Chapter 3

Language Analysis and Understanding

3.1 Overview

Annie Zaenen[a] **& Hans Uszkoreit**[b]

[a] Rank Xerox Research Centre, Grenoble, France
[b] Deutsches Forschungszentrum für Künstliche Intelligenz
and Universität des Saarlandes, Saarbrücken, Germany

We understand larger textual units by combining our understanding of smaller ones. The main aim of linguistic theory is to show how these larger units of meaning arise out of the combination of the smaller ones. This is modeled by means of a grammar. Computational linguistics then tries to implement this process in an efficient way. It is traditional to subdivide the task into syntax and semantics, where syntax describes how the different formal elements of a textual unit, most often the sentence, can be combined and semantics describes how the interpretation is calculated.

In most language technology applications the encoded linguistic knowledge, i.e., the grammar, is separated from the processing components. The grammar consists of a lexicon, and rules that syntactically and semantically combine words and phrases into larger phrases and sentences. A variety of representation languages have been developed for the encoding of linguistic knowledge. Some of these languages are more geared towards conformity with formal linguistic theories, others are designed to facilitate certain processing models or specialized applications.

Several language technology products that are on the market today employ annotated phrase-structure grammars, grammars with several hundreds or thousands of rules describing different phrase types. Each of these rules is annotated by features and sometimes also by expressions in a programming language. When such grammars reach a certain size they become difficult to maintain, to extend and to reuse. The resulting systems might be sufficiently efficient for some applications but they lack the speed of processing needed for interactive systems (such as applications involving spoken input) or systems that have to process large volumes of texts (as in machine translation).

In current research, a certain polarization has taken place. Very simple grammar models are employed, e.g., different kinds of finite-state grammars that support highly efficient processing. Some approaches do away with grammars altogether and use statistical methods to find basic linguistic patterns. These approaches are discussed in section 3.7. On the other end of the scale, we find a variety of powerful linguistically sophisticated representation formalisms that facilitate grammar engineering. An exhaustive description of the current work in that area would be well beyond the scope of this overview. The most prevalent family of grammar formalisms currently used in computational linguistics, constraint based formalisms, is described in short in section 3.3. Approaches to lexicon construction inspired by the same view are described in section 3.4.

Recent developments in the formalization of semantics are discussed in section 3.5.

The computational issues related to different types of sentence grammars are discussed in section 3.6. Section 3.7 evaluates how successful the different techniques are in providing robust parsing results, and section 3.2 addresses issues raised when units smaller than sentences need to be parsed.

3.2 Sub-Sentential Processing[1]

Fred Karlsson[a] **& Lauri Karttunen**[b]

[a] University of Helsinki, Finland
[b] Rank Xerox Research Centre, Meylan, France

3.2.1 Morphological Analysis

In the last 10–15 years computational morphology has advanced further towards real-life applications than most other subfields of natural language processing. The quest for an efficient method for the analysis and generation of word-forms is no longer an academic research topic, although morphological analyzers still remain to be written for all but the commercially most important languages. This survey concentrates on the developments that have lead to large-scale practical analyzers, leaving aside many theoretically more interesting issues.

To build a syntactic representation of the input sentence, a parser must map each word in the text to some canonical representation and recognize its morphological properties. The combination of a surface form and its analysis as a canonical form and inflection is called a lemma.

The main problems are:

1. morphological alternations: the same morpheme may be realized in different ways depending on the context.

2. morphotactics: stems, affixes, and parts of compounds do not combine freely, a morphological analyzer needs to know what arrangements are valid.

A popular approach to 1 is the cut-and-paste method. The canonical form is derived by removing and adding letters to the end of a string. The best known ancestor of these systems is MITalk's DECOMP dating back to the 1960s (Allen, Hunnicutt, et al., 1987). The MORPHOGEN system (Petheroudakis, 1991) is a commercial toolkit for creating sophisticated cut-and-paste analyzers. In the MAGIC system (Schüller, Zierl, et al., 1993), cut-and-paste rules are applied in advance to produce the right allomorph for every allowed combination of a morpheme.

[1]By *sub-sentential processing* we mean *morphological analysis*, *morphological disambiguation*, and *shallow* (light) *parsing*.

The use of finite-state technology for automatic recognition and generation of word forms was introduced in the early 1980s. It is based on the observation (Johnson, 1972; Kaplan & Kay, 1994) that rules for morphological alternations can be implemented by finite-state transducers. It was also widely recognized that possible combinations of stems and affixes can be encoded as a finite-state network.

The first practical system incorporating these ideas is the two-level model (Koskenniemi, 1983; Karttunen, 1993; Antworth, 1990; Karttunen & Beesley, 1992; Ritchie, Russell, et al., 1992; Sproat, 1992). It is based on a set of linked letter trees for the lexicon and parallel finite-state transducers that encode morphological alternations. A two-level recognizer maps the surface string to a sequence of branches in the letter trees using the transducers and computes the lemma from information provided at branch boundaries.

In a related development during the 1980s, it was noticed that large spellchecking wordlists can be compiled to surprisingly small finite-state automata (Appel & Jacobson, 1988; Lucchesi & Kowaltowski, 1993). An automaton containing inflected word forms can be upgraded to a morphological analyzer, for example, by adding a code to the end of the inflected form that triggers some predefined cut-and-paste operation to produce the lemma. The RELEX lexicon format, developed at the LADL institute in Paris in the late 1980s, is this kind of combination of finite-state and cut-and-paste methods (Revuz, 1991; Roche, 1993).

Instead of cutting and pasting it at runtime, the entire lemma can be computed in advance and stored as a finite-state transducer whose arcs are labeled by a pair of forms (Tzoukermann & Liberman, 1990). The transducer format has the advantage that it can be used for generation as well as analysis. The number of nodes in this type of network is small but the number of arc-label pairs is very large as there is one symbol for each morpheme-allomorph pair.

A more optimal lexical transducer can be developed by constructing a finite-state network of lexical forms, augmented with inflectional tags, and composing it with a set of rule transducers (Karttunen & Beesley, 1992; Karttunen, 1993). The arcs of the network are labeled by a pair of individual symbols rather than a pair of forms. Each path through the network represents a lemma.

Lexical transducers can be constructed from descriptions containing any number of levels. This facilitates the description of phenomena that are difficult to describe within the constraints of the two-level model.

Because lexical transducers are bidirectional they are generally nondeterministic in both directions. If a system is only to be used for analysis, a simple finite-state network derived just for that purpose may be faster to operate.

3.2.2 Morphological Disambiguation

Word-forms are often ambiguous. Alternate analyses occur because of categorial homonymy, accidental clashes created by morphological alternations, multiple functions of affixes, or uncertainty about suffix and word boundaries. The sentential context normally decides which analysis is appropriate. This is called disambiguation.

There are two basic approaches to disambiguation: rule-based and probabilistic. Rule-based taggers Greene and Rubin (1971); Karlsson, Voutilainen, et al. (1994) typically leave some of the ambiguities unresolved but make very few errors; statistical taggers generally provide a fully disambiguated output but they have a higher error rate.

Probabilistic (stochastic) methods for morphological disambiguation have been dominant since the early 1980s. One of the earliest is Constituent-Likelihood Automatic Word-tagging System (CLAWS), developed for tagging the Lancaster-Oslo/Bergen Corpus of British English in 1978–1983 (Marshall, 1983).

CLAWS uses statistical optimization over n-gram probabilities to assign to each word one of 133 part-of-speech tags. The success rate of CLAWS2 (an early version) is 96–97% (Garside, Leech, et al., 1987). An improved version, CLAWS4, is used for tagging the 100-million-word British National Corpus (Leech, Garside, et al., 1994). It is based on a tagset of 61 tags. Similar success rates as for CLAWS, i.e., 95–99%, have been reported for English in many studies, e.g., Church (1988); De Rose (1988).

Most of the stochastic systems derive the probabilities from a handtagged training corpus. Probabilistic taggers based on a Hidden Markov Model can also be trained on an untagged corpus with a reported success rate around 96% for English (Kupiec, 1992; Cutting, Kupiec, et al., 1992; Elworthy, 1993).

The accuracy of probabilistic taggers for English has remained relatively constant for the past ten years under all of the various methods. This level has recently been surpassed by a rule-based disambiguator (Karlsson, Voutilainen, et al., 1994; Voutilainen, 1994). The system consists of some 1,100 disambiguation rules written in Karlsson's Constraint Grammar formalism. The accuracy in running text is 99.7% if 2–6% of the words are left with the most recalcitrant morphological ambiguities pending. Standard statistical methods can be applied to provide a fully disambiguated output.

3.2.3 Shallow Parsing

We use the term *shallow syntax* as a generic term for analyses that are less complete than the output from a conventional parser. The output from a shallow analysis is not a phrase-structure tree. A shallow analyzer may identify some phrasal constituents, such as noun phrases, without indicating their internal structure and their function in the sentence. Another type of shallow analysis identifies the functional role of some of the words, such as the main verb, and its direct arguments.

Systems for shallow parsing normally work on top of morphological analysis and disambiguation. The basic purpose is to infer as much syntactic structure as possible from the lemmata, morphological information, and word order configuration at hand. Typically shallow parsing aims at detecting phrases and basic head/modifier relations. A shared concern of many shallow parsers is the application to large text corpora. Frequently partial analyses are allowed if the parser is not potent enough to resolve all problems.

Church (1988) has designed a stochastic program for locating simple noun phrases which are identified by inserting appropriate brackets, [...]. Thus, a phrase such as *a former top aide* would be bracketed as a noun phrase on the basis of the information available in separately coded morphological tags, in the following example: AT (article), AP (attributive adjective), and NN (common singular noun): [a/AT former/AP top/NN aide/NN]. Hindle's parser *Fidditch* (Hindle, 1989) provides an annotated surface structure, especially phrase structure trees. It has been applied to millions of words.

The IBM/Lancaster approach to syntax is based on probabilistic parsing methods which are tested and refined using as reference corpus a manually bracketed set of sentences (Black, Garside, et al., 1993). These sentences are partly *skeleton parsed*, i.e., clear constituents are bracketed but difficult problems may be left open.

The PEG (PLNLP English Grammar) is a broad-coverage system for lexical, morphological, and syntactic analysis of running English text (Jensen & Heidorn, 1993). It provides approximate parses if all requisite information is not available. Rules are available for ranking alternative parses. For many sentences, PEG provides thorough syntactic analyses.

The TOSCA parser for English created in Nijmegen (Oostdijk, 1991) is representative of shallow parsing in the sense that rule formulation is based on extensive corpus study.

Constraint Grammar syntax stamps each word in the input sentence with a surface syntactic tag. 85–90 English words out of 100 get a unique syntactic tag, 2% are erroneous. The system was used for the morphosyntactic tagging of the 200-million-word Bank of English corpus (Järvinen, 1994).

Koskenniemi (1990) has designed a surface syntactic parser where the syntactic constraints are applied in parallel and implemented as finite-state automata. One central idea is to have most of the morphological disambiguation done by the syntactic constraints proper.

3.2.4 Future Directions

There is a need for automatic or semiautomatic discovery procedures that infer rules and rule sets for morphological analyzers from large corpora. Such procedures would make it possible to partially automate the construction of morphological analyzers.

Much work remains to be done on interfacing morphological descriptions with lexicon, syntax, and semantics in a maximally informative way. This presupposes a global view of how the various processing components relate to one another. One current line of research concerns the integration of shallow syntactic parsers with *deeper* syntactic approaches. A shallow parser used as a kind of preprocessor paves the way for a parser addressing the most recalcitrant syntactic structures such as coordination and ellipsis, thus making the task of *deeper* parsers more manageable, e.g., by reducing the number of ambiguities.

Work remains to be done on a general theory for combining rule-based approaches and stochastic approaches in a principled way. Both are needed in the task of tagging (parsing) unrestricted running text. Their respective reasonable tasks and order of application are not yet clearly understood.

Much work is currently being done on refining the methodology for testing candidate rules on various types of corpora. The importance of having available flexible methods for corpus testing is growing.

