heuristics. This is one aspect of the following goal:

(G-16) The user should have a lot of optional control over the parser.

On a general level, there is just one goal for a parser: to assign appropriate structure to input texts. In practice, however, (other than toy) parsing is done in different contexts and for different purposes. Partial grammars and errors are more tolerable for some purposes than others. For some application, it could be beneficial to require every word and sentence to have some unique analysis, at the expense of a growing error rate. For another application it could be good to present all the ambiguities that are impossible to solve safely to the user for manual disambiguation on the spot. A third alternative is to deliver such ambiguities as output. Statistics on constraint application are not needed for every application.

Sometimes there is time to do a more time-consuming optimal analysis of a long text. At other times there might be a hurry to obtain less than perfect results rapidly. Heuristics may be used at several points in the Constraint Grammar parsing process, such as for morphological analysis (of unknown word-forms), for morphological disambiguation (e.g. of remaining part of speech ambiguities), and for syntactic disambiguation, i.e. for determination of the appropriate syntactic label. Furthermore, the user might want to use none, one, or several of these options in any combination. Output format requirements might vary. More experimentation and debugging tools are needed during constraint testing and grammar compilation than during ”production runs”. In short, the parsing environment should be powerful and flexible.

(G-17) The interaction of grammar and parser should be such that it is possible to let the evolving analysis of the current text affect parser decisions, i.e. to let the parser be adaptable and to ”learn” from the text at hand.

We are not talking of automatic acquisition of rules or constraints here (cf. Hindle 1989), but rather of making it possible to decide unclear cases of disambiguation on the basis of previously determined clear cases where the same content words were involved.

(G-18) The parser should be fast.

This is an obvious desideratum. A parser spending hours or even several minutes (or one minute) on a sentence is not practical. Especially troublesome is the exponential behaviour of many parsers which tends to be encountered more and more frequently as the number of rules grows.

Koskenniemi's implementation in the C programming language of the two-level model for morphological analysis of word-forms, using the ENGTWOL lexicon and morphological description, analyzes running text at a speed of roughly 600 words per second,

more than 1 million words per hour, when running on a Sun SPARCstation 2 (Koskenniemi, personal communication).

Morphosyntactic parsing standing on top of such an analyzer will of course be slower. The Lisp version of the Constraint Grammar Parser (CGP) presently performs the heaviest type of full morphosyntactic analysis of English, including preprocessing and morphological analysis, and applying all morphological and syntactic disambiguation constraints, at 3–5 words per second on a Sun SPARCstation 2. Plain morphological disambiguation of parts of speech and minor morphological features is done at a rate of 8–10 words per second. This version is a development version apt for stepwise compilation and testing of a constraint grammar for a particular language. It offers the full environment and all the options mentioned above (and more fully described in Chapter 2). Given the purpose, the speed is bearable in real testing situations.

There also are production versions of the Constraint Grammar Parser available, programmed in C and C++. These have the same kernel functionality but not all the environmental features. The fastest production version, programmed by Pasi Tapanainen in C, does full syntactic analysis at a speed of 400-500 words per second on a Sun SPARCstation 10/30.

(G-19) Optimally, the parser should be implemented in the finite-state framework.

We say optimally, because the Lisp, C, and C++ implementations of the Constraint Grammar Parser are all fairly ad hoc rule interpreters without any deeper technical contributions to parsing theory. These implementations are not based on the use of well understood and theoretically sound parsing algorithms. Rather, they could be characterized as situation-action parsing programs. They have two principal positive computational properties. First, the modules for reading and applying constraints are dissociated from the grammars as such. Thereby grammar writing is separated from parser implementation and application. Second, the rule (constraint) application components in the programs have been successfully optimized as witnessed by the fact that processing speeds of several hundred words per second have been achieved.

There are several well-known benefits of the finite-state frame-work, such as simplicity, well understood properties, and rapidity. Koskenniemi (e.g. 1990) is exploring the general feasibility of finite-state syntax in a linguistic framework that resembles that of Constraint Grammar; also cf. Koskenniemi – Tapanainen – Voutilainen (1992) and Voutilainen – Tapanainen (1993).

1.4. Compiling a Constraint Grammar

(G-20) Constraints are derived partly from (the consequences of) basic descriptions in extant large grammars, partly from study of tagged or untagged corpora.

(G-20) adheres to ordinary practice in fundamental descriptive grammar. Knowledge of basic grammatical regularities forms the basis of any attempt to construct a Constraint Grammar. But this has to be amended with systematic use of corpora, especially tagged and disambiguated ones from which many additional regularities can be inferred. Constraint Grammar shares many of the concerns of recent corpus linguistics, cf. Aijmer – Altenberg (1991, eds.), Svartvik (1992, ed.).

(G-21) Incremental addition of new constraints to the grammar should be easy.

Unexpected rule interactions frequently arise during the compilation of large computational grammars. In connection with (G-15, G-16), the need for flexible debugging and diagnosis tools was already stressed. More generally, the formalism should be such that individual constraints are unordered and maximally independent of each other. These properties facilitate incremental

adding of new constraints to the grammar, especially to a large one. By this we mean that new constraints should have predictable and easily observable effects. These principles get more important as the grammar grows.

(G-22) The kernel of the whole Constraint Grammar Parsing system is the lexicon and the morphological analyzer. The lexicon should be compiled with special care.

Appropriate lexical and morphological description is so important for successful Constraint Grammar parsing that it is separately discussed in several chapters of this book.

(G-23) The grammar formalism should be bidirectional (reversible).

It would be optimal if the formalism were general and abstract enough for it, or rather for grammars written in the framework of it, to be used both for sentence analysis and sentence generation. Constraint Grammar takes no explicit stand on this issue. In practice, however, constraints are geared towards parsing, and Constraint Grammars are analysis grammars. It remains to be demonstrated that full-scale syntax can be done by one and the same reversible grammar.

(G-24) The parser and the grammar should be made accessible to the research community.

This is a good way of spotting problems and shortcomings and it could be practically useful e.g. for tagging purposes. It could even contribute to the development of parsing theory and practice. As stated in the Preface of this book, interested readers may test the English Constraint Grammar Parser by sending short texts containing maximally 300 words to the appropriate e-mail address.

2. Ambiguity as a parsing problem

2.1. Types of ambiguity

In everyday parlance, the notion ”ambiguity” means either some kind of communication disturbance, or, more specifically, that certain sentences may be interpreted in more than one way, or that they have more than one meaning. Real communicative ambiguities arise if there is not enough information available for the (purported) optimal and/or intended reading to be retrieved.

Awareness of the existence of ambiguities is as old as the study of language. Kooij (1971: 2) points out that Aristotle presented a taxonomy of ambiguities, distinguishing i.a. between lexical ambiguity, constructional homonymy, wrong combination of elements, and wrong division of elements.

There are many types of ambiguity. First, we make a basic tripartition between structural ambiguities, also called grammatical ambiguities, meaning ambiguities, and pragmatic ambiguities. Constraint Grammar in its present form is actively concerned with structural ambiguities only.

There are several types of meaning ambiguities. Their most pervasive variant is polysemy, often called lexical ambiguity (cf. Crystal 1991, s.v. ambiguity). Polysemy and disambiguation of it have been treated from many angles in the literature, cf. Kelly – Stone (1975), Hirst (1987), Cottrell (1989), Ravin (1990) for some expressly computational approaches to resolution of polysemy. Many scope ambiguities are semantic, especially those pertaining to quantifiers or negation (Hurum 1988). Ambiguities exist also in the domain of thematic roles, in the sense of case grammar (Bear – Hobbs 1988). Idioms offer special problems because there often is an ambiguity between the idiomatic and the compositional meaning of a string of words, cf. kick the bucket (van der Linden – Kraaij 1990). Disambiguation of cue phrases, metatextual

elements such as now, well have been treated by Litman – Hirschberg (1990). Such elements are partly pragmatic in nature.

Resolution of pragmatic ambiguities requires knowledge of the spatiotemporal context or of speaker intentions. Such ambiguities are e.g. speech act ambiguities as discussed by Hinkelman – Allen (1989). An utterance like:

Can you pass the salt?

is an indirect request, or a question concerning ability. Another frequent pragmatic ambiguity is pronominal reference (Jensen – Binot 1988, Rich – LuperFoy 1988, Dagan – Itai 1990). Semantic and pragmatic ambiguities will not be further pursued in this book.

The resolution of meaning ambiguity has been a classical theme in psycholinguistics. Cf. Altmann (1988) and Gorfein (1989, ed.) for a survey of relevant problems and the state of the art.

Henceforth, the term ”ambiguity” denotes structural ambiguities only. In terms of domain or extension, two types of structural ambiguities can be distinguished. Local ambiguities normally concern one or a few adjacent words, i.e. they have a fairly short span. Local ambiguities may or may not be possible to resolve, given access to a sufficiently large portion of the sentence. Global ambiguities concern the whole sentence and are by definition not possible to resolve without extrasentential knowledge.

The borderline between local and global ambiguity is not always clear. Given the definitions, the ambiguity starting at the word that would be local in the sentence below. Yet, such a large portion of the sentence is involved that the ambiguity resembles true global ambiguities.

She saw that gasoline can explode.

The most frequent type of local ambiguity resides in one word-form. There are received terms in use for the phenomenon of structural ambiguity of individual items, homonymy regardless of whether the medium is spoken or written, homophony or homography depending upon medium.

Both free and bound morphemes may be homonymous. A classical example of free morpheme homonymy is Engl. $PORT_1$ `harbour' vs. $PORT_2$ `brand of wine', where the word-form port instantiates two different lexemes or paradigms. As for bound morphemes, Lat. 1st declension ablative plural -is vs. dative plural -is provides a well-known example.

Structural ambiguity of bound morphemes is normally treated in relation to the respective word-forms. Thus, terris is ambiguous between the ablative plural and the dative plural interpretations. An ambiguous item such as port or terris has at least two readings (interpretations). The term morphological ambiguity will be used to denote all ambiguities the domain of which is one word-form.

Lat. terris is an instance of paradigm-internal morphological ambiguity. The readings populate distinct slots in the inflexional matrix of the same lexeme. $PORT_1$ and $PORT_2$ exemplify paradigm-external ambiguity. Here, the readings belong to (inflexional matrices of) different lexemes. Paradigm-external ambiguities may concern base forms only ($PORT_1$, $PORT_2$). They may also concern inflexional forms only, e.g. the Swedish word-form anden which is the nominative singular definite reading of either AND `[wild] duck' or ANDE `ghost, spirit'.

Paradigm-external ambiguities may also involve any combination of ambiguities between base-forms and inflexional forms. The Finnish word-form alusta is paradigm-internally three different inflexional forms of the verb ALUSTA+A `start [e.g. a discussion]', viz. the 2nd person singular imperative, the negative present indicative, and the stem form of the negated 2nd person singular imperative. In addition, alusta is paradigm-externally i.a. the elative singular case form of the noun ALKU `beginning', the partitive singular case form of the noun ALUS `boat', and the nominative singular case of the noun ALUSTA `base'.

It is not evident in all situations how much ambiguity should be postulated.

Descriptive problems arise especially when both terms of some morphological opposition lack material realization in some word-form. Is e.g. the Swedish word-form hus `house' or `houses' ambiguous between a singular and a plural reading, or vague in regard to number, or underspecified? Such questions can be answered only in relation to an overall morphological description and the aims towards which it is directed.

Morphological ambiguities may concern any combination of major and/or minor morphological categories. In English, categorial ambiguity (of parts of speech) is pervasive and is one of the most serious problems facing anybody trying to construct a realistic and succesful English parser. Categorial ambiguities (especially N/V) are problematic because they tend to confound the basic structure of the sentence if unresolved. In morphologically more elaborate languages such as Finnish there is an unpredictable wealth of accidental morphological ambiguities invoking minor morphological features, often across parts of speech.

Local ambiguities may also extend over a few adjacent words. This is true of attachment ambiguities of which there are several types. Hirst (1987, Chapter 6) offers an overview of English attachment problems, also cf. Church – Patil (1982), Bear – Hobbs (1988). Normally attachment ambiguity is a problem of modifier placement.

A prepositional phrase may e.g. have either a preceding verb or a noun as head, or one of two or three preceding nouns. Relative clauses do not always modify the immediately preceding noun but an earlier one. A final adverbial may belong to the current subordinate clause or to its matrix clause. Reduced gerundive clauses may have multiple attachment as in example (a) below. An adverbial at the border of two clauses may belong to either clause. Nominal compounds may have a multilayered internal structure such as example (b) (Hirst 1987: 144). Many attachments are truly ambiguous in terms of syntactic structure, cf. the classical laboratory example (c). If possible at all, the disambiguation of such instances presupposes contextual or fully extralinguistic knowledge.

(a) I saw the Grand Canyon, flying to New York.
(b) airport long term car park courtesy vehicle pickup point
(c) I saw the man on the hill with the telescope.

From Hirst (1987: 137B150) we borrow the class of analytic ambiguities. These occur when there is more than one possible analysis of a subpart, either word or longer constituent, of a sentence. There are several typical English instances. A preposition may be a verb particle or a true preposition starting a prepositional phrase as in example (d) below. Present participle readings and noun readings may be ambiguous (e). The end of a noun group may be unclear in a sequence of nouns (f). Reduced relative clauses and finite verb structures look alike (g). Two syntactic labels might be equally possible (h). Most of these ambiguities are local but some have a global flavour, especially (f) that invokes coordination and gapping.

(d) A good pharmacist dispenses with accuracy.
(e) We discussed running.
(f) Bill gave the dog water and Sue the cat food.
(g) The horse raced past the barn fell.
(h) They are flying planes.

True global ambiguities range over most of the sentence, as in classical laboratory sentences like:

(i) Time flies like an arrow.

In actual usage such instances seem to be rare, perhaps save for intentional puns. A fairly global type of ambiguity is that connected with gap finding and filling, e.g.:

(j) Those are the boys that the police debated / about fighting /.

where the slash ”/” indicates a potential gap site (cf. Ford – Bresnan – Kaplan 1982).

In Constraint Grammar, ambiguities are conceived of as strict surface phenomena. A phrase such as the shooting of the hunters is not surface ambiguous, it consists of a plain nominal head followed by a plain nominal postmodifier. The possible ambiguity must be recognized at some ”deeper” level of syntax or semantics. However, it is not clear how the borderline between surface morphosyntax and such a deeper level of analysis should be conceived.

A particularly recalcitrant type of ambiguity arises in connection with coordination, which, in turn, often co-occurs with various types of ellipses. Do syntagms like old men and women or fuel pump and filter have two surface syntactic analyses, thus being ambiguous and posing a challenge for Constraint Grammar parsing, or do they have only one syntactic, but possibly two semantic analyses? So much basic descriptive data seem to be lacking that this issue and similar ones must be left undecided for the time being. The individual grammar writer will have to make his/her own decisions.

2.2. The magnitude of the ambiguity problem

DeRose (1988) analyzed the 1 million word Brown corpus and concluded that the rate of categorial ambiguity is around 11.0% for word form types, and over 40% for word form tokens. This figure covers paradigm-external part of speech ambiguities only. In more heavily inflected languages there may be additional extensive paradigm-internal ambiguities, such as for Finnish verbs the finite 3rd person plural present indicative vs. the nominative plural present participle (harjoittele+vat `they exercise' vs. harjoittele+va+t `exercising + participle + nom. pl.').

In Swedish, morphological ambiguities are more pervasive than in English. More than 60% of the word-form tokens are at least two-ways ambiguous, as determined on the basis of the 1 million word corpus Press-65 (Allén 1971, Berg 1978). Herz – Rimon (1990) report that the word-form ambiguity rate of Hebrew is nearly 60%.

The comparable Finnish figures are considerably smaller. According to Niemikorpi (1979), whose study is based on roughly 80,000 word-form tokens from the radio genre of the 429,058 word-form Oulu corpus, some 3.3% of the word-form types, and 11.2% of the word-form tokens, are ambiguous. It seems a reasonable hypothesis that there would be an inverse correlation between number of distinct word-forms and rate of ambiguity: the more word-forms there are, the less ambiguous they tend to be. This topic cannot be further pursued in the present framework.

Figures indicating amount of structural ambiguity should be interpreted with due care because they are in many ways a product of description. Such figures are not properly comparable across languages unless the descriptions have been duly standardized. No attachment ambiguities or more global ambiguities were included in the above figures. Thus, they do not inform about the ”total” rate of structural ambiguity.

2.3. Different approaches to the disambiguation problem

Considerable experience of large-scale disambiguation of text corpora accumulated already during the 1960's and 1970's. Greene – Rubin's (1971) TAGGIT algorithm performed successful part of speech tagging on some 77% of the one million words in the Brown Corpus.

Several methods have been proposed and applied for the purpose of structural disambiguation. Here, a brief overview will be given, with special reference to part of speech disambiguation which is the basic application area and the most important prerequisite for successful parsing of running text.

(i) Manual intervention. This is used as one of the main disambiguation methods e.g. in the TOSCA project (Oostdijk 1991: 156B). Also cf. Tomita (1984).

(ii) Statistical optimization based on transition probabilities. Several approaches of this type were mentioned in the discussion concerning grammar-based and probabilistic approaches to parsing (G-8). Some more approaches of this type are worthy of mention. Binot (1987) has developed a part of speech disambiguation preprocessor for French using weighted heuristic rules and local contexts (a few surrounding words). Newman (1988) proposes a single global optimization procedure for all ambiguities in a sentence.

(iii) Unification is an excellent tool for structural disambiguation, actually removing the burden of disambiguation from grammar to parser. Unification means monotonic combination of consistent pieces of information. Non-consistent pieces are discarded as a consequence of the unification process. The ALVEY parser is representative of this type (Carroll et al. 1988). Unification-based parsing utilizes grammars in the strict sense. It is somewhat unclear how suitable large unification-based grammars are for massive application to running text, cf. Briscoe – Carroll (1991) who analyze the problems and present i.a. a mixed method invoking manual intervention.

(iv) Lambek calculus in conjunction with categorial grammar descriptions is reported to be used in the Dutch CELEX project for disambiguating a 45,000,000 word token and 800,000 word type Dutch corpus (van der Woulen – Heyden 1988, also cf. CELEX News, April 1988). Not all ambiguities are possible to resolve in this way, however. The residue is supposed to be handled by proof-theoretical methods (the Gentzen Proof Machine). This set-up has not yet been demonstrated to work in a robust fashion.

(v) Utilization of independently existing grammar rules for doing compatibility checks on the local environment. In the framework of deterministic Marcus-type parsing, Milne (1986) used the rules of the parser in conjunction with the surface consequences of restrictions like number agreement and word order for part of speech disambiguation.

(vi) Special disambiguation rules. Such devices have been proposed several times in the literature. Marcus (1980) used ”diagnostic rules” for part of speech disambiguation, with the power of inspecting the contents of Parsifal's buffers. Hindle (1983, 1989) uses 350 disambiguation rules, constituting half of the rules of Fidditch. Shieber (1983) postulated disambiguation rules for solving attachment ambiguities in connection with a shift-reduce parser. Herz – Rimon (1990) use local constraints disallowing certain sequences of tags from being in the near context of a target word with designated features. The local constraints are retrieved from a context-free grammar of the language.

(vii) Cost-based ambiguity resolution has been proposed by Kitano – Tomabechi – Levin (1989). The cost of a particular parsing hypothesis is defined as the workload required to take the path representing the hypothesis. Details are lacking on the efficiency of this approach in the domain of structural disambiguation.

(viii) A knowledge-based approach to structural disambiguation, especially attachment ambiguities, has been proposed by Nagao (1990), using preferential evaluation of dependency structures stored in a knowledge base. The knowledge base is constructed semiautomatically from dictionary definitions. PEG (PLNLP English Grammar) output is used as input. The system is reported to be experimental. Also cf. Jensen – Binot (1987).