3.3 Grammar Formalisms

Hans Uszkoreit[a] **& Annie Zaenen**[b]

[a] Deutsches Forschungzentrum für Künstliche Intelligenz, Saarbrücken, Germany
and Universität des Saarlandes, Saarbrücken, Germany
[b] Rank Xerox Research Centre, Grenoble, France

A very advanced and wide-spread class of linguistic formalisms are the so-called constraint-based grammar formalisms which are also often subsumed under the term unification grammars. They go beyond many earlier representation languages in that they have a clean denotational semantics that permits the encoding of grammatical knowledge independent from any specific processing algorithm. Since these formalisms are currently used in a large number of systems, we will a provide a brief overview of their main characteristics.

Among the most used, constraint-based grammar models are Functional Unification Grammar (FUG) (Kay, 1984) Head-Driven Phrase-Structure Grammar (HPSG) (Pollard & Sag, 1994) Lexical Functional Grammar (LFG) (Bresnan, 1982), Categorial Unification Grammar (CUG) (Haddock, Klein, et al., 1987; Karttunen, 1989; Uszkoreit, 1986), and Tree Adjunction Grammar (TAG) (Joshi & Schabes, 1992). For these or similar grammar models, powerful formalisms have been designed and implemented that are usually employed for both grammar development and linguistic processing, e.g, LFG (Bresnan, 1982), PATR (Shieber, Uszkoreit, et al., 1983), ALE (Carpenter, 1992), STUF (Bouma, Koenig, et al., 1988), ALEP (Alshawi, Arnold, et al., 1991), CLE (Alshawi, 1992) TDL (Krieger & Schaefer, 1994) TFS (Emele & Zajac, 1990).

One essential ingredient of all these formalisms is complex formal descriptions of grammatical units (words, phrases, sentences) by means of sets of attribute-value pairs, so called feature terms. These feature terms can be nested, i.e., values can be atomic symbols or feature terms. Feature terms can be underspecified. They may contain equality statements expressed by variables or coreference markers. The formalisms share a uniform operation for the merging and checking of grammatical information, which is commonly referred to as unification.

The formalisms differ in other aspects. Some of them are restricted to feature terms with simple unification (PATR). Others employ more powerful data types such as disjunctive terms, functional constraints, or sets. Most formalisms combine phrase-structure rules or other mechanisms for building trees with the feature-term component of the language (LFG, TAG, TDL). A few formalisms incorporate the phrase-structure information into the feature terms (HPSG, TFS).

Some frameworks use inheritance type systems (HPSG, TFS, TDL, ALE). Classes of feature terms belong to types. The types are partially ordered in a tree or in a (semi) lattice. The type hierarchy determines for every type from which other types attributes and values are inherited, which attributes are allowed and needed for a well-formed feature term of the type, which types of values these attributes need, and with which other types the type can be conjoined by means of unification.

If the feature system allows complex features, attribute-value pairs in which values may again be feature-terms, this recursion can be constrained by recursive type definitions. In fact, all of grammatical recursion can be elegantly captured by such recursive types. In the extreme, the entire linguistic derivation (parsing, generation) can be construed as type deduction (HPSG, TFS).

The strength of unification grammar formalisms lies in the advantages they offer for grammar engineering. Experience has proven that large grammars can be specified, but that their development is extremely labour-extensive. Currently no methods exist for efficient distributed grammar engineering. This constitutes a serious bottleneck in the development of language technology products. The hope is that the new class of declarative formalisms will greatly facilitate linguistic engineering and thus speed up the entire development cycle. There are indications that seem to support this expectation. For some sizable grammars written in unification grammar formalisms, the development time was four years or less (TUG, CLE, TDL), whereas the development of large annotated phrase structure grammars had taken 8–12 years.

Another important issue in grammar engineering is the reusability of grammars. The more a grammar is committed to a certain processing model, the less are the chances that it can be adapted to other processing models or new application areas. Although scientists are still far from converging on a uniform representation format, the declarative formulation of grammar greatly facilitates porting of such grammars from one formalism to the other. Recent experiments in grammar porting seem to bear out these expectations.

It is mainly because of their expected advantages for grammar engineering that several unification formalisms have been developed or are currently used in industrial laboratories. Almost all ongoing European Union-funded language technology projects involving grammar development have adopted unification grammar formalisms.

3.4 Lexicons for Constraint-Based Grammars

Antonio Sanfilippo

Sharp Industries of Europe, Oxford, UK

The intelligent processing of natural language for real world applications requires lexicons which provide rich information about morphological, syntactic and semantic properties of words, are well structured and can be efficiently implemented (Briscoe, 1992). These objectives can be achieved by developing tools which facilitate the acquisition of lexical information from machine readable dictionaries and text corpora, as well as database technologies and theories of word knowledge offering an encoding of the information acquired which is desirable for NLP purposes. In the last decade, there has been a growing tendency to use unification-based grammar formalisms (Kay, 1979; Kaplan & Bresnan, 1982; Pollard & Sag, 1987; Pollard & Sag, 1994; Zeevat, Klein, et al., 1987) to carry out the task of building such lexicons. These grammar formalisms encode lexical descriptions as feature structures, with inheritance and unification as the two basic operations relating these structures to one another. The use of inheritance and unification is appealing from both engineering and linguistic points of view as these operations can be formalized in terms of lattice-theoretic notions (Carpenter, 1992) which are amenable to efficient implementation and are suitable to express the hierarchical nature of lexical structure. Likewise, feature structures have a clear mathematical and computational interpretation and provide an ideal data structure to encode complex word knowledge information.

Informally, a feature structure is a set of attribute-value pairs, where values can be atomic or feature structures themselves, providing a partial specification of words, affixes and phrases. Inheritance makes it possible to arrange feature structures into a subsumption hierarchy so that information which is repeated across sets of word entries needs only specifying once (Flickinger, 1987; Pollard & Sag, 1987; Sanfilippo, 1993). For example, properties which are common to all verbs (e.g., part of speech, presence of a subject) or subsets of the verb class (presence of a direct object for verbs such as *amuse* and *put*; presence of an indirect object for verbs such as *go* and *put*) can be defined as templates. Unification provides the means for integrating inherent and inherited specifications of feature structure descriptions.

In general, unification is monotonic: all information, whether inherently specified or inherited, is preserved. Consequently, a valid lexical entry can never contain conflicting values. Unification thus provides a way to perform a consistency check on lexical descriptions. For example, the danger of inadvertently assigning distinct orthographies or parts of speech to the same word entry is easily avoided as the unification of incompatible information leads to failure. An even more stringent regime of grammar checking has recently been made available through the introduction of *typed* feature structures (Carpenter, 1992). Through typing, feature structures can be arranged into a closed hierarchy so that two feature structures unify only if their types have a common subtype. Typing is also used to specify exactly which attributes are appropriate for a given feature structure so that arbitrary extensions of feature structures are easily eschewed.

A relaxation of monotonicity, however, is sometimes useful in order to capture regularities across the lexicon. For example, most irregular verbs in English follow the same inflectional patterns as regular verbs with respect to present and gerundive forms, while differing in the simple past and/or past participle. It would therefore be convenient to state that all verbs inherit the same regular morphological paradigm by default and then let the idiosyncratic specifications of irregular verbs override inherited information which is incompatible.

Default inheritance in the lexicon is desirable to achieve compactness and simplicity in expressing generalizations about various aspects of word knowledge (Flickinger, 1987; Gazdar, 1987), but it can be problematic if used in an unconstrained manner. For example, it is well known that multiple default inheritance can lead to situations which can only be solved *ad hoc* or nondeterministically when conflicting values are inherited from the parent nodes (Touretzsky, Horty, et al., 1987). Although a number of proposals have been made to solve these problems, a general solution is still not available so that the use of default inheritance must be tailored to specific applications.

Another difficult task in lexicon implementation, perhaps the most important with regard to grammar processing, concerns the treatment of lexical ambiguity. Lexical ambiguity can be largely related to our ability to generate appropriate uses of words in context by manipulation of semantic and/or syntactic properties of words. For example, *accord* is synonymous with either *agree* or *give/grant* depending on its valency, *move* can also be interpreted as a psychological predicate when used transitively with a sentient direct object, and *enjoy* can take either a noun or verb phrase complement when used in the *experience* sense:

a Senator David Lock's bill does not accord State benefits to illegal aliens
 They accorded him a warm welcome

b The two alibis do not accord
 Your alibi does not accord with his

c Her sadness moves him

d John enjoys $\left\{ \begin{array}{l} \text{the book} \\ \text{reading the book} \end{array} \right\}$

Although the precise mechanisms which govern lexical knowledge are still largely unknown, there is strong evidence that word sense extensibility is not arbitrary (Atkins

& Levin, 1992; Pustejovsky, 1991; Pustejovsky, 1994; Ostler & Atkins, 1992). For example, the amenability of a verb such as *move* to yield either a movement or psychological interpretation can be generalized to most predicates of caused motion (e.g., *agitate, crash, cross, lift, strike, sweep, unwind*). Moreover, the metonymical and metaphoric processes which are responsible for polysemy appear to be subject to crosslinguistic variation. For example, the "meat vs. animal" alternation that is found in English—viz. *feed the lamb* vs. *eat lamb*—is absent in Eskimo (Nunberg & Zaenen, 1992) and is less productive in Dutch where nominal compounding is often used instead, e.g., *lam* vs. *lamvlees*.

Examples of this sort show that our ability to extend word use in context is often systematic or conventionalized. Traditional approaches to lexical representation assume that word use extensibility can be modeled by exhaustively describing the meaning of a word through closed enumeration of its senses. Word sense enumeration provides highly specialized lexical entries, but

- it fails to make explicit regularities about word sense extensibility which are necessary in promoting compactness in lexical description,
- it is at odds with our ability to create new word uses in novel contexts, and
- it generates massive lexical ambiguity.

Consequently, several attempts have been made to develop a more dynamic approach to lexical specification which provides a principled treatment of polysemy and can be used to model creative aspects of word use. For example, Pustejovsky (1991); Pustejovsky (1994) and Pustejovsky and Boguraev (1993) propose an integrated multilayered representation of word meaning which incorporates salient aspects of world knowledge, e.g., purpose, origin, form and constituency properties are specified for object-denoting nominals. This makes it possible to conflate different uses of the same word into a single *meta-entry* which can be extended to achieve contextual congruity using lexical rules (Copestake & Briscoe, 1992). Equivalent results can be obtained using abductive reasoning to generate different word senses from polysemic lexical representations (Hobbs, Stickel, et al., 1993). The use of lexical rules or abductive reasoning provide a principled alternative to word sense enumeration in the treatment of polysemy and can be made to cater for novel uses of words. However, it is not clear whether these practices can address the question of lexical ambiguity efficiently as there is no known general control regime on lexical rules or abductive reasoning which would deterministically restricts polysemic expansion without preempting the generation of possible word uses. A promising alternative is to use contextual information to guide sense extension. For example Sanfilippo, Benkerimi, et al. (1994); Sanfilippo (1995) propose that polysemy be expressed as lexical polymorphism within a Typed Feature Structure formalism by assigning to an ambiguous word entry a lexical type with subtype extensions describing all admissible uses of the word. Lexical ambiguities can then be solved deterministically by using syntactic and semantic contextual information during language processing to ground underspecified word entries.

3.5 Semantics[2]

Stephen G. Pulman

SRI International, Cambridge, UK
and University of Cambridge Computer Laboratory, Cambridge, UK

3.5.1 Basic Notions of Semantics

A perennial problem in semantics is the delineation of its subject matter. The term *meaning* can be used in a variety of ways, and only some of these correspond to the usual understanding of the scope of linguistic or computational semantics. We shall take the scope of semantics to be restricted to the literal interpretations of sentences in a context, ignoring phenomena like irony, metaphor, or *conversational implicature* (Grice, 1975; Levinson, 1983).

A standard assumption in computationally oriented semantics is that knowledge of the meaning of a sentence can be equated with knowledge of its truth conditions: that is, knowledge of what the world would be like if the sentence were true. This is not the same as knowing whether a sentence is true, which is (usually) an empirical matter, but knowledge of truth conditions is a prerequisite for such verification to be possible. *Meaning as truth conditions* needs to be generalized somewhat for the case of imperatives or questions, but is a common ground among all contemporary theories, in one form or another, and has an extensive philosophical justification, e.g., Davidson (1969); Davidson (1973).