(ix) Ambiguities have been represented in terms of shared structure in connection with context-free chart parsing (Billot – Lang 1989). This is a convenient solution to (G-7), i.e. output should not be flooded with a wealth of alternatives, but it is not a disambiguation method.

As is obvious from the survey, the disambiguation problem has attracted considerable attention. One of the earliest comprehensive treatments of general disambiguation is Boguraev (1979). Another seminal contribution is that by Hirst (1987) who deals with both semantic and structural disambiguation in depth. Hindle's (1989) disambiguation and parsing system has been applied to considerable amounts of text (millions of words).

Constraint Grammar belongs to type (vi) in the above taxonomy and bears functional resemblance to the other type (vi) models. The main difference is that not only morphological disambiguation but also the whole syntactic description, and all of the parsing process, is conducted in the framework of one and the same rule type, the constraints.

3. The notion ”constraint”

The most important property of constraints, as conceived in Constraint Grammar, is that they disambiguate. By disambiguation is here meant a reduction of existing morphological and syntactic ambiguities.[5]

A typical morphological disambiguation constraint could state that readings containing verbal features such as PRES, PAST, IMP, SUBJUNCTIVE are to be discarded if occurring after an article. When applied to a syntagm like the bear, this constraint eliminates all verbal readings of the word-form bear, i.e. the forms of the verb lexeme BEAR, thus disambiguating the phrase morphologically because only the nominal reading of the word-form bear would remain.

English nouns may have about a dozen different surface syntactic functions such as subject, object, predicate complement of the subject, prenominal modifier, or prepositional complement. Prior to syntactic analysis, Constraint Grammar takes every noun to display all of these functions. Note the extension of the term ”(syntactic) ambiguity” here. Constraint Grammar could claim that, prior to analysis, the word John is syntactically twelve-ways ambiguous every time it occurs, e.g. in the sentence John runs. A typical syntactic constraint could state that the syntactic function of a noun is subject if it is followed by an active finite verb form, and no nouns intervene. This constraint syntactically disambiguates John in the example. One syntactic label is picked, the others are discarded.

Optimal disambiguation results in just one genuine alternative remaining, either morphological reading or surface syntactic function label. But disambiguation may also be partially successful so that one or more alternatives are deemed inappropriate but more than one still remain.

The constraints will mostly be on a more delicate level, more concrete, closer to real words and sentences, than ordinary grammar rules tend to be in much of current syntactic theory. Most certainly there will be no single general or purportedly simple rule stating all aspects of the syntax of, say, grammatical subjects or noun phrases in language L.

Rather, there will be a few dozens of down-to-earth constraints that state bits and pieces of the phenomenon, approaching or even reaching the correct description by ruling out what cannot be the case in this or that context. Each single statement is true when examined in isolation, either absolutely or with some degree of certainty, depending upon how careful the grammar writer has been. Furthermore, disregarding morphosyntactic mappings, the constraints are unordered.

Thus, Voutilainen postulates some 35 constraints for the morphological disambiguation of the English word-form that which can be a determiner, a demonstrative pronoun, a relative pronoun, a subjunction, or an intensifier. Anttila uses some 30 syntactic constraints for the disambiguation of the English grammatical subject.

Constraint Grammars are not maximally compact. It is not a mandatory goal to express the syntactic regularities of a language in the form of very general and few rules. On the other hand, due to their concreteness, constraints are easy to manage and their effects are clearly discernible and traceable in all circumstances, also B and in particular B when the grammar grows large. Furthermore, due to the mutual independence and the absence of ordering of individual constraints, constraint grammars do not show exponential behaviour at parse time. And finally it is easy to start experimenting with just one or a few constraints on running text because every constraint is self-contained regardless of what other constraints there might be in the grammar.

The ensemble of constraints postulated for language L is a grammar of L, a model of L as a projection of Constraint Grammar theory. It is a ”good grammar” (for parsing purposes) if it can be used to parse running text with reasonable success.

4. Disambiguation strategies

Most approaches to parsing have three sequential steps: (i) preprocessing, (ii) morphological analysis, (iii) syntactic parsing proper, normally including morphological disambiguation (if any). The general set-up of Constraint Grammar parsing makes no exception. Because the task of morphological and syntactic disambiguation during step (iii) is a complex one, every effort should be made during the first two steps to pave the way for the parser.

The notion ”disambiguation strategy” does not refer to Constraint Grammar constraints, nor to their use during actual parsing. Rather, they are a set of methodological maxims guiding the construction of the preprocessor, the Master Lexicon, and the morphological analyzer. (S-1) is an important overall strategy, a common denominator for several of the subsequent more specific strategies.

(S-1) Disambiguate as soon as possible.

Some disambiguation can be done already during preprocessing. Consider the English complex preposition in spite of. It makes little sense to analyze the three parts of it as if they were normal words. This would create at least two readings for in, PREP and ADV, neither of which is correct, because the three words together should get the unique part of speech code PREP. In order to avoid spurious processing, in spite of should be recognized as a complex item during preprocessing. It should also be properly entered in the lexicon as one entity. Thus:

(S-2) Start disambiguating in the preprocessing module.

The general strategy (S-1) is important also in Master Lexicon design. Many spurious ambiguities can be avoided if this is carefully compiled. Therefore:

(S-3) Optimize the Master Lexicon.

The morphological information in the lexicon should be as precise as possible, especially the morphotactic information guiding compound formation in languages such as Finnish, German, and Swedish. A sloppy lexicon and morphology might cause excessive overgeneration in the domain of productive compound formation. On the other hand, a stringent lexicon speeds up disambiguation by minimizing the amount of spurious readings (Karlsson 1992).

Consider the possible automatic analyses of the Swedish word-form frukosten `the breakfast'. An unoptimized Swedish lexicon would contain i.a. the basic lexemes FRUKOST `breakfast', FRU `wife', KO `cow', KOST `nutrition', STEN `stone', EN `juniper'. If all possible readings were assigned to the word-form frukosten by this lexicon and a morphological analyzer accounting for compounds by loosely formulated productive rules, the following readings would be generated. Note that word-final -en also is the definite ending of non-neuter nouns, the understroke ”_” indicates compound boundary, and the character ”+” inflectional boundary:

(a)	frukost+en	`the breakfast'
(b)	frukost_en	`breakfast juniper'
(c)	fru_kost_en	`wife nutrition juniper'
(d)	fru_kost+en	`the wife nutrition'
(e)	fru_ko_sten	`wife cow stone'

In order to avoid excessive overgeneration of spurious compound readings, especially of type (c), the noun EN `juniper' must not participate freely in the productive process of compound formation. Consequently compounds containing the noun EN as non-first element must be listed in the lexicon. See Karlsson (1992) for more examples, including statistics, of how the lexicon can be optimized.

As a rule of thumb, segmentally short items are risky in compound processes, especially if they are frequent. Another relevant Swedish example is the string or which is a common plural ending for nouns but also a rare zoological term the meaning of which is `mite'. The latter possibility must not be offered as a productive alternative for compound formation. Other similar ”dangerous” Swedish words are e.g. AL `alder', AR `are', HET `hot', IS `ice', REN `clean', Å `river', Ö `island'.

Another way of eliminating spurious compound readings is to minimize the effect of homonymy of first parts of compounds. For example, the Swedish noun lexemes LAG `law' and LAG `team' both form compounds but the doubling of compound interpretations may be avoided by letting only one of the LAG-words recurse to the main lexicon.

Of course, this is an artificial solution that is defensible only from a purely technical point of view. The most important subcase of this type concerns the pairwise homonymy between noun and verb stems, noun and adjective stems, and verb and adjective stems. All of these create spurious ambiguities in compound formation. The N/V homography is the most frequent one of these in Swedish, e.g. BLICK (`gaze', N) vs. BLICKA (`to gaze', V) the stem of which also is blick.

The readings generated for the (e.g. 5,000 or 10,000) most frequent word-forms should be carefully evaluated. Marginal words, or marginal forms of some words, should not unreasonably create ambiguities for the most frequent words. For example, in Swedish, the finite auxiliaries är `is', har `has', kan `can (modal auxiliary)' have the rank numbers 7, 15, and 23, respectively, based on a count of 1 million word-forms (Allén 1972). It would not be reasonable to regard all instances of these word-forms as ambiguous due to the existence of exceedingly rare homonymous words like the noun ÄR `air', the non-standard abbreviation HAR `hectare', or the noun KAN (oriental title).

(S-4) Optimize the morphological description.

This strategy is a complement to (S-3). A conspicuous ambiguity problem in the inflectional morphology of several languages is the interpretation of word-forms where inflectional categories are represented by zero morphs. Consider a Swedish perfect participle word-form such as hoppa+d+e `jumped' (instantiating the lexeme HOPPA `to jump') with the following maximal set of readings (there is an additional past tense reading hoppa+de `jumped'):

(i) (”hoppa” V PERF PCP UTR DEF PL NOM)
(”hoppa” V PERF PCP NEU DEF PL NOM)
(”hoppa” V PERF PCP UTR INDEF PL NOM)
(”hoppa” V PERF PCP NEU INDEF PL NOM)

(ii) (”hoppa” V PERF PCP UTR DEF SG NOM)
(”hoppa” V PERF PCP NEU DEF SG NOM)

Are these really six different morphological readings? The two readings in group (ii) differ only in regard to the minor feature of gender (NEU = neuter gender, UTR = non-neuter gender). The readings in group (i) differ in gender and definiteness. Using the notion ”underspecification”, the groups could also be consistently represented as:

(i') (”hoppa” V PERF PCP UTR/NEU DEF/INDEF PL NOM)

(ii') (”hoppa” V PERF PCP UTR/NEU DEF SG NOM)

However, (i') and (ii') cannot be collapsed further in terms of number (SG/PL) into one underspecified representation because there is no indefinite singular reading for this word-form.

Optimizing the morphological description by allowing consistent feature

underspecification considerably reduces the amount of ambiguities generated by morphological analysis. On the other hand, it must be kept in mind that underspecification implies information decrease.