A semantic description of a language is some finitely stated mechanism that allows us to say, for each sentence of the language, what its truth conditions are. Just as for grammatical description, a semantic theory will characterize complex and novel sentences on the basis of their constituents: their meanings, and the manner in which they are put together. The basic constituents will ultimately be the meanings of words and morphemes. The modes of combination of constituents are largely determined by the syntactic structure of the language. In general, to each syntactic rule combining some sequence of child constituents into a parent constituent, there will correspond some semantic operation combining the meanings of the children to produce the meaning of the parent.

A corollary of knowledge of the truth conditions of a sentence is knowledge of what

[2]This survey draws in part on material prepared for the European Commission LRE Project 62-051, *FraCaS: A Framework for Computational Semantics*. I am grateful to the other members of the project for their comments and contributions.

inferences can be legitimately drawn from it. Valid inference is traditionally within the province of logic (as is truth) and mathematical logic has provided the basic tools for the development of semantic theories. One particular logical system, first order predicate calculus (FOPC), has played a special role in semantics (as it has in many areas of computer science and artificial intelligence). FOPC can be seen as a small model of how to develop a rigorous semantic treatment for a language, in this case an artificial one developed for the unambiguous expression of some aspects of mathematics. The set of sentences or well formed formulae of FOPC are specified by a grammar, and a rule of semantic interpretation is associated with each syntactic construct permitted by this grammar. The interpretations of constituents are given by associating them with set-theoretic constructions (their *denotation*) from a set of basic elements in some universe of discourse. Thus for any of the infinitely large set of FOPC sentences we can give a precise description of its truth conditions, with respect to that universe of discourse. Furthermore, we can give a precise account of the set of valid inferences to be drawn from some sentence or set of sentences, given these truth conditions, or (equivalently, in the case of FOPC) given a set of rules of inference for the logic.

3.5.2 Practical Applications of Semantics

Some natural language processing tasks (e.g., message routing, textual information retrieval, translation) can be carried out quite well using statistical or pattern matching techniques that do not involve semantics in the sense assumed above. However, performance on some of these tasks improves if semantic processing is involved. (Not enough progress has been made to see whether this is true for all of the tasks).

Some tasks, however, cannot be carried out at all without semantic processing of some form. One important example application is that of database query, of the type chosen for the Air Travel Information Service (ATIS) task (DARPA, 1989). For example, if a user asks, *"Does every flight from London to San Francisco stop over in Reykyavik?"* then the system needs to be able to deal with some simple semantic facts. Relational databases do not store propositions of the form *every X has property P* and so a logical inference from the meaning of the sentence is required. In this case, *every X has property P* is equivalent to *there is no X that does not have property P* and a system that knows this will also therefore know that the answer to the question is *no* if a non-stopping flight is found and *yes* otherwise.

Any kind of generation of natural language output (e.g., summaries of financial data, traces of KBS system operations) usually requires semantic processing. Generation requires the construction of an appropriate meaning representation, and then the production of a sentence or sequence of sentences which express the same content in a way that is natural for a reader to comprehend, e.g., McKeown, Kukich, et al. (1994). To illustrate, if a database lists a 10 a.m. flight from London to Warsaw on the 1st–14th, and 16th–30th of November, then it is more helpful to answer the question *What days does that flight go?* by *Every day except the 15th* instead of a list of 30 days of the month. But to do this the system needs to know that the semantic representations of the two propositions are equivalent.

3.5.3 Development of Semantic Theory

It is instructive, though not historically accurate, to see the development of contemporary semantic theories as motivated by the deficiencies that are uncovered when one tries to take the FOPC example further as a model for how to do natural language semantics. For example, the technique of associating set theoretic denotations directly with syntactic units is clear and straightforward for the artificial FOPC example. But when a similar programme is attempted for a natural language like English, whose syntax is vastly more complicated, the statement of the interpretation clauses becomes in practice extremely baroque and unwieldy, especially so when sentences that are semantically but not syntactically ambiguous are considered (Cooper, 1983). For this reason, in most semantic theories, and in all computer implementations, the interpretation of sentences is given indirectly. A syntactically disambiguated sentence is first translated into an expression of some artificial logical language, where this expression in its turn is given an interpretation by rules analogous to the interpretation rules of FOPC. This process factors out the two sources of complexity whose product makes direct interpretation cumbersome: reducing syntactic variation to a set of common semantic constructs; and building the appropriate set-theoretical objects to serve as interpretations.

The first large scale semantic description of this type was developed by Montague (1973). Montague made a further departure from the model provided by FOPC in using a more powerful logic (*intensional logic*) as an intermediate representation language. All later approaches to semantics follow Montague in using more powerful logical languages: while FOPC captures an important range of inferences (involving, among others, words like *every*, and *some* as in the example above), the range of valid inference patterns in natural languages is far wider. Some of the constructs that motivate the use of richer logics are sentences involving concepts like *necessity* or *possibility* and *propositional attitude* verbs like *believe* or *know*, as well as the inference patterns associated with other English quantifying expressions like *most* or *more than half*, which cannot be fully captured within FOPC (Barwise & Cooper, 1981).

For Montague, and others working in frameworks descended from that tradition (among others, Partee, e.g., Partee, 1986, Krifka, e.g., Krifka, 1989, and Groenendijk and Stokhof, e.g., Groenendijk & Stokhof, 1984; Groenendijk & Stokhof, 1991a) the intermediate logical language was merely a matter of convenience which could in principle always be dispensed with provided the *principle of compositionality* was observed. (I.e., *The meaning of a sentence is a function of the meanings of its constituents*, attributed to Frege, (Frege, 1892)). For other approaches, (e.g., Discourse Representation Theory, Kamp, 1981) an intermediate level of representation is a necessary component of the theory, justified on psychological grounds, or in terms of the necessity for explicit reference to representations in order to capture the meanings of, for example, pronouns or other referentially dependent items, elliptical sentences or sentences ascribing mental states (beliefs, hopes, intentions). In the case of computational implementations, of course, the issue of the dispensability of representations does not arise: for practical purposes, some kind of meaning representation is a *sine qua non* for any kind of computing.

3.5.4 Discourse Representation Theory

Discourse Representation Theory (DRT) (Kamp, 1981; Kamp & Reyle, 1993), as the name implies, has taken the notion of an intermediate representation as an indispensable theoretical construct, and, as also implied, sees the main unit of description as being a discourse rather than sentences in isolation. One of the things that makes a sequence of sentences constitute a discourse is their connectivity with each other, as expressed through the use of pronouns and ellipsis or similar devices. This connectivity is mediated through the intermediate representation, however, and cannot be expressed without it. The kind of example that is typically used to illustrate this is the following:

A computer developed a fault.

A simplified first order representation of the meaning of this sentence might be:

exists(X,computer(X) and develop_a_fault(X))

There is a computer X and X developed a fault. This is logically equivalent to:

not(forall(X,not(computer(X) and develop_a_fault(X))))

It isn't the case that every computer didn't develop a fault. However, whereas the first sentence can be continued thus:

A computer developed a fault.
It was quickly repaired.

—its logically equivalent one cannot be:

It isn't the case that every computer didn't develop a fault.
It was quickly repaired.

Thus the form of the representation has linguistic consequences. DRT has developed an extensive formal description of a variety of phenomena such as this, while also paying careful attention to the logical and computational interpretation of the intermediate representations proposed. Kamp and Reyle (1993) contains detailed analyses of aspects of noun phrase reference, propositional attitudes, tense and aspect, and many other phenomena.

3.5.5 Dynamic Semantics

Dynamic semantics (e.g., Groenendijk & Stokhof, 1991a; Groenendijk & Stokhof, 1991b) takes the view that the standard truth-conditional view of sentence meaning deriving from the paradigm of FOPC does not do sufficient justice to the fact that uttering a sentence changes the context it was uttered in. Deriving inspiration in part from work on the semantics of programming languages, dynamic semantic theories have developed several variations on the idea that the meaning of a sentence is to be equated with the changes it makes to a context.

Update semantics (e.g., Veltman, 1985; van Eijck & de Vries, 1992) approaches have been developed to model the effect of asserting a sequence of sentences in a particular context. In general, the order of such a sequence has its own significance. A sequence like:

Someone's at the door. Perhaps it's John. It's Mary!

is coherent, but not all permutations of it would be:

Someone's at the door. It's Mary. Perhaps it's John.

Recent strands of this work make connections with the artificial intelligence literature on truth maintenance and belief revision (e.g Gärdenfors, 1990).

Dynamic predicate logic (Groenendijk & Stokhof, 1991a; Groenendijk & Stokhof, 1990) extends the interpretation clauses for FOPC (or richer logics) by allowing assignments of denotations to subexpressions to carry over from one sentence to its successors in a sequence. This means that dependencies that are difficult to capture in FOPC or other non-dynamic logics, such as that between *someone* and *it* in:

Someone's at the door. It's Mary.

can be correctly modeled, without sacrificing any of the other advantages that traditional logics offer.

3.5.6 Situation Semantics and Property Theory

One of the assumptions of most semantic theories descended from Montague is that information is total, in the sense that in every situation, a proposition is either true or it is not. This enables propositions to be identified with the set of situations (or *possible worlds*) in which they are true. This has many technical conveniences, but is descriptively incorrect, for it means that any proposition conjoined with a tautology (a logical truth) will remain the same proposition according to the technical definition. But this is clearly wrong: *all cats are cats* is a tautology, but *The computer crashed*, and *The computer crashed and all cats are cats* are clearly different propositions (reporting the first is not the same as reporting the second, for example).

Situation theory (Barwise & Perry, 1983) has attempted to rework the whole logical foundation underlying the more traditional semantic theories in order to arrive at a satisfactory formulation of the notion of a *partial state of the world* or situation, and in turn, a more satisfactory notion of proposition. This reformulation has also attempted to generalize the logical underpinnings away from previously accepted restrictions (for example, restrictions prohibiting sets containing themselves, and other apparently paradoxical notions) in order to be able to explore the ability of language to refer to itself in ways that have previously resisted a coherent formal description (Barwise & Etchemendy, 1987).

Property theory (Turner, 1988; Turner, 1992) has also been concerned to rework the logical foundations presupposed by semantic theory, motivated by similar phenomena.

In general, it is fair to say that, with a few exceptions, the contribution of dynamic semantics, situation theory, and property theory has so far been less in the analysis of new semantic phenomena than in the exploration of more cognitively and computationally plausible ways of expressing insights originating within Montague-derived approaches. However, these new frameworks are now making it possible to address data that resisted any formal account by more traditional theories.

3.5.7 Implementations

Whereas there are beginning to be quite a number of systems displaying wide syntactic coverage, there are very few that are able to provide corresponding semantic coverage. Almost all current large scale implementations of systems with a semantic component are inspired to a greater or lesser extent by the work of Montague (e.g., Bates, Bobrow, et al., 1994; Allen, Schubert, et al., 1995; Alshawi, 1992). This reflects the fact that the majority of descriptive work by linguists is expressed within some form of this framework, and also the fact that its computational properties are better understood.

However, Montague's own work gave only a cursory treatment of a few context-dependent phenomena like pronouns, and none at all of phenomena like ellipsis. In real applications, such constructs are very common and all contemporary systems supplement the representations made available by the base logic with constructs for representing the meaning of these context-dependent constructions. It is computationally important to be able to carry out at least some types of processing directly with these *underspecified representations*: i.e., representations in which the contextual contribution to meaning has not yet been made explicit, in order to avoid a combinatorial explosion of potential ambiguities. One striking motivation for underspecification is the case of quantifying noun phrases, for these can give rise to a high degree of ambiguity if treated in Montague's fashion. For example, *every keyboard is connected to a computer* is interpretable as involving either a single computer or a possibly different one for each keyboard, in the absence of a context to determine which is the plausible reading: sentences do not need to be much more complex for a large number of possibilities to arise.