(S-5) Test and refine the lexicon iteratively by applying it to new texts.

Every major step of lexical optimization should be followed by an application of the updated lexicon and morphology to sufficiently large amounts of text. Only by careful inspection of such output is it possible to evaluate the behaviour of the system, or to spot remaining problems or unforeseen mistakes caused by the optimization decisions. It is in fact a task that never ends to update a large lexicon to cope with more and more words and constructions brought in from fresh texts that belong to many different text types.

A word-form and the readings generated for it by a morphological analyzer together constitute a cohort (see Chapter 2 for details). A cohort is ambiguous if it contains more than one reading. Of course, lexical and morphological optimization in the spirit of (S-3, S-4) eliminates some prospective morphological ambiguities but not all. Some of the remaining ambiguities may be discarded using the principles of local disambiguation proposed by Karlsson (1989, 1990, 1992).

Local disambiguation is performed solely within cohorts, by examination and internal comparison of the readings at hand, using no cohort-external information. Consider the following cohorts generated by SWETWOL, a comprehensive morphological analyzer for Swedish (Karlsson 1992). English glosses are added below the individual readings (many of the spurious readings have decidedly odd glosses).

```
("<bytesbil>"
        ("bytes_bil" N UTR INDEF SG NOM)
        `second-hand car'
        ("by_tes_bil" N UTR INDEF SG NOM))
        `village thesis car'

("<tonkontroll>"
        ("ton_kontroll" N UTR INDEF SG NOM)
        `tone control'
        ("ton_kon_troll" N NEU INDEF SG/PL NOM))
        `tone cone goblin'

("<bankomat>"
        ("bankomat" N UTR INDEF SG NOM)
        `cash-dispenser'
        ("ban_koma" N NEU DEF SG NOM)
        `track coma'
        ("ban_ko_mat" N UTR INDEF SG NOM))
        `track cow food'

("<konkurrenssamhälle>"
        ("konkurrens_samhälle" N NEU INDEF SG NOM)
        `competition society'
        ("kon_kur_rens_samhälle" N NEU INDEF SG NOM))
        `cone cure offal society'

("<konsultuppdrag>"
        ("konsult_uppdrag" N NEU INDEF SG/PL NOM)
        `consultant task'
```

(”konsult_uppdra” V ACT IMP)
`commission to consult' (imperative)
(”kon_sul_tupp_drag” N NEU INDEF SG/PL NOM)
`cone sole cock pull' (noun)
(”kon_sul_tupp_dra” V ACT IMP))
`cone sole cock draw' (imperative)

Many spurious compound readings may obviously be discarded if the respective cohort contains some reading with fewer compound boundaries, above indicated by the understroke ”_”. This is the Compound Elimination Principle that constitutes one important subcomponent of local disambiguation:

(S-6) If a cohort c contains readings with n and m compound boundaries, discard all readings with m compound boundaries if m > n.

For the first four examples, the Compound Elimination principle leaves the following readings as uniquely correct ones: bytes_bil `second-hand car', ton_kontroll `tone control', bankomat `cash-dispenser' (with no compound boundaries), konkurrens_samhälle `competition society'. Full disambiguation was thus achieved by the Compound Elimination Principle alone. More than one reading might be left pending if they are at the same level of compound complexity, cf. the two readings with one compound boundary for konsult_uppdrag.

There are extremely few instances where the Compound Elimination Principle overgenerates and discards a reading that should have been retained. Even after extensive corpus study, only a handful of word-forms have been found that have two attested readings with variable number of compound boundaries, one reading always being non-compounded. One such word is the Swedish word finskor. This word has the interpretations (i) finsko+r `Finnish women', which is the indefinite plural of the lexeme FINSKA `Finnish woman', and (ii) fin_sko+r `fine shoes', which is the same form of a compound the parts of which are FIN `fine' and SKO `shoe'. Here, the more complex reading must be separately treated in the lexicon in order to ensure that it will not be erroneously discarded.

There is also the Derivative Elimination Principle:

(S-7) If a cohort c contains readings some of which are derived and others simplex, discard the derived ones.

Derivational status in the output of SWETWOL is technically indicated by the presence of a feature the name of which contains the initial segments DER-. In the following cohort, the second reading is spuriously derived from bill `share' (as an agricultural term) and must therefore be discarded, the proper reading being the upper one meaning `cheap':

(”<billig>”
(”billig” A UTR INDEF SG NOM)
`cheap'
(”billig” DER-ig A UTR INDEF SG NOM))
`share-like'

Both elimination principles (S-6, S-7) have a common basis: eliminate morphologically complex readings in favour of simpler ones. Local disambiguation has been implemented in two ways, by the present author as a Lisp program, and by Kimmo Koskenniemi as an AWK program running under Unix. The latter provides the option of being directly applied as a post-filter to TWOL, thus delivering immediate locally disambiguated output. See Karlsson (1992) for details and an evaluation of the quantitative impact of local disambiguation in Swedish.

5. Syntax in Constraint Grammar

Input to Constraint Grammar syntax is a morphologically analyzed and, in the optimal case, a morphologically fully disambiguated string of words. Constraint Grammar syntax is based on dependency and assigns flat, functional, surface syntactic labels, optimally one to each word-form in the sentence if the sentence is not truly ambiguous.

Most of the labels are drawn from the classical repertoire of heads and modifiers. Typical heads are e.g. subject, indirect object, object predicate complement, or adverbial. Typical modifiers are e.g. determiner, intensifier (premodifier of an adjective), postnominal modifier, or prepositional complement. This is what we mean by stating that Constraint Grammar syntax is functional. Constituent structure plays no direct explicit role. Of course, in actual practice there is a high degree of isomorphy between dependency-oriented functional syntax of this type and constituent structure syntax.

Constraint Grammar syntax is surface syntax rather than deep syntax because no syntactic structure is postulated or assigned that is not in direct correspondence with the word-form tokens that ”are there” in the sentence. Furthermore, the surface nature of Constraint Grammar syntax is emphasized by the fact that it is just an abstraction from morphological properties (drawn from lexicon and morphological analysis) and word order configurations. Constraint Grammar syntax maps morphological categories and word order information onto syntactic labels. In many respects, this approach relies heavily on the traditional conception of syntax, as opposed to modern conceptions that tend to emphasize the importance of constituent structure and abstract levels of representation.

The flatness of Constraint Grammar syntax means that no trees or other hierarchic structures are generated as a result of the parsing process. Rather, every word is assigned a syntactic label (code) which in no obvious way differs e.g. from the morphological properties of the word. The syntactic description of a sentence is constituted by the syntactic labels that make up a dependency tree. Technically, the syntactic codes are just appended (as a separate list) at the end of the respective reading.

The syntactic labels are basically of two types, indicating either heads or modifiers. Typical head functions are e.g. subject, direct object, indirect object, predicate complement of the subject, finite main verb, non-finite auxiliary, etc. The head functions are generally close to those of traditional grammar.

Modifier functions are labels indicating the head of a modifier, i.a. by giving the part of speech of the head and the direction where it is to be found. Thus, the label NN> could mean ”noun modifying some noun (normally the next head noun) to the right”, NN2> means ”noun modifying the second noun to the right”, <P means ”complement of the preposition to the left”, etc.

The set of modifier labels postulated in a thorough Constraint Grammar description is generally richer than that known from traditional grammar. It is left to the discretion of the grammarian to determine what the appropriate head labels and modifier labels are for the language being described. A natural desideratum is that the syntactic labels be easy to interpret, and easy to integrate with morphological and semantic analyses.

Here is a simple, maximally successful example of the output of Constraint Grammar syntax:

```
(”<Bill>”
        (”Bill” <Proper> N NOM SG (SUBJ)))
(”<saw>”
        (”see” <as/SVOC/A> <SVO> <SV> <InfComp> V PAST VFIN (+FMAINV)))
(”<the>”
        (”the” <Def> DET CENTRAL ART SG/PL (DN>)))
(”<little>”
```

("little" A ABS (AN>)))
("<dog>"
("dog" N NOM SG (OBJ)))
("<in>"
("in" PREP (<NOM ADVL)))
("<the>"
("the" <Def> DET CENTRAL ART SG/PL (DN>)))
("<park>"
("park" N NOM SG (<P)))

Bill is subject (SUBJ), saw is finite main verb (+FMAINV), the is determiner and modifier of a noun to the right (DN>), little is an adjective and modifier of a (normally the next) noun to the right (AN>), dog is direct object (OBJ), in is either postmodifier (<NOM) of the next nominal to the left (here, the noun dog), or head of a clause-level adverbial (ADVL). The noun park is complement of a preposition to the left (<P).

Notice how the attachment ambiguity is represented here, just as two alternate syntactic labels in the same representation, rather than as two totally distinct representations. Of course, there is no non-semantic way of resolving the attachment ambiguity of the adverbial in the park. This ambiguity is therefore properly left unresolved.