One of the most highly developed of the implemented approaches addressing these issues is the *quasi-logical form* developed in the Core Language Engine (CLE) (Alshawi, 1990; Alshawi, 1992) a representation which allows for meanings to be of varying degrees of independence of a context. This makes it possible for the same representation to be used in applications like translation, which can often be carried out without reference to context, as well as in database query, where the context-dependent elements must be resolved in order to know exactly which query to submit to the database. The ability to operate with underspecified representations of this type is essential for computational tractability, since the task of spelling out all of the possible alternative fully specified interpretations for a sentence and then selecting between them would be computationally intensive even if it were always possible in practice.

3.5.8 Future Directions

Currently, the most pressing needs for semantic theory are to find ways of achieving wider and more robust coverage of real data. This will involve progress in several directions: (i) Further exploration of the use of underspecified representations so that some level of semantic processing can be achieved even where complete meaning representations cannot be constructed (either because of lack of coverage or inability to carry out contextual resolution). (ii) Closer cooperation with work in lexicon construction. The tradition in semantics has been to assume that word meanings can by and large simply be *plugged in* to semantic structures. This is a convenient and largely correct assumption when dealing with structures like *every X is P*, but becomes less tenable as more complex phenomena are examined. However, the relevant semantic properties of individual words or groups of words are seldom to be found in conventional dictionaries and closer cooperation between semanticists and computationally aware lexicographers is required. (iii) More integration between sentence or utterance level semantics and theories of text or dialogue structure. Recent work in semantics has shifted emphasis away from the purely sentence-based approach, but the extent to which the interpretations of individual sentences can depend on dialogue or text settings, or on the goals of speakers, is much greater than had been suspected.

3.6 Sentence Modeling and Parsing

Fernando Pereira

AT&T Bell Labs, Murray Hill, New Jersey, USA

The complex hidden structure of natural-language sentences is manifested in two different ways: *predictively*, in that not every constituent (for example, word) is equally likely in every context, and *evidentially*, in that the information carried by a sentence depends on the relationships among the constituents of the sentence. Depending on the application, one or the other of those two facets may play a dominant role. For instance, in *language modeling* for large-vocabulary connected speech recognition, it is crucial to distinguish the relative likelihoods of possible continuations of a sentence prefix, since the acoustic component of the recognizer may be unable to distinguish reliably between those possibilities just from acoustic evidence. On the other hand, in applications such as machine translation or text summarization, relationships between sentence constituents, such as that a certain noun phrase is the direct object of a certain verb occurrence, are crucial evidence in determining the correct translation or summary. *Parsing* is the process of discovering *analyses* of sentences, that is, consistent sets of relationships between constituents that are judged to hold in a given sentence, and, concurrently, what the constituents are, since constituents are typically defined inductively in terms of the relationships that hold between their parts.

It would not be possible to model or parse sentences without mechanisms to compute the properties of larger constituents from the properties of their parts, appropriately defined, since the properties of new sentences, which are unlikely to have been seen before, can only be inferred from knowledge of how their parts participate in the sentences we have observed previously. While this point may seem obvious, it has deep consequences both in language modeling and parsing. Any language model or parser must include a generative mechanism or *grammar* that specifies how sentences are built from their parts, and how the information associated to the sentence derives from the information associated to its parts. Furthermore, to be able to cope with previously unseen sentences, any such system must involve *generalization* with respect to the data from which the language model or parser was developed.

3.6.1 Grammars and Derivations

It is useful to think of the grammar in a language model or parser as the specification of a configuration space in which the configurations represent stages of constituent combination, and transitions between configurations describe how constituents are combined in deriving larger constituents. For instance, the configurations may be the states of a finite-state machine and the transitions represent how words may be appended to the end of a sentence prefix. In the more complex case of phrase-structure grammars, configurations represent sequences of phrases (*sentential forms*), and transitions the possible combinations of adjacent phrases into larger phrases. A *derivation* of a sentence according to the grammar is a path in the configuration space of the grammar from an initial configuration to a final configuration in which all the elementary constituents are consumed. (We will call such elementary constituents *words* in what follows, although the informal notion of word may not correspond to the appropriate technical definition, especially in languages with complex morphology.)

Even with respect to procedural parsers and language models which are not normally described as containing a grammar, such as certain deterministic (Marcus, 1980; Hindle, 1993) and probabilistic parsers (Black, Jelinek, et al., 1993; Magerman & Marcus, 1991), it is useful to identify the implicit grammar defined by the possible derivation moves which the parser can use under the control of its control automaton. For instance, in a parser based on a pushdown automaton such as a shift-reduce parser (Shieber, 1983; Pereira, 1985), the grammar corresponds to the possible transitions between stack and input configurations, while the automaton's finite-state control determines which transitions are actually used in a derivation.

3.6.2 Precision versus Coverage

The choice of a grammar for a particular parsing or language modeling application involves two conflicting requirements: *precision* and *coverage*. By precision we mean how well the grammar encodes constraints on possible sentences and possible meaningful relationships carried by those sentences. By coverage we mean what proportion of actual sentences have a reasonable derivation in the grammar. We are interested in precision because a more precise grammar is better able to rule out bad sentence hypotheses in predictive tasks and bad meaningful relationship hypotheses in evidential tasks. We are interested in coverage so that our systems will handle appropriately a wide range of actual spoken or written language. But as we increase precision by encoding more constraints in the grammar, we tend to lose coverage of those actual sentences that violate some of the constraints while being still acceptable to language users. The reason for the problem is that the most powerful constraints are idealizations of the actual performance of language users. The tension between precision and coverage is central to the design tradeoffs we will now survey.

3.6.3 Search Space and Search Procedure

We see thus a sentence parser or language model as consisting of a grammar and a *search procedure* which, given an input sentence, will apply transitions specified by the grammar to construct derivations of the sentence and associated analyses. In cases where the input sentence is uncertain, such as speech recognition, we may further generalize the above picture to a simultaneous search of the configuration space for the grammar and of a space of *sentence hypotheses*, represented for instance as a *word lattice* (Murveit, Butzberger, et al., 1993).

The computational properties of parsers and language models depend on two main factors: the *structure* of the search space induced by the grammar, and the *exhaustiveness* of the search procedure.

Search Space Structure

Under the above definition of grammar, transitions from a configuration may have to take into account the whole configuration. However, most useful grammar classes have some degree of *locality* in that transitions involve only on a bounded portion of a configuration. In that case, derivations can be factored into sub-derivations concerning independent parts of configurations, allowing independent sub-derivations to be shared among derivations, for potentially exponential reductions in the size of the search space. The search algorithm can then tabulate each sub-derivation and reuse it in building any derivation that shares that sub-derivation.[3] Such *tabular* algorithms are widely used in parsing and language modeling with appropriate kinds of grammars (Younger, 1967; Kay, 1986; Earley, 1970; Lang, 1974; Graham, Harrison, et al., 1980; Tomita, 1987), because they support exhaustive search algorithms with polynomial space and time with respect to sentence length. Furthermore, tabular algorithms can be readily extended to *dynamic programming* algorithms to search for optimal derivations with respect to appropriate evaluation functions on derivations, as we will see below.

Finite-state grammars have a straightforward tabular algorithm in which table entries consist of a state and an input position (such a table is called a *trellis* in the speech recognition literature). Context-free grammars are the standard example of a phrase structure grammar class whose derivations can be tabulated. In a bottom-up (from words to sentences) derivation for a context-free grammar, the portion of the derivation that corresponds to the recognition of a constituent labeled by a given nonterminal can be simply represented by a table entry giving the nonterminal as a possible label of a

[3]The required properties are analogous to the cut-elimination property that underlies the connection between sequent and natural-deduction presentations of logics (Prawitz, 1965).

substring of the input (Younger, 1967). Although this information leaves out the sequence of steps of the actual derivation, all derivations of that phrase can be easily reconstructed from the set of all table entries derivable for a given input string (Pointers can also be used to keep track of what table entries are used in deriving other entries.) (Younger, 1967; Earley, 1968).

Other grammar classes such as the mildly-context-sensitive grammars (Joshi, Vijay-Shanker, et al., 1991) and some constraint-based grammars are also suitable for tabular procedures (Shieber, 1992).

Search Exhaustiveness

Even if the grammar allows tabular search, it may not be computationally feasible to explore the entire search space because of the effect of grammar size on search space size. For example, the simple finite-state language models used in large-vocabulary speech recognition may have millions of states and transitions. Since each state is potentially considered at each input position, the computation per word recognized is too large for real-time performance. Many techniques have been explored in speech recognition to deal with this problem (Bahl, Jelinek, et al., 1983; Kenny, Hollan, et al., 1993; Paul, 1992; Murveit, Butzberger, et al., 1993; Nguyen, Schwartz, et al., 1993).

In general, the techniques to avoid exploring the entire grammar search space fall into two main classes, *pruning* and *admissible search*. In pruning, an *evaluation function* applied to configurations determines whether they will be expanded further. Since the evaluation function cannot predict the future (to do so accurately it would have to explore the entire search space), pruning may in fact block the correct derivation. The choice of evaluation function is thus a tradeoff between reduced search space (and thus reduced runtime and memory requirements) and the risk of missing the correct analysis (or even every analysis). Currently there is no theory of pruning tradeoffs relating bounds on risk of error to the form of the evaluation function, so the design of evaluation functions is an empirical art.

Although pruning away the correct derivation is as a problem in practical applications, in psycholinguistic modeling it may in fact correspond to failures of human sentence processing, for instance garden paths. *Deterministic parsers* (Marcus, 1980; Hindle, 1993) take pruning to an extreme in using elaborate evaluation functions to select exactly one course of derivation. Dead ends are then supposed to model the situations in which human subjects are forced to recover from parsing failures. Other models, particularly those based on neuronal notions of activation and lateral inhibition, may allow a local *race* between alternative expansions of a configuration but inhibit all but one of the alternatives within a bounded number of steps (Stevenson, 1993; McRoy &

Hirst, 1990; Pritchett, 1988).

Admissible search procedures do not block potential derivations. Instead, they order sub-derivations so that the ones that are more likely to be expanded to the best complete derivations will be considered before less promising ones. The difficulty is of course to define ordering criteria with high probability of reaching the best derivations before exploring a large set of useless configurations. Of particular interest here are A*-type algorithms (Nilsson, 1980) which expand the configuration with lowest *cost estimate*, where the estimate is required to be a lower bound of the true cost (under a cost model appropriate for the task, see below) and identical to the true cost for complete derivations. The first complete derivation reached by an A* algorithm is then guaranteed to have the lowest cost. However, since it is difficult to choose cost estimates that narrow the search sufficiently, more aggressive estimates that may overshoot the true cost are often used, with the result that the first complete derivation may not be the best one.

The selection of evaluation functions for pruning or admissible search is clearly closely tied to the precision-coverage tradeoff.

3.6.4 Grammar Classes

A wide range of grammar classes have been investigated in parsing and language modeling, depending on the nature of the application and on particular insights on language structure and sentence distribution. Grammar classes have been characterized along many different theoretical dimensions. What is known in those areas about certain important grammar classes is described elsewhere in this document 3.6.1.

Here, we consider a more informal and empirical dimension of variation that has great impact in the development of parsers and language models: how much of the required predictive and evidential power belongs to the grammar itself and how much resides in the search procedure controlling the use of the grammar. Choices along this dimension often involve philosophical disagreements on whether language is fundamentally governed by an innate system of rules (the rationalist position most closely identified with Chomsky) or rather a system of statistical regularities, associations and constructions derived by learning (the empiricist position informing much work in statistical language modeling). But they also relate to different choices with respect to the coverage/precision tradeoff.

At one end of the spectrum, which is often associated with empiricist work, extremely unconstraining grammars are controlled by search evaluation functions automatically learned from language data. An extreme example are finite-state *n-gram* grammars, in which states encode information on the last $n-1$ observed words, have been used with practical success in speech recognition (Jelinek, Mercer, et al., 1992). In these grammars every sequence of words is considered a possible sentence, but probabilities are assigned to state transitions to model the relative likelihoods of different strings. As we will see, the association of probabilities to transitions is a useful technique in a wide range of grammatical settings.

While n-gram grammars have proven very useful for language modeling, derivation steps do not correspond in any direct way to possible meaningful relations in the sentence, for instance, those between a main verb and its arguments. Parsing requires more complex grammars, in which derivation steps are associated to possible relations of interest. Even in language modeling, distributional regularities associated to meaningful relationships may be an important source of additional predictive power (Hindle, 1990; Hindle & Rooth, 1991; Dagan, Markus, et al., 1993; Lafferty, Sleator, et al., 1992).