An important aspect of the flatness of syntactic representation is the use of the notion "verb chain". Complex verbals make up one flat, perhaps discontinuous string rather than a deep hierarchical structure. Given the two verbal distinctions finite (+F) versus non-finite (BF), and main (MAIN) versus auxiliary (AUX), English verb chains could be flatly represented as follows:

Finite chains

is/was/were/has/have/had/does +FMAINV (no more verbs rightwards)
sleeps/sees/saw +FMAINV
has/had seen +FAUXV BFMAINV
is/was/were seeing/seen +FAUXV BFMAINV
has/had been seeing/seen +FAUXV BFAUXV BFMAINV
may/might/should/can//dare/ought to/used to see +FAUXV BFMAINV
had to/was about to/is able to/was willing to see +FAUXV BFMAINV
may/should have seen +FAUXV BFAUXV BFMAINV
may/might have been seeing/seen +FAUXV BFAUXV BFAUXV BFMAINV
is being seen +FAUXV BFAUXV BFMAINV
had been being seen +FAUXV BFAUXV BFAUXV BFMAINV
might have been being seen +FAUXV BFAUXV BFAUXV BFAUXV BFMAINV
do/don't/does/did like +FAUXV BFMAINV
has been doing +FAUXV BFAUXV BFMAINV
appears to/seems to/happens to/attempts to come +FMAINV BFMAINV
hopes to go +FMAINV BFMAINV
begins going +FMAINV BFMAINV
saw (Bill) go(ing) +FMAINV BFMAINV
do want to come +FAUXV BFMAINV BFMAINV
would like to be +FAUXV BFMAINV BFMAINV
wants to have +FMAINV BFMAINV
expected (Bill) to claim to have been seeing (Sue) +FMAINV BFMAINV BFAUXV BFAUXV BFMAINV

Nonfinite chains

to have seen BFAUXV BFMAINV
to be seeing BFAUXV BFMAINV
having seen BFAUXV BFMAINV
to have been seeing/seen BFAUXV BFAUXV BFMAINV
to be being seen BFAUXV BFAUXV BFMAINV
to have been being seen BFAUXV BFAUXV BFAUXV BFMAINV

A maximally precise Constraint Grammar description would actually use distinct labels for all members of the same verb chain. It would also have all nominal dependents labelled according to, i.e. stamped onto, their respective verbal regent. This descriptive practice was developed in the original FPARSE parsing system for Finnish that was a precursor to the Constraint Grammar formalism. Thus, in a sentence such as:

Bill saw Sue leave the house yesterday.

the word saw would be labelled ”finite main verb”, Bill ”subject of the finite main verb”, leave ”1st infinitive in the verb chain”, Sue ”subject of the 1st infinitive in the verb chain”, house ”object of the 1st infinitive in the verb chain”, and yesterday either ”adverbial of the 1st infinitive in the verb chain” or ”adverbial of the finite main verb”, presupposing that the attachment ambiguity is real (Karlsson 1986). If the word yesterday had occurred in sentence-initial position, it would have been assigned only the syntactic label ”adverbial of the finite main verb”. Under such a scheme, all syntactic dependencies are succinctly expressed.

As pointed out repeatedly above, Constraint Grammar syntactic parsing is performed by the same mechanism of discarding alternatives that is used for morphological disambiguation. Thus, prior to syntactic parsing, every word-form is assigned all possible syntactic functions it can have. This assignment is normally done by morphosyntactic mapping statements (see Chapter 2). The syntactic constraints discard improper alternatives, optimally yielding a unique, i.e. a fully disambiguated representation.

6. Theoretical epilogue

The Constraint Grammar approach to parsing relies on two traditional insights:

(i) Language is an open-ended system where there is no strict demarcation line between grammatical and ungrammatical sentences. Therefore all grammars are bound to leak.

(ii) The cornerstone of syntax is morphology, especially the language-particular systems of morphological features. Syntactic rules are generalizations telling (a) how word-forms, conceived as complexes of morphological features, occur in particular word order configurations, and (b) what natural classes, ”syntactic functions”, can be isolated and inferred in such configurations.

Given this much, it seemed necessary to anchor the notion ”constraint” in surface-near facts of word morphology, syntactic dependency, and word order, rather than in more abstract principles of structuring. One consequence of this is that an explicit level of constituent structure is dispensed with in Constraint Grammar. However, if needed, constituent structures can be inferred from the dependency trees typical of Constraint Grammar syntactic tagging.

Several more desiderata concerning the grammar formalism and the parsing scheme were identified. Two important requirements shall be repeated here. First, the grammar formalism must not be based on such a rigid and idealized conception of grammatical correctness that parsing of real texts turns out to be practically impossible because many sentences either run

counter to the grammar or simply are not catered for by the rules postulated so far. The requisite liberalism could be achieved by making the formalism eliminative, i.e. reductionistic, rather than licencing. Everything is licenced that is not explicitly eliminated. Second, the grammar and the parser should be efficient enough. Each grammar statement in isolation, as well as the grammar as a whole, should be testable, modifiable, and efficiently applicable to large amounts of unedited running text.

It is not a primary aim of Constraint Grammar to express or describe morphological and syntactic phenomena as maximally few, unitary, fairly abstract, and complex generalizations. Rather, bits and pieces of the same phenomenon may be expressed by several different constraints. For example, CG constraints have individual words, or single morphological properties, as their object domains much more frequently than the rules of mainstream grammar-based parsers do. The fundamental purpose of Constraint Grammar is thus to demonstrate that descriptively reasonable and practically efficient parsing grammars can be designed that are based on pure surface generalizations. The Constraint Grammar conception of morphosyntactic structure is actually close to ”traditional syntax” whose core parts are inflection, concord, and order.

The constraints of Constraint Grammar are on a lower level of theoretical abstraction than the rules of current formal syntax, say, Government and Binding theory or Generalized Phrase Structure Grammar. On the other hand, Constraint Grammar constraints are linguistically more abstract than the statistical optimization functions used in purely stochastic parsing. One of the main concerns of Constraint Grammar is in fact to find a level of descriptive linguistic abstraction for parsing purposes that would somehow reconcile the conflicts often encountered when the requirements of theoretical generality, variability of language use, and parsing efficiency clash. Furthermore, in the domain of linguistic description, we have wanted to make heuristics (probabilism) possible, but in a regimented fashion, using linguistic categories as the basis of heuristic constraints, and above all by stressing that heuristics is subordinate to analysis based on safer, purely linguistic means.

Constraint Grammar descriptions of individual languages look more atomistic, redundant, and repetitive than other, more theoretically inclined grammars. Such an impression is corroborated by the fact that many constraints express redundancies rather than positive restrictions. These constraints explicitly discard what is not feasible in a certain configuration rather than tell in positive terms what the configuration could or should look like. This explicitly reductionistic approach does not seem to have any obvious counterparts in the grammatical literature or in current formal syntactic theory.

Recall that a Constraint Grammar parse starts with all alternatives present after lexical analysis and syntactic mapping (cf. Chapter 2). It is precisely the reductionistic approach that guarantees that every input will receive some analysis. In other words, the Constraint Grammar Parser is robust and it can be applied to running text of arbitrary length and complexity. Currently, the English Constraint Grammar description described in the later chapters of this book is being used for morphological and syntactic parsing, i.e. morphosyntactic tagging, of the whole Bank of English which contains 200 million words of text.

The tagging of this corpus is a collaboration project between the Research Unit for Computational Linguistics, University of Helsinki, HarperCollins Publishers, Glasgow, and the COBUILD Group at the University of Birmingham. The tagging of the Bank of English proceeds at a speed of 10 million words per month. As these concluding remarks are being written in December 1993, 80 million words of parsed English have already been delivered to the COBUILD group in Birmingham, and the tagging project is due for completion in early 1995. From the viewpoint of parsing efficiency, we observe that the English Constraint Grammar Parser is capable of morphosyntactically analyzing more than 1 million words of running text per hour.

Notice, in passing, that the overall reductionistic set-up gives an interesting psycholinguistic prediction. In terms of processing load, less processing is required under Constraint Grammar to turn out an ambiguous and therefore unclear analysis than to turn out a disambiguated and therefore structurally clear analysis. In our opinion, this seems to be a

reasonable psycholinguistic hypothesis a priori. Mental effort is needed for achieving clarity, precision, and maximal information. Less efforts imply (retention of) unclarity and ambiguity, i.e. information decrease. In several types of parsers, rule applications create rather than discard ambiguities: the more processing, the less unambiguous information.

On the positive side, the concreteness of the individual CG constraints makes them (fairly) perspicuous and easy to understand. Above all, it is simple to test and evaluate the consequences and predictions of such concrete constraints, and to develop and modify them, if necessary. Ordering plays no essential role in Constraint Grammar. This property contributes to the manageability of individual constraints.

The number of constraints does not seem to grow prohibitively large in a full Constraint Grammar description of a language. The sum total of the morphological and syntactic disambiguation constraints postulated by Voutilainen and Anttila for English is less than 2,000. Should this be considered a small or a large number? We feel inclined to say ”fairly small”, in view of the fact that an ordinary language user knows tens (if not hundreds) of thousands of lexical items, and hundreds of thousands (if not millions) of senses. In this perspective, we would regard the number of low-level morphosyntactic constraints, some 2,000 in the present description of English, as relatively moderate. This magnitude makes syntax seem efficiently learnable which it obviously is, as proven by normal first-language acquisition. Thus, natural language syntax would not be an overwhelmingly complex area of human cognition.

Notes to Chapter 1

1. The first version of this chapter was written in 1989 but was not published. It has occasionally been referred to in the literature under the title “Parsing and Constraint Grammar”. A condensed version of the basic ideas of Constraint Grammar was first published as Karlsson (1990). The details of the Constraint Grammar formalism are presented, in elaborated and finalized form, as Chapter 2 in the present volume. Constraint Grammar is a clarification and further development of ideas originally conceived and implemented in the FPARSE parser of Finnish (Karlsson 1985, 1986).

2. Not to be confused with the generative notion Core Grammar, nor with Categorial Grammar, both occasionally abbreviated CG.

3. CLAWS = Constituent-Likelihood Automatic Word-tagging System.

4. UCREL = Unit for Computer Research on the English Language, University of Lancaster.

5. Crystal (1991) gives a different definition (s.v. disambiguate): “A term used in LINGUISTICS, and especially in TRANSFORMATIONAL GRAMMAR, to refer to an analysis which demonstrates the alternative STRUCTURAL interpretations of an AMBIGUOUS SENTENCE Y”. In Constraint Grammar, disambiguation does not mean “bring out all alternatives” but rather “pick the appropriate alternative(s) by discarding one or more inappropriate ones”. The Constraint Grammar notion of morphological disambiguation is functionally similar to the notion “homograph separation”, discussed e.g. by Allén (1971) from the viewpoint of computational lexicography.

References

Aarts, J. & Th. van den Heuvel
1985 Computational Tools for the Syntactic Analysis of Corpora, Linguistics 23: 303-332.

Adams, V.
1973 An Introduction to Modern English Word-Formation. London: Longman.