Grammatical representations of meaningful relationships may be usefully classified into three main classes: *linguistic grammars*, *task-oriented grammars* and *data-oriented grammars*. Linguistic grammars and task-oriented grammars have been in use since the beginning of computational linguistics. Data-oriented grammars, in their finite-state form discussed above, go back to the beginning of statistical studies of language by Markov, but data-oriented grammars capable of representing meaningful relationships have only recently started being investigated.

Linguistic Grammars

Most formal linguistic theories have been used at some time or other as the basis for computational grammars. The main issues in applying linguistic theory to the development of computational grammars: *coverage*, *predictive power* and *computational requirements*.

Coverage: Linguistic theories are typically developed to explain puzzling aspects of linguistic competence, such as the relationships between active and passive sentences, the constraints on use of anaphoric elements, or the possible scopes of quantifying elements such as determiners and adverbs. However, actual language involves a wide range of other phenomena and constructions, such as idioms, coordination, ellipsis, apposition and extraposition, which may not be germane to the issues addressed by a particular linguistic theory or which may offer unresolved challenges to the theory. Therefore, a practical grammar will have to go far beyond the proposals of any given theory to cover a substantial proportion of observed language. Even then, coverage gaps are relatively frequent and difficult to fill, as they involve laborious design of new grammar rules and representations.

Predictive Power: Linguistic grammars, being oriented towards the description of linguistic competence, are not intended to model *distributional* regularities arising from pragmatics, discourse and conventional use that manifest themselves in word and construction choice. Yet those are the regularities that appear to contribute most to the estimation of relatively likelihoods of sentences or analyses. The encoding of distributional predictions must be left thus to the search procedure, in the form of an appropriate evaluation function. However, the configurations generated by a grammar may not carry the most useful information in evaluating them. For example, whether a particular prepositional phrase modifies a direct object noun phrase or the main verb depends heavily on the actual verb, noun, preposition and prepositional object (Hindle & Rooth, 1991), but a traditional phrase-structure grammar does not make that information available in the syntactic categories of noun phrase, verb phrase and prepositional phrase. Therefore, in a phrase-structure setting whole derivations rather than individual configurations would have to be evaluated. But this would preclude the factorization of derivations that leads to tractable search as noted above. These considerations explain in part the recent growing interest in *lexicalized* grammatical frameworks such as dependency grammar (Mel'čuk, 1988; Hudson, 1990; Sleator & Temperley, 1991), slot grammar (McCord, 1980; McCord, 1989), categorial grammar (Lambek, 1958; Ades & Steedman, 1982; Moortgat, 1988), Head-Driven Phrase-Structure Grammar (HPSG) (Pollard & Sag, 1987) and lexicalized tree-adjoining grammar (Schabes, 1990), all of which lead to configurations made up of lexical items and direct relationships between them.

Computational Requirements: The best formal explanation of a particular aspect of linguistic competence has no necessary correlation with computational efficiency. For instance, modern versions of transformational grammar based on the theory of government and binding or its more recent developments involve either very complex search procedures or very complex compilation procedures into formalisms with better search properties (Stabler, 1992; Fong, 1992; Johnson, 1992). Similar problems have been noted with respect to HPSG and certain varieties of categorial grammar.

While direct use of formalized linguistic theories for parsing and language modeling seems computationally problematic, much progress has been made in the development of tractable grammatical formalisms capable of encoding important aspects of linguistic theory. The class of mildly-context sensitive formalisms (Joshi, Vijay-Shanker, et al., 1991), of which tree-adjoining grammars (Joshi, Levy, et al., 1975; Joshi, 1985) and combinatory categorial grammar (Ades & Steedman, 1982) are two notable instances, has polynomial-time and space parsing algorithms, and can encode important aspects of transformational and categorial linguistic analysis. Constraint-based grammar formalisms can be intractable or even undecidable in general (Carpenter, 1992), but special cases of interest are often efficiently parsable (Alshawi, 1992). For instance, lexical-functional grammar combines a context-free skeleton with constraints describing non-constituent syntactic properties. Although the combination is intractable in general, a carefully designed constraint-application schedule can make it possible to parse with linguistically-plausible grammars in such a way that the intractability does not arise (Maxwell & Kaplan, 1989).

However, even polynomial-time algorithms may not be sufficiently fast for practical applications, given effect of grammar size on parsing time. Search reduction techniques like those described in section 3.6.3 would then be needed to keep performance within reasonable bounds, at the risk of worse coverage.

Task-Oriented Grammars

For most current applications in text summarization, information retrieval and speech understanding, the predictive and evidential power of a general-purpose grammar and a general control mechanism are insufficient for reasonable performance in the task. Furthermore, even when parameters of the grammar and control mechanism can be learned automatically from training corpora, the required corpora do not exist or are too small for proper training. The alternative is then to devise grammars that specify directly how relationships relevant to the task may be expressed in natural language. For instance, one may use a phrase-structure grammar in which nonterminals stand for task concepts and relationships (for example, *flight* or *leave* in an airline reservation task) and rules specify possible expressions of those concepts and relationships (Seneff, 1992; Ward, 1991b). Such *semantic grammars* have often been used for database access tasks. More generally, a knowledge-representation language (for instance, a frame language) can be used to specify the possible relationships between concepts, and relatively low-power grammatical descriptions (often finite-state) describe natural-language expressions that give strong evidence for concepts and relationships (Jacobs & Rau, 1993; Hobbs, Appelt, et al., 1993).

Task-oriented grammars provide very strong guidance to a parser, but that guidance is bought at the expense of generality and coverage, since the detailed specifications they rely on may often fail to fit naturally-occurring language. Therefore, parsing algorithms for task-oriented grammars are usually allowed to *relax* the grammar by ignoring portions of the input that do not fit the given grammar (Ward, 1991a; Jackson, Appelt, et al., 1991). This can increase coverage usefully in applications such as limited-domain speech understanding and text-summarization, where there are very strong expectations of what are the relevant inputs, but the increase of coverage is in general at the expense of precision.

Data-Oriented Grammars

In so far as linguistic grammars and task-oriented grammars provide strong constraints for modeling and parsing, they risk low coverage because the constraints limit the transitions between configurations, and thus the availability of derivations for strings. In a task-oriented setting this problem can be alleviated, as we have seen, but as far as we know relaxation is a sensible policy only for highly-constrained tasks. An alternative way of increasing coverage is to start with less constraining grammars, and rely on an evaluation function to select the most likely derivations in the more densely connected search space that results from the less constraining grammar. However, this requires finding an appropriate evaluation function. In a data-oriented framework, a *learning* or *training* procedure tries to determine the evaluation function that produces best results on an appropriate training corpus. For example, an n-gram grammar allows any word sequence, but transitions are given probabilities derived from how often states were reached and transitions crossed running the grammar over a training corpus (Jelinek, Mercer, et al., 1992). As another example, frequencies of rule and nonterminal use can be used to estimate rule probabilities for an underconstrained context-free grammar (Baker, 1979; Lari & Young, 1990; Pereira & Schabes, 1992).

Although there have been some successes in training evaluation functions for previously-designed grammars (Fujisaki, Jelinek, et al., 1989; Black, Lafferty, et al., 1992), training with respect to a fixed grammar has the problem that either the grammar allows many transitions that are never observed in reality, forcing the evaluation function to be more complex to rule them out effectively, or it is too restrictive and does not allow transitions that actually occur. That difficulty has motivated the investigation of grammatical frameworks and learning algorithms that will concurrently learn a grammar and an appropriate evaluation function (Sampson, Haigh, et al., 1989; Bod, 1993; Hindle, 1992; Stolcke & Omohundro, 1993). One particular class of such procedures constructs a dictionary of commonly observed substrings or sub-analyses that can be combined by a small set of rules to yield the observed sentences or analyses, with the evaluation function discriminating between alternatives ways of reconstructing a sentence or analysis from the fragments in the dictionary (Bod, 1993; Hindle, 1992). A variety of predictive power and grammar size criteria (for example, Bayesian, minimum-description length) may then used to find good tradeoffs between grammar (dictionary) size, prediction of the training set, and generalization to new material.

3.6.5 Evaluating Derivations

In the overall view of parsing and language modeling given above, a parser or language model searches the configuration space defined by a grammar for possible derivations of the sentence(s) under analysis. Since the grammar by itself is unlikely to encode all the semantic, pragmatic and discourse information relevant to distinguishing plausible analyses from implausible ones, the search needs to be guided by an evaluation function that assigns plausibility scores to derivations. An especially important case is that of *probabilistic grammars*, which associate to each transition the conditional probability of taking that transition from a configuration given that the configuration was reached. Such grammars are based on a *Markovian* or *conditional independence* assumption that the probability of a (partial) derivation depends just on its penultimate configuration and the transition taken from it. Then the probability of a derivation is just the product of the probability of its initial configuration by the product of the probabilities of the transitions in the derivation.

When transitions are directly associated to observable events (for example, extension of a partial sentence by one word in a finite-state model), transition probabilities can be estimated by simply counting the number of times the transition is taken for all possible derivations of all sentences in a training corpus. In general, however, the transition probabilities are not associated to directly observable events. In that case iterative procedures may be used to find a the transition probabilities that maximize the probability that the training corpus was observed (Dempster, Laird, et al., 1977; Baum & Petrie, 1966; Baker, 1979). For language modeling, the training corpus may just be a set of sentences, while for parsing a set of sentences tagged with constraints on possible grammatical relationships (for example, phrase boundaries) is often preferable (Black, Lafferty, et al., 1992; Pereira & Schabes, 1992).

While probabilistic evaluation functions dominate in language modeling, where they are used to estimate the likelihood that a certain word sequence was uttered, other types of evaluation function are often used in parsing, especially those based on the degree of agreement of the best scoring analyses and analyses in a training corpus (Alshawi & Carter, 1994).

Computationally, the critical property of an evaluation function is whether it is compatible with tabular algorithms for searching the derivation space, in the sense that the score of a derivation is determined by the scores of the subderivations into which the derivation is factored by tabulation. For probabilistic functions, this amounts to a strengthened Markovian condition for derivations, which for instance is satisfied by stochastic context-free grammars (Booth & Thompson, 1973; Baker, 1979), certain kinds of parsers for constraint-based grammars (Briscoe & Carroll, 1993) and stochastic tree-adjoining grammars (Schabes, 1992). In such cases, the tabular search algorithms can converted into dynamic programming algorithms (Teitelbaum, 1973; Baker, 1979; Lang, 1989; Jelinek, Lafferty, et al., 1990; Lafferty, Sleator, et al., 1992; Schabes, 1992) to search efficiently for best-scoring derivations.

3.6.6 Future Directions

The issue that dominates current work in parsing and language modeling is to design parsers and evaluation functions with high coverage and precision with respect to naturally occurring linguistic material (for example, news stories, spontaneous speech interactions). Simple high-coverage methods such as n-gram models miss the higher-order regularities required for better prediction and for reliable identification of meaningful relationships, while complex hand-built grammars often lack coverage of the *tail* of individually rare but collectively frequent sentence structures (cf. Zipf's law). Automated methods for grammar and evaluation function acquisition appear to be the only practical way to create accurate parsers with much better coverage. The challenge is to discover how to use linguistic knowledge to constrain that acquisition process.

3.7 Robust Parsing

Ted Briscoe

Computer Laboratory, Cambridge University, Cambridge, UK

Despite over three decades of research effort, no practical domain-independent parser of unrestricted text has been developed. Such a parser should return the correct or a useful *close* analysis for 90% or more of input sentences. It would need to solve at least the following three problems, which create severe difficulties for conventional parsers utilizing standard parsing algorithms with a generative grammar:

1. *chunking*, that is, appropriate segmentation of text into syntactically parsable units;
2. *disambiguation*, that is, selecting the unique semantically and pragmatically correct analysis from the potentially large number of syntactically legitimate ones returned; and
3. *undergeneration*, or dealing with cases of input outside the systems' lexical or syntactic coverage.

Conventional parsers typically fail to return any useful information when faced with problems of undergeneration or chunking and rely on domain-specific detailed semantic information for disambiguation.

The problem of chunking is best exemplified by text sentences (beginning with a capital letter and ending with a period) which—and this sentence is an example—contain text adjuncts delimited by dashes, brackets or commas which may not always stand in a *syntactic* relation with surrounding material. There has been very limited work on this issue—Hindle (1983) describes a system which copes with related problems, such as false starts and *restarts* in transcribed spontaneous speech, whilst Jones (1994) describes a parser which makes limited use of punctuation to constrain syntactic interpretation. Nevertheless, an analysis of the 150K word balanced Susanne Corpus (Sampson, 1994) reveals that over 60% of sentences contain internal punctuation marks and of these around 30% contain text-medial adjuncts. Thus the problem is significant, and further research is required building on linguistic accounts of punctuation (Nunberg, 1990).

Disambiguation using knowledge-based techniques requires the specification of too much detailed semantic information to yield a robust domain-independent parser. Yet analysis of the Susanne Corpus with a crude parser suggests that over 80% of sentences are structurally ambiguous. Several parsers yielding a single *canonical parse* have been developed (Marcus, 1980; Hindle, 1983; de Marcken, 1990). These are often applied to a (partially) disambiguated sequence of lexical syntactic categories. Simplifying the input to the parser in this way circumvents many problems of lexical coverage suffered by systems which require rich sets of syntactic subcategories encoding for example valency of verbs (Jensen, 1991) as well as capitalizing on the relative success and practicality of lexical category disambiguation. Canonical parsers often represent many ambiguities implicitly (Marcus, Hindle, et al., 1983), rather than enumerating possible analyses, and use heuristic disambiguation rules (Hindle, 1989). Such techniques have yielded useful parsers for limited domains but their development is labour intensive and few general principles for their construction have emerged. In recent attempts to manually construct large *treebanks* of parsed texts, canonical parsing has been used as a first but small step of disputed merit (Marcus, Hindle, et al., 1983; Leech & Garside, 1991).

The availability of treebanks and, more generally, large bodies of machine-readable textual data has provided impetus to statistical approaches to disambiguation. Some approaches use stochastic language modeling inspired by the success of HMM-based lexical category disambiguation. For example, probabilities for a probabilistic version of a context-free grammar (PCFG) can be (re-)estimated from treebanks or plain text (Fujisaki, Jelinek, et al., 1989; Sharman, Jelinek, et al., 1990) and used to efficiently rank analyses produced by minimally-modified tabular parsing algorithms. These techniques yielded promising results but have been largely supplanted by statistical parse decision techniques in which the probabilistic model is sensitive to details of parse context (Magerman & Weir, 1992; Briscoe & Carroll, 1993; Black, Lafferty, et al., 1992) and integrated more closely with the parsing algorithm than the grammar. These systems have yielded results of around 75% accuracy in assigning analyses to (unseen) test sentences from the same source as the unambiguous training material. The barrier to improvement of such results currently lies in the need to use more discriminating models of context, requiring more annotated training material to adequate estimate the parameters of such models. This approach may yield a robust automatic method for disambiguation of acceptable accuracy, but the grammars utilized still suffer from undergeneration, and are labour-intensive to develop.

Undergeneration is a significant problem, in one project, a grammar for sentences from computer manuals containing words drawn from a restricted vocabulary of 3000 words which was developed over three years still failed to analyze 4% of unseen examples (Black, Lafferty, et al., 1992). This probably represents an upper bound using manual development of generative grammars; most more general grammars have far higher failure rates in this type of test. Early work on undergeneration focussed on knowledge-based manual specification of *error* rules or rule relaxation strategies (Kwasny & Sonheimer, 1981; Jensen & Heidorn, 1983). This approach, similar to the canonical parse approach to ambiguity, is labour-intensive and suffers from the difficulty of predicting the types of error or extragrammaticality liable to occur. More recently, attempts have been made to use statistical induction to *learn* the correct grammar for a given corpus of data, using generalizations of HMM maximum-likelihood re-estimation techniques to PCFGs (Lari & Young, 1990). This extends the application of stochastic language modeling from disambiguation to undergeneration by assuming the *weakest* grammar for a given category set—that is, the one which contains all possible rules that can be formed for that category set—and using iterative re-estimation of the rule probabilities to converge on the subset of these rules most appropriate to the description of the training corpus.

There are several inherent problems with these statistical techniques which have been partially addressed by recent work. Re-estimation involves considering all possible analyses of each sentence of the training corpus given an (initially) weak grammar, the search space is large and the likelihood of convergence on a useful grammar is low. Pereira and Schabes (1992); Schabes, Roth, et al. (1993) show that constraining the analyses considered during re-estimation to those consistent with manual parses of a treebank reduce computational complexity and leads to a useful grammar. Briscoe and Waegner (1993); Briscoe (1994) demonstrate that similar results can be obtained by imposing general linguistic constraints on the initial grammar and biasing initial probabilities to favour linguistically motivated *core rules*, while still training on plain text. Nevertheless, such techniques are currently limited to simple grammars with category sets of a dozen or so non-terminals or to training on manually parsed data. The induced PCFG can also be used to rank parses and results of around 80% *fit* between correct and automatically-generated analyses have been obtained. It is not possible to directly compare these results with those from pure disambiguation experiments, but there is no doubt that although these systems are achieving 100% or very close grammatical coverage, the use of the resulting PCFG language model for disambiguation only yields fully correct analyses in around 30% of cases.

3.8 Chapter References

ACL (1983). *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, Cambridge, Massachusetts. Association for Computational Linguistics.

ACL (1990). *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, Pittsburgh, Pennsylvania. Association for Computational Linguistics.

ACL (1992). *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, University of Delaware. Association for Computational Linguistics.

ACL (1993). *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, Ohio State University. Association for Computational Linguistics.

Ades, A. E. and Steedman, M. J. (1982). On the order of words. *Linguistics and Philosophy*, 4(4):517–558.

Allen, J., Hunnicutt, M. S., and Klatt, D. (1987). *From text to speech—the MITalk system.* MIT Press, Cambridge, Massachusetts.

Allen, J. F., Schubert, L. K., Ferguson, G., Heeman, P., Hwang, C. H., Kato, T., Light, M., Martin, N., Miller, B., Poesio, M., and Traum, D. R. (1995). The TRAINS project: a case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI.*

Alshawi, H. (1990). Resolving quasi logical form. *Computational Linguistics*, 16:133–144.

Alshawi, H., editor (1992). *The Core Language Engine.* MIT Press, Cambridge, Massachusetts.

Alshawi, H., Arnold, D. J., Backofen, R., Carter, D. M., Lindop, J., Netter, K., Pulman, S. G., and Tsujii, J.-I. (1991). Rule formalism and virutal machine design study. Technical Report ET6/1, CEC.

Alshawi, H. and Carter, D. (1994). Training and scaling preference functions for disambiguation. *Computational Linguistics*, 20:635–648.

ANLP (1994). *Proceedings of the Fourth Conference on Applied Natural Language Processing*, Stuttgart, Germany. ACL, Morgan Kaufmann.

Antworth, E. L. (1990). PC-KIMMO: a two-level processor for morphological analysis. Technical Report Occasional Publications in Academic Computing No. 16, Summer Institute of Linguistics, Dallas, Texas.

Appel, A. W. and Jacobson, G. J. (1988). The world's fastest scrabble program. *Communications of the ACM*, 31(5):572–578.

ARPA (1993). *Proceedings of the 1993 ARPA Human Language Technology Workshop*, Princeton, New Jersey. Advanced Research Projects Agency, Morgan Kaufmann.

Atkins, B. T. S. and Levin, B. (1992). Admitting impediments. In *Lexical Acquisition: Using On-Line Resources to Build a Lexicon.* Lawrence Earlbaum, Hillsdale, New Jersey.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190.

Baker, J. K. (1979). Trainable grammars for speech recognition. In Wolf, J. J. and Klatt, D. H., editors, *Speech communication papers presented at the 97th Meeting of the Acoustical Society of America*, pages 547–550. Acoustical Society of America, MIT Press.

Barwise, J. and Cooper, R. (1981). Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4:159–219.

Barwise, J. and Etchemendy, J. (1987). *The Liar.* Chicago University Press, Chicago.

Barwise, J. and Perry, J. (1983). *Situations and Attitudes.* MIT Press, Cambridge, Massachusetts.

Bates, M., Bobrow, R., Ingria, R., and Stallard, D. (1994). The delphi natural language understanding system. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, pages 132–137, Stuttgart, Germany. ACL, Morgan Kaufmann.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probablistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563.

Berwick, R. C., Abney, S. P., and Tenny, C., editors (1992). *Principle-Based Parsing: Computation and Psycholinguistics.* Kluwer, Dordrecht, The Netherlands.

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

Black, E., Jelinek, F., Lafferty, J., Magerman, D. M., Mercer, D., and Roukos, S. (1993). Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 31–37, Ohio State University. Association for Computational Linguistics.

Black, E., Lafferty, J., and Roukos, S. (1992). Development and evaluation of a broad-coverage probablistic grammar of English-language computer manuals. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 185–192, University of Delaware. Association for Computational Linguistics.

Bod, R. (1993). Using an annotated corpus as a stochastic parser. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pages 37–44, Utrecht University, The Netherlands. European Chapter of the Association for Computational Linguistics.

Booth, T. L. and Thompson, R. A. (1973). Applying probability measures to abstract languages. *IEEE Transactions on Computers*, C-22(5):442–450.

Bouma, G., Koenig, E., and Uszkoreit, H. (1988). A flexible graph-unification formalism and its application to natural-language processing. *IBM Journal of Research and Development*.

Bresnan, J., editor (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, Massachusetts.

Briscoe, E. J. (1992). Lexical issues in natural language processing. In Klein, E. and Veltman, F., editors, *Natural Language and Speech*, pages 39–68. Springer-Verlag.

Briscoe, E. J. (1994). Prospects for practical parsing: robust statistical techniques. In de Haan, P. and Oostdijk, N., editors, *Corpus-based Research into Language: A Feschrift for Jan Aarts*, pages 67–95. Rodopi, Amsterdam.

Briscoe, E. J. and Carroll, J. (1993). Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–59.

Briscoe, E. J. and Waegner, N. (1993). Undergeneration and robust parsing. In Meijs, W., editor, *Proceedings of the ICAME Conference*, Amsterdam. Rodopi.

Carpenter, B. (1992). ALE—the attribute logic engine user's guide. Technical report, Carnegie Mellon University, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Carpenter, B. (1992). *The Logic of Typed Feature Structures*, volume 32 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press.

Church, K. (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143, Austin, Texas. ACL.

COLING (1994). *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan.

Cooper, R. (1983). *Quantification and Syntactic Theory*. Reidel, Dordrecht.

Copestake, A. and Briscoe, E. J. (1992). Lexical operations in a unification based framework. In Pustejovsky, J. and Bergler, S., editors, *Lexical Semantics and Knowledge Representation*. Springer-Verlag, Berlin.

Cutting, D., Kupiec, J., Pedersen, J., and Sibun, P. (1992). A practical part of speech tagger. In *Proceedings of the 3rd Conference on Applied Language Processing*, pages 133–140, Trento, Italy.

Dagan, I., Markus, S., and Markovitch, S. (1993). Contextual word similarity and estimation from sparse data. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 164–171, Ohio State University. Association for Computational Linguistics.

DARPA (1989). *Proceedings of the Second DARPA Speech and Natural Language Workshop*, Cape Cod, Massachusetts. Defense Advanced Research Projects Agency.

DARPA (1991). *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, Pacific Grove, California. Defense Advanced Research Projects Agency, Morgan Kaufmann.

Davidson, D. (1969). Truth and meaning. In Davis, J. W. et al., editors, *Philosophical*, pages 1–20. Hingham.

Davidson, D. (1973). In defense of Convention T. In Leblanc, H., editor, *Truth, Syntax and Modality*, pages 76–85. North Holland.

de Marcken, C. (1990). Parsing the LOB corpus. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pages 243–251, Pittsburgh, Pennsylvania. Association for Computational Linguistics.

De Rose, S. J. (1988). Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14(1):31–39.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38.