Akkerman, E., H. Voogt - van Zutphen & W. Meijs
1988 Computerized Lexicon for Word-Level Tagging. ASCOT Report No 2. Amsterdam: Rodopi.

Allén, S.
1971 Nusvensk frekvensordbok baserad på tidningstext. 1. Stockholm: Almqvist. & Wiksell.
1972 Tiotusen i topp. Ordfrekvenser i tidningstext. Stockholm: Almqvist. & Wiksell.

Alshawi, H. (ed.)
1992 The Core Language Engine. Cambridge, Mass.: The MIT Press.

Altmann, G.
1988 Ambiguity, Parsing Strategies, and Computational Models, Language and Cognitive Processes 3: 73-97.

Altmann, G. & G. Square
1985 The Resolution of Local Syntactic Ambiguity by the Human Sentence Processing Mechanism, Proceedings of the Second Conference of the European Chapter of the ACL. 123-127.

Anderson, S.R.
1988 Morphological theory, in: Newmeyer, F.J. (ed.) Linguistics: The Cambridge Survey. Vol. I: Linguistic Theory: Foundations. Cambridge: Cambridge University Press. 146-191.

Aronoff, M.
1976 Word Formation in Generative Grammar. Cambridge, Massachusetts & London: The MIT Press.

Atwell, E.
1987a Constituent-likelihood Grammar, in: Garside, Leech & Sampson (eds.), 57-65.
1987b How to Detect Grammatical Errors in a Text Without Parsing It, Proceedings of the Third Conference of the European Chapter of the ACL. 38-45.

Atwell, E. & S. Elliott
1987 Dealing with ill-formed English text, in: Garside, Leech & Sampson (eds.), 120-138.

Bauer, L.
1983 English Word-formation. Cambridge: Cambridge University Press.

Beale, A.
1988 Lexicon and Grammar in Probabilistic Tagging of Written English, Proceedings of the 26th Annual Meeting of the ACL. 211-216.

Bear, J. & J. Hobbs
1988 Localizing Expression of Ambiguity, Proceedings of the Second Conference on Applied Natural Language Processing, ACL. 235-242.

Berg, S.
1978 Olika lika ord. Svenskt homograflexikon. Stockholm: Almqvist. & Wiksell International.

Berwick, R.
1980 Computational Analogues of Constraints on Grammars: A Model of Syntactic Acquisition, Proceedings of the 18th Annual Meeting of the ACL. 49-53.
1983 Syntactic Constraints and Efficient Parsability, Proceedings of the 21st Annual Meeting of the ACL. 119-122.

Berwick, R., S. Abney & C. Tenny (eds.)
1991 Principle-Based Parsing: Computation and Psycholinguistics. Dordrecht/Boston/London: Kluwer Academic Publishers.

Berwick, R. & A. Weinberg
1984 The Grammatical Basis for Linguistic Performance. Cambridge, Mass.: The MIT Press.

Biber, D.
1988 Variation across Speech and Writing. Cambridge: Cambridge University Press.

Billot, S. & B. Lang
1989 The Structure of Shared Forests in Ambiguous Parsing, Proceedings of the 27th Annual Meeting of the ACL. 143-151.

Binot, J.-L.
1987 Fragmentation and Part of Speech Disambiguation, Proceedings of the Third Conference of the European Chapter of the ACL. 284-290.

Black, A., G. Ritchie, S. Pulman & G. Russell
1987 Formalisms for Morphographemic Description, Proceedings of the Third Conference of the European Chapter of the ACL. 11-18.

Black, E.
1993 Statistically-Based Computer Analysis of English, in: E. Black, R. Garside & G. Leech (eds.) 1993, 1-16.

Black, E., R. Garside & G. Leech (eds.)
1993 Statistically Driven Computer Grammars of English: The IBM/Lancaster Approach. Amsterdam/Atlanta: Rodopi.

Blackwell, S.
1987 Syntax versus orthography: problems in the automatic parsing of idioms, in: Garside, Leech & Sampson (eds.), 110-119.

Bloemberg, W. & M. Kesselheim
1988 A System for Creating and Manipulating Generalized Wordclass Transition Matrices from Large Labelled Text Corpora, Proceedings of the 12th International Conference on Computational Linguistics. Vol. 1, John von Neumann Society for Computing Sciences, Budapest. 49-53.

Bobrow, R. & M. Bates
1982 Design Dimensions for Non-Normative Understanding Systems, Proceedings of the 20th Annual Meeting of the ACL. 153-156.

Boguraev, B.
1979 Automatic Resolution of Linguistic Ambiguities. Cambridge: University of Cambridge

Computer Laboratory, Technical Report No. 11.

Boguraev, B. & T. Briscoe
1987 Large Lexicons for Natural Language Processing: Utilising the Grammar Coding System of LDOCE, Computational Linguistics 13. 203-218.

Boguraev, B. & T. Briscoe (eds.)
1989 Computational Lexicography for Natural Language Understanding. London and New York: Longman.

Boguraev, B., T. Briscoe, J. Carroll, D. Carter & C. Grover
1987 The Derivation of a Grammatically Indexed Lexicon from the Longman Dictionary of contemporary English, Proceedings of the 25th Annual Meeting of the ACL. 193-200.

Bresnan, J. (ed.)
1982 The Mental Representation of Grammatical Relations. Cambridge, MA: The MIT Press.

Briscoe, T.
1987 Deterministic Parsing and Unbounded Dependencies, Proceedings of the Third Conference of the European Chapter of the ACL. 211B217.
1990 English Noun Phrases Are Regular: A Reply to Professor Sampson, in: J. Aarts & W. Meijs (eds.), Theory and Practice in Corpus Linguistics. 45-60. Amsterdam: Rodopi.

Briscoe, T. & J. Carroll
1991 Generalised Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars. Cambridge: University of Cambridge Computer Laboratory, Technical Report No. 224.

Briscoe, T., C. Grover, B. Boguraev & J. Carroll
1987 A Formalism and Environment for the Development of a Large Grammar of English, IJCAI-87. Vol. 2: 703-708.

Byrd, R., J. Klavans, M. Aronoff & F. Anshen
1986 Computer Methods for Morphological Analysis, Proceedings of the 24th Annual Meeting of the ACL. 120-127.

Carey, P., J. Mehler & T. Bever
1970 When do we compute all the interpretations of an ambiguous sentence? in: G. Flores d'Arcais & W. Levelt (eds.), 61-75.

Carroll, J., B. Boguraev, C. Grover & T. Briscoe
1988 Development Environment for Large Natural Language Grammars. Cambridge: University of Cambridge Computer Laboratory, Technical Report No. 127.

Carroll, J. & C. Grover
1989 The Derivation of a Large Computational Lexicon for English from LDOCE, in: B. Boguraev & T. Briscoe (eds.), 117-133.

Carstairs, A. & J. Stemberger
1988 A Processing Constraint on Inflectional Homonymy, Linguistics 26: 601-617.

Carston, R.
1989 Modularity and Linguistic Ambiguity, University College London, Working Papers in

Linguistics 1: 340-351.

CELEX Newsletter. No. 4, December 1988.

Church, K.
1980 On Parsing Strategies and Closure, Proceedings of the 18th Annual Meeting of the ACL. 107-111.
1983 A Finite-State Parser for Use in Speech Recognition, Proceedings of the 21st Annual Meeting of the ACL. 91-97.
1988 A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text, Proceedings of the Second Conference on Applied Natural Language Processing, ACL. 136-143.
1992 Current Practice in Part of Speech Tagging and Suggestions for the Future, in: Simmons (ed.) 1992. Sbornik praci: In Honor of Henry Kucera. Michigan Slavic Studies. 243-251.

Church, K. & P. Hanks
1989 Word Association Norms, Mutual Information, and Lexicography, Proceedings of the 27th Annual Meeting of the ACL. 76-83.

Church, K. & D. Hindle
1990 Collocational Constraints and Corpus-Based Linguistics, TIS 1990. 43-47.

Church, K. & R. Patil
1982 Coping with Syntactic Ambiguity or How to Put the Block in the Box on the Table, American Journal of Computational Linguistics 8, No. 3-4.

COBUILD = Collins COBUILD English Language Dictionary. 1987 LondonCGlasgow: Collins.

Cottrell, G.
1989 A Connectionist Approach to Word Sense Disambiguation. London: Pitman.
Crystal, D.
1967 English, Lingua 17: 24-56.
1991 Dictionary of Linguistics and Phonetics. 3rd edition. London: Basil Blackwell.

Dagan, I. & A. Itai
1990 Automatic Processing of Large Corpora for the Resolution of Anaphora References, in: H. Karlgren (ed.), Papers presented to the 13th International Conference on Computational Linguistics, Vol. 3, Helsinki. 330-332.

DeRose, S.
1988 Grammatical Category Disambiguation by Statistical Optimization, Computational Linguistics 14: 31-39.
1991 An Analysis of Probabilistic Grammatical Tagging Methods, in: JohanssonCStenström (eds.), 9-14.

Dowty, D., L. Karttunen & A. Zwicky (eds.)
1985 Natural Language Parsing. Psychological, Computational, and Theoretical Perspectives. Cambridge: Cambridge University Press.

Ejerhed, E.
1988 Finding Clauses in Unrestricted Text by Finitary and Stochastic Methods, Proceedings of the Second Conference on Applied Natural Language Processing, ACL. 219-227.

Flores d'Arcais, G. & R. Jarvella (eds.)

1983 The Process of Language Understanding. Chichester: John Wiley. & Sons.

Flores d'Arcais, G. & W. Levelt
1970 Advances in Psycholinguistics. Amsterdam: North-Holland Publishing Company.

Flores d'Arcais G. & R. Schreuder
1983 The Process of Language Understanding: A Few Issues in Contemporary Psycholinguistics, in: Flores d'Arcais & Jarvella (eds.), 1-41.

Ford, M., J. Bresnan & R. Kaplan
1982 A Competence-Based Theory of Syntactic Closure, in: Bresnan (ed.), 727-796.

Francis, W. & H. Kucera
1982 Frequency Analysis of English Usage: Lexicon and Grammar. Boston, Mass.: Houghton-Mifflin Company.

Fraser N.
1989 Parsing and Dependency Grammar, University College, London, Working Papers in Linguistics 1: 296-319.

Garside, R.
1987 The CLAWS word-tagging system, in: Garside, Leech & Sampson (eds.), 30-41.

Garside, R. & F. Leech
1987 The UCREL Probabilistic Parsing System, in: Garside, Leech & Sampson (eds.), 66-81.

Garside, R., G. Leech & G. Sampson (eds.)
1987 The Computational Analysis of English. London and New York: Longman.

Gazdar, G., E. Klein, G. Pullum & I. Sag
1985 Generalized Phrase Structure Grammar. Oxford: Basil Blackwell.

Gazdar, G. & C. Mellish
1989 Natural Language Processing in LISP. An Introduction to Computational Linguistics. Wokingham, England: Addison-Wesley.

Gorfein, D. (ed.)
1989 Resolving Semantic Ambiguity. New York etc.: Springer Verlag.

Granger, R.
1982 Scruffy Text Understanding: Design and Implementation of `Tolerant' Understanders, Proceedings of the 20th Annual Meeting of the ACL. 157-160.

Green, G.M.
1985 The description of inversions in Generalized Phrase Structure Grammar, Proceedings of the 11th Annual Meeting of the Berkeley Linguistic Society.

Greene, B. & G. Rubin
1971 Automated Grammatical Tagging of English. Department of Linguistics, Brown University, Providence, Rhode Island.

Grover, C. & T. Briscoe, J. Carroll & B. Boguraev

1989 The ALVEY Natural Language Tools Grammar. Cambridge: University of Cambridge Computer Laboratory, Technical Reports No. 162.

Hearst, M.
1990 Applying Lexical Conglomeration to Restricted Text Interpretation, TIS 1990. 61-65.

Heidorn, G.
1982 Experience with an Easily Computed Metric for Ranking Alternative Parses, Proceedings of the 20th Annual Meeting of the ACL. 82-84.

Herz, J. & M. Rimon
1990 Local Syntactic Constraints, A preliminary paper submitted to the 5th Conference of the European Chapter of the ACL. The Computer Science Department, the Hebrew University of Jerusalem.

Heuvel, Th. van den
1988 TOSCA: An Aid for Building Syntactic Databases, Literary and Linguistic Computing 3: 147-151.

Hindle, D.
1983 Deterministic Parsing of Syntactic Non-fluencies, Proceedings of the 21st Annual Meeting of the ACL. 123-128.
1989 Acquiring Disambiguation Rules from Text, Proceedings of the 27th Annual Meeting of the ACL. 118-125.
1990 Noun Classification from Predicate-Argument Structures, Proceedings of the 28th Annual Meeting of the ACL. 268-275.

Hindle, D. & M. Rooth
1991 Structural Ambiguity and Lexical Relations, Proceedings of the 29th Annual Meeting of the ACL. 229-236.

Hinkelman, E. & J. Allen
1989 Two Constraints on Speech Act Ambiguity, Proceedings of the 27th Annual Meeting of the ACL. 212-219.

Hirst, G.
1987 Semantic Interpretation and the Resolution of Ambiguity. Cambridge: Cambridge University Press.
1989 Computational Models of Ambiguity Resolution, in: Gorfein (ed.), 255-275.

Hoppenbrouwers, G., P. Seuren & A. Weijters (eds.)
1985 Meaning and the Lexicon. Dordrecht: Foris Publications.

Huddleston, R.
1984 Introduction to the Grammar of English. Cambridge: Cambridge University Press.

Hurum, S.
1988 Handling Scope Ambiguities in English, Proceedings of the Second Conference on Applied Natural Language Processing, ACL. 58-65.

Jensen, K.
1986 Parsing Strategies in a Broad-coverage Grammar of English. (Research Report RC 12147.) Yorktown Heights, NJ: IBM Thomas J. Watson Research Center.

Jensen, K. & J. L. Binot
1987 Disambiguating Prepositional Phrase Attachments by Using On-Line Dictionary Definitions, Computational Linguistics 13: 251-260.
1988 Dictionary Text Entries as a Source of Knowledge for Syntactic and Other Disambiguations, Proceedings of the Second Conference on Applied Natural Language Processing, 152-159.

Jensen, K. & G. Heidorn
1983 The Fitted Parse: 100% Parsing Capability in a Syntactic Grammar of English, Proceedings of the Conference on Applied Natural Language Processing, ACL, 93-98.

Jensen, K., G. E. Heidorn & S. D. Richardson (eds.)
1993 Natural Language Processing: The PLNLP Approach. Boston: Kluwer.

Johansson, S.
1986 The Tagged LOB Corpus: User's Manual. Bergen: Norwegian Computing Centre for the Humanities.

Johansson, S. & K. Hofland
1989 Frequency Analysis of English Vocabulary and Grammar. Oxford: Clarendon Press.

Johansson, S. & A. B. Stenström (eds.)
1991 English Computer Corpora. Selected Papers and Research Guide. Berlin: Mouton de Gruyter.

Joshi, A.
1985 Tree Adjoining Grammars: How much context-sensitivity is required to provide reasonable structural descriptions? in: D. DowtyCL. KarttunenCA. Zwicky (eds.), 206-250.

Karlsson, F.
1985 Parsing Finnish in terms of Process Grammar, in: F. Karlsson (ed.), Computational Morphosyntax: Report on Research 1981-1984. (Publications of the Department of General Linguistics, University of Helsinki, No. 13.) Helsinki: University of Helsinki, Department of General Linguistics, 137-176.
1986 Process Grammar, in: Ö. Dahl (ed.), Papers from the Ninth Scandinavian Conference of Linguistics. Stockholm: Institute of Linguistics, University of Stockholm, 162-171.
1989 Parsing and Constraint Grammar. [Unpublished MS.]
1990 Constraint Grammar as a Framework for Parsing Running Text, in: H. Karlgren (ed.), COLING 90: Papers presented to the 13th International Conference on Computational Linguistics, Vol. 3. 168-173.
1992 SWETWOL: A Comprehensive Morphological Analyser for Swedish, Nordic Journal of Linguistics 15: 1-45.

Karlsson, F. & K. Koskenniemi
1990 BETA-ohjelma kielentutkijan apuvälineenä. Helsinki: Yliopistopaino.

Karlsson, F., A. Voutilainen, J. Heikkilä & A. Anttila
1991. Constraint Grammar: A Language-Independent System for Parsing Running Text, with an Application to English, Workshop Notes from the Ninth National Conference on Artificial Intelligence (AAAI 91): Natural Language Text Retrieval. Anaheim, California: American Association for Artificial Intelligence.

Karttunen, L. & K. Wittenburg
1983 A two-level morphological analysis of English, Texas Linguistic Forum 22: 217-228.

Karttunen, L., R. M. Kaplan & A. Zaenen
1992 Two-Level Morphology with Composition, Proceedings of the fifteenth International Conference on Computational Linguistics, COLING 92, Vol. I. 141-148.

Kelly, E. & P. Stone
1975 Computer Recognition of English Word Senses. Amsterdam: North Holland.

Kitano, H. & H. TomabechiCL. Levin
1989 Ambiguity Resolution in the DMTRANS PLUS, Proceedings of the Fourth Conference of the European Chapter of the ACL, 72-79.

Klein, S. & R. F. Simmons
1963 A Computational Approach to Grammatical Coding of English Words, JACM 10: 334-347.

Kooij, J.
1971 Ambiguity in Natural Language. An Investigation of Certain Problems in its Linguistic Description. Amsterdam: North Holland.

Koskenniemi, K.
1983 Two-level Morphology: A General Computational Model for Word-form Production and Generation. (Publications of the Department of General Linguistics, University of Helsinki, No. 11.) Helsinki: University of Helsinki, Department of General Linguistics.
1985 A general two-level computational model for word-form recogni¬tion and production, in: F. Karlsson (ed.), Computational Morphosyntax. Report on Research 1981-84. (Publications of the Department of General Linguistics, University of Helsinki, No. 13.) Helsinki: University of Helsinki, Department of General Linguistics, 1-18.
1990 Finite-state Parsing and Disambiguation, in: H. Karlgren (ed.), COLING 90: Papers presented to the 13th International Conference on Computational Linguistics, Vol. 2. 229-232.

Koskenniemi, K., P. Tapanainen & A. Voutilainen
1992 Compiling and Using Finite-state Syntactic Rules, Proceedings of the fifteenth International Conference on Computational Linguistics, COLING 92, Vol. I. 156-162.

Krieger, H. U. & J. Nerbonne
1991 Feature-based inheritance networks for computa¬tional lexicons, in: J. Nerbonne & K. Netter (ed.), The Structure of the Lexicon in NL Systems. Lecture notes, The Third European Summer School in Language, Logic and Information. Saarbrücken: Universität des Saarlandes.

Kroch, A. & D. Hindle
1982 On the Linguistic Character of Non-Standard Input, Proceedings of the 20th Annual Meeting of the ACL, 161-163.

Krovetz, R.
1990 Information Retrieval and Lexical Ambiguity, TIS 1990, 70-72.

Kucera, H. & W. N. Francis
1967 Computational analysis of present-day American English. Providence, R. I.: Brown University Press.

Laalo, K.

1990 Säkeistä patoihin. Suomen kielen monitulkintaiset sananmuodot. Helsinki: Suomalaisen Kirjallisuuden Seura.

LDOCE = Longman Dictionary of Contemporary English. New Edition. 1987 London: Longman.

Leech, G. & R. Garside
1991 Running a Grammar Factory: The Production of Syntactically Analysed Corpora or `Treebanks', in: S. Johansson & A. B. Stenström (eds.), 15-32.

Leech, G.
1992 Corpora and Theories of Linguistic Performance, in: J. Svartvik (ed.), 105-122.
Linden, E. J. van derCW. Kraaij
1990 Ambiguity Resolution and the Retrieval of Idioms: Two Approaches, in: H. Karlgren (ed.), COLING 90: Papers presented to the 13th International Conference on Computational Linguistics, Vol. 2. 245-250.