EACL (1993). *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, Utrecht University, The Netherlands. European Chapter of the Association for Computational Linguistics.

Earley, J. C. (1968). *An Efficient Context-Free Parsing Algorithm.* PhD thesis, Computer Science Department, Carnegie-Mellon University.

Earley, J. C. (1970). An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102.

Elworthy, D. (1993). Part-of-speech tagging and phrasal tagging. Technical report, University of Cambridge Computer Laboratory, Cambridge, England.

Emele, M. and Zajac, R. (1990). Typed unification grammars. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, Pittsburgh, Pennsylvania. Association for Computational Linguistics.

Flickinger, D. (1987). *Lexical Rules in the Hierarchical Lexicon.* PhD thesis, Stanford University.

Fong, S. (1992). The computational implementation of principle-based parsers. In Berwick, R. C., Abney, S. P., and Tenny, C., editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages 65–82. Kluwer, Dordrecht, The Netherlands.

Frege, G. (1892). Über sinn und bedeutung (translated as 'on sense and reference'). In Geach and Black, editors, *Translations from the Philosophical Writings of Gottlob Frege.* Blackwell, Oxford. translation 1960.

Fujisaki, T., Jelinek, F., Cocke, J., Black, E., and Nishino, T. (1989). A probabilistic parsing method for sentence disambiguation. In *Proceedings of the International Workshop on Parsing Technologies*, Pittsburgh.

Gärdenfors, P. (1990). The dynamics of belief systems: Foundations vs. coherence theories. *Revue Internationale de Philosophie*, 172:24–46.

Garside, R., Leech, G., and Sampson, G. (1987). *Computational Analysis of English: A Corpus-based Approach.* Longman, London.

Gazdar, G. (1987). Linguistic applications of default inheritance mechanisms. In Whitelock, P. H., Somers, H., Bennet, P., Johnson, R., and Wood, M. M., editors, *Linguistic Theory and Computer Applications*, pages 37–68. Academic Press, London.

Graham, S. L., Harrison, M. A., and Ruzzo, W. L. (1980). An improved context-free recognizer. *ACM Transactions on Programming Languages and Systems*, 2(3):415–462.

Greene, B. B. and Rubin, G. M. (1971). Automatic grammatical tagging of English. Technical report, Brown University.

Grice, H. P. (1975). Logic and conversation. In Cole, P., editor, *Speech Acts, Syntax and Semantics, Vol III: Speech Acts*. Academic Press, New York.

Groenendijk, J. and Stokhof, M. (1984). On the semantics of questions and the pragmantics of answers. In Landman, F. and Veltman, F., editors, *Varieties of Formal Semantics*, pages 143–170. Foris, Dordrecht.

Groenendijk, J. and Stokhof, M. (1990). Dynamic montague grammar. In Kalman, L. and Polos, L., editors, *Papers from the Second Symposium on Logic and Language*, pages 3–48. Akademiai Kiadoo, Budapest.

Groenendijk, J. and Stokhof, M. (1991a). Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100.

Groenendijk, J. and Stokhof, M. (1991b). Two theories of dynamic semantics. In van Eijck, J., editor, *Logics in AI—European Workshop JELIA '90, Springer Lecture Notes in Artificial Intelligence*, pages 55–64. Springer-Verlag, Berlin.

Haddock, J. N., Klein, E., and Morrill, G. (1987). *Unification Categorial Grammar, Unification Grammar and Parsing.* University of Edinburgh.

Hindle, D. (1983). Deterministic parsing of syntactic nonfluencies. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 123–128, Cambridge, Massachusetts. Association for Computational Linguistics.

Hindle, D. (1983). User manual for Fidditch, a deterministic parser. Technical Report Technical Memorandum 7590-142, Naval Research Laboratory.

Hindle, D. (1989). Acquiring disambiguation rules from text. In *Proceeds of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 118–125, Vancouver, Canada.

Hindle, D. (1990). Noun classification from predicate-argument structures. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pages 268–275, Pittsburgh, Pennsylvania. Association for Computational Linguistics.

Hindle, D. (1992). An analogical parser for restricted domains. In *Proceedings of the Fifth DARPA Speech and Natural Language Workshop*, pages 150–154. Defense Advanced Research Projects Agency, Morgan Kaufmann.

Hindle, D. (1993). A parser for text corpora. In Atkins, B. T. S. and Zampolli, A., editors, *Computational Approaches to the Lexicon*. Oxford University Press.

Hindle, D. and Rooth, M. (1991). Structural ambiguity and lexical relations. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 229–236, Berkeley, California. Association for Computational Linguistics.

Hobbs, J. R., Appelt, D., Bear, J., Israel, D., Kameyama, M., and Tyson, M. (1993). FASTUS: a system for extracting information from text. In *Proceedings of the 1993 ARPA Human Language Technology Workshop*, pages 133–137, Princeton, New Jersey. Advanced Research Projects Agency, Morgan Kaufmann.

Hobbs, J. R., Stickel, M., Appelt, D., and Martin, P. (1993). Interpretation as abduction. *Artificial Intelligence*, 63(1-2):69–142.

Hudson, R. (1990). *English Word Grammar*. Blackwell, Oxford, England.

Jackson, E., Appelt, D., Bear, J., Moore, R., and Podlozny, A. (1991). A template matcher for robust natural-language interpretation. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, pages 190–194, Pacific Grove, California. Defense Advanced Research Projects Agency, Morgan Kaufmann.

Jacobs, P. S. and Rau, L. F. (1993). Innovations in text interpretation. *Artificial Intelligence*, 63(1-2):143–191.

Järvinen, T. (1994). Annotating 200 million words. In *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan.

Jelinek, F., Lafferty, J. D., and Mercer, R. L. (1990). Basic methods of probabilistic context free grammars. Technical Report RC 16374 (72684), IBM, Yorktown Heights, NY 10598.

Jelinek, F., Mercer, R. L., and Roukos, S. (1992). Principles of lexical language modeling for speech recognition. In Furui, S. and Sondhi, M. M., editors, *Advances in Speech Signal Processing*, pages 651–699. Marcel Dekker.

Jensen, K. (1991). A broad-coverage natural language analysis system. In Tomita, M., editor, *Current Issues in Parsing Technology*. Kluwer Academic Press, Dordrecht.

Jensen, K. and Heidorn, G. (1993). *Natural Langauage Processing: The PLNLP Approach*. Kluwer Academic, Boston, Dordrecht, London.

Jensen, K. and Heidorn, G. E. (1983). The fitted parse: 100% parsing capability in a syntactic grammar of English. In *Proceedings of the First Conference on Applied Natural Language Processing*, pages 3–98.

Johnson, C. D. (1972). *Formal Aspects of Phonological Description*. Mouton, The Hague.

Johnson, M. (1992). Deductive parsing: The use of knowledge of language. In Berwick, R. C., Abney, S. P., and Tenny, C., editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages 39–64. Kluwer, Dordrecht, The Netherlands.

Jones, B. (1994). Can punctuation help parsing? In *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan.

Joshi, A. K. (1985). How much context-sensitivity is necessary for characterizing structural descriptions—Tree adjoining grammars. In Dowty, D., Karttunen, L., and Zwicky, A., editors, *Natural Language Processing—Theoretical, Computational and Psychological Perspectives*. Cambridge University Press, New York.

Joshi, A. K., Levy, L. S., and Takahashi, M. (1975). Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1).

Joshi, A. K. and Schabes, Y. (1992). Tree-adjoining grammars and lexicalized grammers. In *Tree Automata and LGS*. Elsevier Science, Amsterdam.

Joshi, A. K., Vijay-Shanker, K., and Weir, D. J. (1991). The convergence of mildly context-sensitive grammatical formalisms. In Sells, P., Shieber, S., and Wasow, T., editors, *Foundational Issues in Natural Language Processing*. MIT Press.

Kamp, H. (1981). A theory of truth and semantic representation. In Groenendijk, J., Janssen, T., and Stokhof, M., editors, *Formal Methods in the Study of Language*. Mathematisch Centrum, Amsterdam.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic*. Kluwer, Dordrecht.

Kaplan, R. M. and Bresnan, J. (1982). Lexical-functional grammar: a formal system for grammatical representation. In Bresnan, J., editor, *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, Massachusetts.

Kaplan, R. M. and Kay, M. (1994). Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378. written in 1980.

Karlgren, H., editor (1990). *Proceedings of the 13th International Conference on Computational Linguistics*, Helsinki. ACL.

Karlsson, F., Voutilainen, A., Heikkilä, J., and Anttila, A., editors (1994). *Constraint Grammar: A Language-Independent Formalism for Parsing Unrestricted Text.* Mouton de Gruyter, Berlin, New York.

Karttunen, L. (1989). Radical lexicalism. In Baltin, M. and Kroch, A., editors, *Alternative Conceptions of Phrase Structure.* The University of Chicago Press, Chicago.

Karttunen, L. (1993). Finite-state lexicon compiler. Technical Report ISTL-NLTT-1993-04-02, Xerox PARC, Palo Alto, California.

Karttunen, L. and Beesley, K. R. (1992). Two-level rule compiler. Technical Report ISTL-92-2, Xerox PARC, Palo Alto, California.

Kay, M. (1979). Functional grammar. In *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistic Society*, pages 142–158.

Kay, M. (1984). Functional unification grammar: a formalism for machine translation. In *Proceedings of the 10th International Conference on Computational Linguistics*, Stanford University, California. ACL.

Kay, M. (1986). Algorithm schemata and data structures in syntactic processing. In Grosz, B. J., Sparck Jones, K., and Webber, B. L., editors, *Readings in Natural Language Processing*, chapter I. 4, pages 35–70. Morgan Kaufmann Publishers, Inc., Los Altos, California. Originally published as a Xerox PARC technical report, 1980.

Kenny, P., Hollan, R., Gupta, V. N., Lenning, M., Mermelstein, P., and O'Shaughnessy, D. (1993). A*-admissible heuristics for rapid lexical access. *IEEE Transactions on Speech and Audio Processing*, 1(1):49–57.

Koskenniemi, K. (1983). *Two-Level Morphology: a General Computational Model for Word-Form Recognition and Production.* PhD thesis, University of Helsinki. Publications of the Department of General Linguistics,University of Helsinki, No. 11. Helsinki.

Koskenniemi, K. (1990). Finite-state parsing and disambiguation. In Karlgren, H., editor, *Proceedings of the 13th International Conference on Computational Linguistics*, volume 2, pages 229–232, Helsinki. ACL.

Krieger, H.-U. and Schaefer, U. (1994). TDL—a type description language of HPSG. Technical report, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Saarbrücken, Germany.

Krifka, M. (1989). Nominal reference, temporal constitution and quantification in event semantics. In Bartsch, R., van Benthem, J., and van Emde-Boas, P., editors, *Semantics and Contextual Expressions*, pages 75–115. Foris, Dordrecht.

Kupiec, J. (1992). Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6.

Kwasny, S. and Sonheimer, N. (1981). Relaxation techniques for parsing ill-formed input. *American journal of Computational Linguistics*, 7(2):99–108.

Lafferty, J., Sleator, D., and Temperley, D. (1992). Grammatical trigrams: a probabilistic model of link grammar. In Goldman, R., editor, *AAAI Fall Symposium on Probabilistic Approaches to Natural Language Processing*, Cambridge, Massachusetts. AAAI Press.

Lambek, J. (1958). The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–170.

Lang, B. (1974). Deterministic techniques for efficient non-deterministic parsers. In Loeckx, J., editor, *Proceedings of the 2nd Colloquium on Automata, Languages and Programming*, pages 255–269, Saarbrücken, Germany. Springer-Verlag.

Lang, B. (1989). A generative view of ill-formed input processing. In *ATR Symposium on Basic Research for Telephone Interpretation*, Kyoto, Japan.

Lari, K. and Young, S. J. (1990). The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language Processing*, 4:35–56.

Leech, G. and Garside, R. (1991). Running a grammar factory: the production of syntactically analysed corpora or 'treebanks'. In Johansson, S. and Stenstrom, A., editors, *English Computer Corpora: Selected Papers and Bibliography*. Mouton de Gruyter, Berlin.