Litman, D. & J. Hirschberg
1990 Disambiguating Cue Phrases in Text and Speech, in: H. Karlgren (ed.), COLING 90: Papers presented to the 13th International Conference on Computational Linguistics, Vol. 2. 251-256.

Lyons, J.
1977 Semantics. 2 Vols. Cambridge: Cambridge University Press.

Lytinen, S.
1990 Robust Processing of Terse Text, TIS 1990, 10-14.

Marchand, H.
1969 The Categories and Types of Present-Day English Word-Formation. Second edition. München: Verlag C. H. Beck.

Marcken, C. de
1990 Parsing the LOB Corpus, Proceedings of the 28th Annual Meeting of the ACL, 243-251.

Marcus, M.
1980 Theory of Syntactic Recognition for Natural Language. Cambridge, Mass.: The MIT Press.
1982 Building Non-Normative Systems - the Research for Robustness, an Overview, Proceedings of the 20th Annual Meeting of the ACL, 152.

Marshall, I.
1983 Choice of grammatical word-class without global syntactic analysis: tagging words in the LOB Corpus, Computers in the Humanities 17: 139-150.
1987 Tag selection using probabilistic methods, in: R. GarsideCG. LeechCG. Sampson (eds.), 42-56.

Maruyama, H.
1990 Structural Disambiguation with Constraint Propagation, Proceedings of the 28th Annual Meeting of the ACL, 31-38.

Matthews, P. H.
1974 Morphology. Cambridge: Cambridge University Press.

McCord, M.
1990 Slot Grammar, A System for Simpler Construction of Practical Natural Language Grammars, in: R. Studer (ed.), Natural Language and Logic. (Lecture Notes in Artificial Intelligence 459.) Berlin: Springer Verlag, 118-145.

McDonald, D.
1990 Robust Partial Parsing through Incremental, Multi-Level Processing: Rationales and Biases, TIS 1990, 17-19.

Miller, L.
1988 Natural Language Texts are Not Necessarily Grammatical and Unambiguous or Even Complete, Proceedings of the 26th Annual Meeting of the ACL, 167-168.

Milne, R.
1986 Resolving Lexical Ambiguity in a Deterministic Parser, Computational Linguistics 12: 1-12.

Nagao, K.
1990 Dependency Analyzer: A Knowledge-Based Approach to Structural Disambiguation, in: H. Karlgren (ed.), COLING 90: Papers presented to the 13th International Conference on Computational Linguistics, Vol. 2. 282-287.

Nakamura, M. & K. MaruyamaCT. KawabataCK. Shikano
1990 Neutral Network Approach to Word Category Prediction for English Texts, in: H.

Karlgren (ed.), COLING 90: Papers presented to the 13th International Conference on Computational Linguistics, Vol. 3. 213-218.

Newman, P.
1988 Combinatorial Disambiguation, Proceedings of the Second Conference on Applied Natural Language Processing, ACL. 243-252.

Niemikorpi, A.
1979 Automatic Data Processing in the Compilation of Word Lists, in: K. HäkkinenCF. Karlsson (eds.), Suomen kielitieteellisen yhdistyksen julkaisuja 2: 117-126.

Nunberg, G.
1990 The Linguistics of Punctuation. (CSLI Lecture Notes, Number 18.) Menlo Park, CA: CSLI.

Oostdijk, N.
1991 Corpus Linguistics and the Automatic Analysis of English. Amsterdam: Rodopi.
Oxford Advanced Learner's Dictionary of Current English. Fourth edition. 1989. Oxford: Oxford University Press.

Pennanen, E. V.
1980 On the function and behaviour of stress in English noun com¬pounds, English Studies 61: 252-263.

Phillips, J. D. & H. S. Thompson
1987 A Parser for Generalised Phrase Structure Grammars, in: N. Haddock, E. Klein & G. Morrill (eds.), Categorial Grammar, Unification Grammar and Parsing. (Edinburgh Working Papers in Cognitive Science, Vol. 1). Edinburgh: Centre for Cognitive Science, University of Edinburgh. 115-136.

Pollard, C. & I. A. Sag
1987 Information-Based Syntax and Semantics, Vol. 1. Fundamentals. (CSLI Lecture Notes, Number 13.) Menlo Park: CSLI.

Pulman, S., G. Russell, G. Ritchie & A. Black
1988 Computational morphology of English, Linguistics 26: 545-560.

Quirk, R. & S. GreenbaumCG. LeechCJ. Svartvik
1985 Comprehensive Grammar of the English Language. London: Longman.

Ravin, Y.
1990 Disambiguating and Interpreting Verb Definitions, Proceedings of the 28th Annual Meeting of the ACL, 260-267.

Rich, E. & S. LuperFoy
1988 An Architecture for Anaphora Resolution, Proceedings of the Second Conference on Applied Natural Language Processing, ACL, 18-24.

Richardson, S. & L. Braden Harder
1988 The Experience of Developing a Large-Scale Natural Language Text Processing System: Critique, Proceedings of the Second Conference on Applied Natural Language Processing, 195-202.

Ritchie, G., S. Pulman, A. Black & G. Russell
1987 A Computational Framework for Lexical Description, Computational Linguistics 13: 290-307.

Russell, G., S. Pulman, G. Ritchie & A. Black
1986 Dictionary and Morphological Analyser for English, Proceedings of the 11th International Conference on Computational Linguistics. 277-279.

Sampson, G.
1987a Probabilistic Models of Analysis, in: R. GarsideCG. LeechCG. Sampson (eds.), 16-29.
1987b Alternative grammatical coding systems, in: R. GarsideCG. LeechCG. Sampson (eds.), 165-183.
1992 Probabilistic Parsing, in: J. Svartvik (ed.), 425-447.
forthcoming: The need for grammatical stocktaking, to appear in N. Ostler (ed.), Proceedings of the 1992 Pisa Symposium on European Textual Corpora.

Saukkonen, P. & M. HaipusCA. NiemikorpiCH. Sulkala
1982 Suomen kielen homonyymeja, in: Språkhistoria och språkkontakt i Finland och Nord-Skandinavien. Studier tillägnade Tryggve Sköld 2.11.1982. (Kungl. Skytteanska Samfundets Förhandlingar 26.) Stockholm, 255-272.

Sharman, R.
1989 Observational Evidence for a Statistical Model of Language. (IBM UK Scientific Centre Report No. 205.) Winchester: IBM UK Scientific Centre.

Shieber, S.
1983 Sentence Disambiguation by a Shift-Reduce Parsing Technique, Proceedings of the 21st Annual Meeting of the ACL, 113-118.

Sinclair, J. M.

1992 The Automatic Analysis of Corpora, in: J. Svartvik (ed.), 379-397.

Slocum, J.
1988 Morphological Processing in the Nabu System, Proceedings of the Second Conference on Applied Natural Language Processing, ACL, 228-234.

Souter, C. & T. O'Donoghue
1991AProbabilistic Parsing in the COMMUNAL Project, in: S. Johansson & A. B. Stenström (eds.), 33-48.

Stallard, D.
1987 The Logical Analysis of Lexical Ambiguity, Proceedings of the 25th Annual Meeting of the ACL, 179-185.

Svartvik, J.
1987 Taking a new look at word class tags, in: W. Meijs (ed.), Corpus Linguistics and Beyond: Proceedings of the Seventh International Conference on English Language Research on Computerized Corpora. Amsterdam: Rodopi, 33-43.

Svartvik, J. (ed.)
1992 Directions in Corpus Linguistics: Proceedings of Nobel Symposium 82, Stockholm, 4-8 August 1991. (Trends in Linguistics: Studies and Monographs 65.) Berlin: Mouton de Gruyter.

Tapanainen, P.
1991 Äärellisinä automaatteina esitettyjen kielioppisääntöjen soveltaminen luonnollisen kielen jäsentäjässä. [Natural language parsing with finite-state syntactic rules. Unpublished Master's thesis. Department of Computer Science, University of Helsinki.]
1992 Äärellisiin automaatteihin perustuva luonnollisen kielen jäsennin [A finite-state parser of natural language. Licentiate (pre-doctoral) thesis. Department of Computer Science, University of Helsinki.]

Taylor, L., C. Grover & T. Briscoe
1989 The Syntactic Regularity of English Noun Phrases, Proceedings of the Fourth Conference of the European Chapter of the ACL, 256-263.

TIS 1990. Working Notes, March 27-29, 1990, Stanford University Sponsored by the American Association for Artificial Intelligence.

Tomita, M.
1984 Disambiguating Grammatically Ambiguous Sentences by Asking, Proceedings of the 10th International Conference on Computational Linguistics, 476-479.
1988 Graph-structured Stack and Natural Language Parsing, Proceedings of the 26th Annual Meeting of the ACL, 249-257.

Trishman R. & J. Sterling
1990 Towards Robust Natural Language Analysis, TIS 1990, 106-108.

Voutilainen, Atro
1989 Observations on the Maintenance of the English Master Lexicon. SIMPR Document SIMPR-RUCL-1989-5.4e.
1993 NPtool, a detector of English noun phrases, Proceedings of the Workshop on Very Large Corpora, June 22, 1993. Columbus, Ohio: Ohio State University.
1994 Designing a parsing grammar. (Publications of the Department of General Linguistics,

University of Helsinki.)

Voutilainen, A. & P. Tapanainen
1993 Ambiguity resolution in a reductionistic parser, Proceedings of EACL'93, 394-403.

Wilks, Y.
1985 Right Attachment and Preference Semantics, Proceedings of the Second Conference of the European Chapter of the ACL, 89-92.
1990 Combining Weak Methods in Large-Scale Text Processing, TIS 1990, 86-90.

Wouden, T. van der & D. Heylen
1988 Massive Disambiguation of Large Text Corpora with Flexible Categorial Grammar, Proceedings of the 12th International Conference on Computational Linguistics, Vol. 1. Budapest: John von Neumann Society for Computing Sciences, 694-698.