Leech, G., Garside, R., and Bryant, M. (1994). The large-scale grammatical tagging of text. In Oostdijk, N. and de Haan, P., editors, *Corpus-Based Research into Language*, pages 47–63. Rodopi, Atlanta.

Levinson, S. C. (1983). *Pragmatics*. Cambridge University Press.

Lucchesi, C. L. and Kowaltowski, T. (1993). Applications of finite automata representing large vocabularies. *Software-Practice and Experience*, 23(1):15–30.

Magerman, D. M. and Marcus, M. P. (1991). Pearl: A probabilistic chart parser. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, Pacific Grove, California. Defense Advanced Research Projects Agency, Morgan Kaufmann.

Magerman, D. M. and Weir, C. (1992). Efficiency, robustness and accuracy in Picky chart parsing. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, University of Delaware. Association for Computational Linguistics.

Marcus, M., Hindle, D., and Fleck, M. (1983). D-theory: talking about talking about trees. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 129–136, Cambridge, Massachusetts. Association for Computational Linguistics.

Marcus, M. P. (1980). *A Theory of Syntactic Recognition for Natural Language.* MIT Press, Cambridge, Massachusetts.

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.

Maxwell, John T., I. and Kaplan, R. M. (1989). An overview of disjunctive constraint satisfaction. In Tomita, M., editor, *Proceedings of the First International Workshop on Parsing Technology*, Pittsburgh, Pennsylvania. Carnegie-Mellon University.

McCord, M. C. (1980). Slot grammars. *American journal of Computational Linguistics*, 6(1):255–286.

McCord, M. C. (1989). Design of LMT: A Prolog-based machine translation system. *Computational Linguistics*, 15(1):33–52.

McKeown, K., Kukich, K., and Shaw, J. (1994). Practical issues in automatic documentation generation. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, pages 7–14, Stuttgart, Germany. ACL, Morgan Kaufmann.

McRoy, S. and Hirst, G. (1990). Race-based parsing and syntactic disambiguation. *Cognitive Science*, 14:313–353.

Mel'čuk, I. A. (1988). *Dependency Syntax: Theory and Practice.* State University of New York Press, Albany, New York.

Montague, R. (1973). The proper treatment of quantification in ordinary English. In Hintikka, J., editor, *Approaches to Natural Language*, pages 221–242. Reidel.

Moortgat, M. (1988). *Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus.* PhD thesis, University of Amsterdam, The Netherlands.

Murveit, H., Butzberger, J., Digilakis, V., and Weintraub, M. (1993). Large-vocabulary dictation using SRI's DECIPHER speech recognition system: Progressive search techniques. In *Proceedings of the 1993 International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 319–322, Minneapolis, Minnesota. Institute of Electrical and Electronic Engineers.

Nguyen, L., Schwartz, R., Kubala, F., and Placeway, P. (1993). Search algorithms for software-only real-time recognition with very large vocabularies. In *Proceedings of the 1993 ARPA Human Language Technology Workshop*, pages 91–95, Princeton, New Jersey. Advanced Research Projects Agency, Morgan Kaufmann.

Nilsson, N. J. (1980). *Principles of Artificial Intelligence.* Tioga Publishing Company, Palo Alto, California.

Nunberg, G. (1990). The linguistics of punctuation. Technical Report Lecture Notes 18, CSLI, Stanford, California.

Nunberg, G. and Zaenen, A. (1992). Systematic polisemy in lexicology and lexicography. In *Proceedings of Eurolex 92*, Tampere, Finland.

Oostdijk, N. (1991). *Corpus Linguistics and the Automatic Analysis of English.* Rodopi, Amsterdam, Atlanta.

Ostler, N. and Atkins, B. T. S. (1992). Predictable meaning shifts: Some linguistic properties of lexical implication rules.

Partee, B. (1986). Noun phrase interpretation and type shifting principles. In Groenendijk, J. et al., editors, *Studies in Discourse Representation Theory and the Theory of Generalised Quantifiers*, pages 115–144. Foris, Dordrecht.

Paul, D. B. (1992). An efficient A* stack decoder algorithm for continuous speech recognition with a stochastic language model. In *Proceedings of the 1992 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 25–28, San Francisco. Institute of Electrical and Electronic Engineers.

Pereira, F. C. N. (1985). A new characterization of attachment preferences. In Dowty, D. R., Karttunen, L., and Zwicky, A. M., editors, *Natural Language Parsing—Psychological, Computational and Theoretical perspectives*, pages 307–319. Cambridge University Press.

Pereira, F. C. N. and Schabes, Y. (1992). Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 128–135, University of Delaware. Association for Computational Linguistics.

Petheroudakis, J. (1991). MORPHOGEN automatic generator of morphological information for base form reduction. Technical report, Executive Communication Systems ECS, Provo, Utah.

Pollard, C. and Sag, I. (1994). *Head-driven Phrase Structure Grammar.* Center for the Study of Language and Information (CSLI) Lecture Notes. Stanford University Press and University of Chicago Press.

Pollard, C. and Sag, I. A. (1987). *An Information-Based Approach to Syntax and Semantics: Fundamentals.* Number 13 in Center for the Study of Language and Information (CSLI) Lecture Notes. Stanford University Press and Chicago University Press.

Prawitz, D. (1965). *Natural Deduction: A Proof-Theoretical Study.* Almqvist and Wiksell, Uppsala, Sweden.

Pritchett, B. (1988). Garden path phenomena and the grammatical basis of language processing. *Language*, 64(3):539–576.

Pustejovsky, J. (1991). The generative lexicon. *Computational Linguistics*, 17(4).

Pustejovsky, J. (1994). Linguistic constraints on type coercion. In St. Dizier, P. and Viegas, E., editors, *Computational Lexical Semantics.* Cambridge University Press.

Pustejovsky, J. and Boguraev, B. (1993). Lexical knowledge representation and natural language processing. *Artificial Intelligence*, 63:193–223.

Revuz, D. (1991). *Dictionnaires et lexiques, méthodes et algorithmes.* PhD thesis, Université Paris, Paris.

Ritchie, G. D., Russell, G. J., Black, A. W., and Pulman, S. G. (1992). *Computational Morphology.* MIT Press, Cambridge, Massachusetts.

Roche, E. (1993). Dictionary compression experiments. Technical Report IGM 93-5, Université de Marne la Vallée, Noisy le Grand, France.

Sampson, G. (1994). Susanne: a doomsday book of English grammar. In Oostdijk, N. and de Haan, P., editors, *Corpus-based Linguistics: A Feschrift for Jan Aarts*, pages 169–188. Rodopi, Amsterdam.

Sampson, G., Haigh, R., and Atwell, E. (1989). Natural language analysis by stochastic optimization: a progress report on project APRIL. *Journal of Experimental and Theoretical Artificial Intelligence*, 1:271–287.

Sanfilippo, A. (1993). LKB encoding of lexical knowledge. In Briscoe, T., Copestake, A., and de Paiva, V., editors, *Default Inheritance within Unification-Based Approaches to the Lexicon*. Cambridge University Press.

Sanfilippo, A. (1995). Lexical polymorphism and word disambiguation. In *Working Notes of the AAAI Spring Symposium on Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity and Generativity*. Stanford University.

Sanfilippo, A., Benkerimi, K., and Dwehus, D. (1994). Virtual polysemy. In *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan.

Schabes, Y. (1990). *Mathematical and Computational Aspects of Lexicalized Grammars*. PhD thesis, University of Pennsylvania, Philadelphia. Also technical report (MS-CIS-90-48, LINC LAB179) from the Department of Computer Science.

Schabes, Y. (1992). Stochastic lexicalized tree-adjoining grammars. In *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France. ACL.

Schabes, Y., Roth, M., and Osborne, R. (1993). Parsing the Wall Street Journal with the inside-outside algorithm. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, Utrecht University, The Netherlands. European Chapter of the Association for Computational Linguistics.

Schüller, G., Zierl, M., and Hausser, R. (1993). MAGIC. A tutorial in computational morphology. Technical report, Friedrich-Alexander Universität, Erlangen, Germany.

Seneff, S. (1992). TINA: A natural language system for spoken language applications. *Computational Linguistics*, 18(1):61–86.

Sharman, R., Jelinek, F., and Mercer, R. L. (1990). Generating a grammar for statistical training. In *Proceedings of the Third DARPA Speech and Natural Language Workshop*, pages 267–274, Hidden Valley, Pennsylvania. Defense Advanced Research Projects Agency, Morgan Kaufmann.

Shieber, S. M. (1983). Sentence disambiguation by a shift-reduce parsing technique. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 113–118, Cambridge, Massachusetts. Association for Computational Linguistics.

Shieber, S. M. (1992). *Constraint-Based Grammar Formalisms.* MIT Press, Cambridge, Massachusetts.

Shieber, S. M., Uszkoreit, H., Robinson, J., and Tyson, M. (1983). *The formalism and Implementation of PATR-II.* SRI International, Menlo Park, California.

Sleator, D. and Temperley, D. (1991). Parsing English with a link grammar. Technical report CMU-CS-91-196, Department of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Sproat, R. (1992). *Morphology and Computation.* MIT Press, Cambridge, Massachusetts.

Stabler, Edward P., J. (1992). *The Logical Approach to Syntax: Foundations, Specifications and Implementations of Theories of Government and Binding.* MIT Press, Cambridge, Massachusetts.

Stevenson, S. (1993). A competition-based explanation of syntactic attachment preferences and garden path phenomena. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 266–273, Ohio State University. Association for Computational Linguistics.

Stolcke, A. and Omohundro, S. (1993). Hidden Markov model induction by Bayesian model merging. In Hanson, S. J., Cowan, J. D., and Giles, C. L., editors, *Advances in Neural Information Processing Systems 5*, pages 11–18. Morgan Kaufmann.

Teitelbaum, R. (1973). Context-free error analysis by evaluation of algebraic power series. In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, pages 196–199, Austin, Texas.

Tomita, M. (1987). An efficient augmented context-free parsing algorithm. *Computational Linguistics*, 13(1):31–46.

Touretzsky, D. S., Horty, J. F., and Thomason, R. M. (1987). A clash of intuitions: the current state of nonmonotonic multiple inheritance systems. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pages 476–482, Milan, Italy. Morgan Kaufmann.

Turner, R. (1988). A theory of properties. *The journal of Symbolic Logic*, 54.

Turner, R. (1992). Properties, propositions and semantic theory. In Rosner, M. and Johnson, R., editors, *Computational Linguistics and Formal Semantics.* Cambridge University Press, Cambridge.

Tzoukermann, E. and Liberman, M. Y. (1990). A finite-state morphological processor for Spanish. In Karlgren, H., editor, *Proceedings of the 13th International Conference on Computational Linguistics*, volume 3, pages 277–286, Helsinki. ACL.

Uszkoreit, H. (1986). Categorial unification grammars. In *Proceedings of the 11th International Conference on Computational Linguistics*, Bonn. ACL.

van Eijck, J. and de Vries, F. J. (1992). A sound and complete calculus for update logic. In Dekker, P. and Stokhof, M., editors, *Proceedings of the Eighth Amsterdam Colloquium*, pages 133–152, Amsterdam. ILLC.

Veltman, F. (1985). *Logics for Conditionals*. PhD thesis, University of Amsterdam, Amsterdam.

Voutilainen, A. (1994). *Three Studies of Grammar-Based Surface Parsing of Unrestricted English Text*. PhD thesis, University of Helsinki, Department of General Linguistics, University of Helsinki.

Ward, W. (1991a). Evaluation of the CMU ATIS system. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, pages 101–105, Pacific Grove, California. Defense Advanced Research Projects Agency, Morgan Kaufmann.

Ward, W. (1991b). Understanding spontaneous speech: the Phoenix system. In *Proceedings of the 1991 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 365–367, Toronto. Institute of Electrical and Electronic Engineers.

Younger, D. H. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control*, 10(2):189–208.

Zeevat, H., Klein, E., and Calder, J. (1987). An introduction to unification categorial grammar. In Haddock, J. N., Klein, E., and Morrill, G., editors, *Edinburgh Working Papers in Cognitive Science, volume 1: Categorial Grammar, Unification Grammar, and Parsing*, volume 1 of *Working Papers in Cognitive Science*. Centre for Cognitive Science, University of Edinburgh.