Maintaining the Forest and Burning out the Underbrush in XTAG

Christine Doran, Beth Hockey, Philip Hopely, Joseph Rosenzweig
Anoop Sarkar, B. Srinivas, Fei Xia
IRCS, University of Pennsylvania
Philadelphia, PA 19104
{cdoran,beth,phopely,josephr,anoop,srini,fxia}@linc.cis.upenn.edu

Alexis Nazr, Owen Rambow
CoGenTex, Inc.
840 Hanshaw Road, Suite 11
Ithaca, NY 14850
{nasr,owen}@cogentex.com

Abstract

In this paper we report on the recent advancements and current status of the XTAG Project, housed at the University of Pennsylvania. We discuss the current coverage of the system, as evaluated on the TSNLP English sentences, hierarchical organization of the grammar, and the new and more portable implementation of the X-interface to the grammar and all of the supporting tools in CLISP, which is freely available. We also present a methodology for specializing our grammar to a particular domain, and give some results on this effort.

1 Development and Current Status of XTAG

1.1 History of XTAG

The XTAG project has been ongoing at Penn in some form or another since 1988. It began with a toy grammar run on LISP machines, and currently has a large English grammar, small grammars in several other languages, a sophisticated X-windows based grammar development environment and numerous satellite tools. Approximately 35 people have worked extensively on the system, and at least that many have worked more peripherally. Thus, while it is not a geographically distributed project, it has been temporally distributed. At any given time, there is no single person who is completely familiar with all aspects of either the grammar or the tool kit. As a result, careful documentation has proven to be invaluable. Historically, this has taken the form of distinct papers on individual components; this is still the case for the tools. For the grammar, however, there is now a single document, available as a (frozen) technical report (XTAG-Group, 1995) or a constantly updated html document.[1] The tech report has been useful not only for the people working on the project at Penn, but also for those outside of Penn who are either interested in Tree Adjoining Grammar specifically, or simply interested in seeing how we handled some particular aspect of the grammar.

1.2 Current status of XTAG

Working with and developing a large grammar is a challenging process, and the importance of having good visualization tools cannot be over-emphasized. Currently the XTAG system has X-windows based tools for viewing and updating the morphological and syntactic databases (Karp et al., 1992; Egedi and Martin, 1994), and a sophisticated parsing and grammar development interface. This interface includes a tree editor, the ability to vary parameters

[1] Both are freely available from the project's web page, at http://www.cis.upenn.edu:80/~xtag.

Component	Details
Morphological Analyzer and Morph Database	Consists of approximately 317,000 inflected items. Entries are indexed on the inflected form and return the root form, POS, and inflectional information. Database does not address derivational morphology.
POS Tagger and Lex Prob Database	Wall Street Journal-trained trigram tagger (Church, 1988) extended to output N-best POS sequences (Soong and Huang, 1990). Decreases the time to parse a sentence by an average of 93% .
Syntactic Database	More than 105,000 entries. Each entry consists of: the uninflected form of the word, its POS, the list of trees or tree-families associated with the word, and a list of feature equations that capture lexical idiosyncrasies.
Tree Database	768 trees, divided into 54 tree families and 164 individual trees. Tree families represent subcategorization frames; the trees in a tree family would be related to each other transformationally in a movement-based approach.
X-Interface	Menu-based facility for creating and modifying tree files. User controlled parser parameters: parser's start category, enable/disable/retry on failure for POS tagger. Storage/retrieval facilities for elementary and parsed trees. Graphical displays of tree and feature data structures. Hand combination of trees by adjunction or substitution for grammar development. Ability to assign POS tag and/or Supertag before parsing

Table 1: System Summary

in the parser, work with multiple grammars and/or parsers, and use metarules for more efficient tree editing and construction (Becker, 1994). An interface for the lexical organization hierarchy is under development.

The large grammar (database) version of XTAG has recently been ported to CLISP, contemporary public-domain software, with the specific goal of permitting XTAG to run under the (public-domain) Linux operating system.[2] The public domain software suite as of this writing has been tested under SunOS 5.4 and Linux 1.2.13, 2.0.20 & 2.0.21 on Intel-based platforms. A user currently may demo a live version of XTAG from a CD-ROM on Intel Linux without having to install or recompile the suite; those interested can contact xtag-request@linc.cis.upenn.edu for further information. Development of an MS-DOS-loadable demo CD-ROM version of the software suite using Linux is underway, and a Maclinux version is also planned.[3]

A snap-shot of the English grammar and parser is shown in Figure 1. We also have a large French grammar (started at Penn and expanded at Paris 7, by Anne Abeillé), and small grammars for Korean, Chinese and Hindi. The X-windows interface is completely modular and can be (and has been) used with any of these grammars.

1.3 Grammar Coverage

To evaluate the coverage of the English grammar, we ran it on the Test Suites for Natural Language Processing (TSNLP) English corpus (Lehmann et al., 1996). The corpus is intended to be a systematic collection of English grammatical phenomena, including complementation, agreement, modification, diathesis, modality, tense and aspect, sentence and clause types, coordination, and negation. It contains 1409 grammatical sentences and phrases and 3036 ungrammatical ones.

Before parsing the TSNLP data, we made a few tokenization changes: we changed contractions from two tokens to one, downcased the first words of sentences, changed a pair of square brackets to parentheses and changed quotes to pairs of opens and closes. There were 42 examples which we judged ungrammatical, and removed from the test corpus.

[2] Linux is the largest working example of a distributed software development project, and has been ported to more machines than any other operating system.

[3] Configurations for various memory sizes are being developed, but it is recommended that an Intel-based user running a demo have somewhere between 16 and 64 megabytes of memory and at least a 586-level processor for relatively decent operation speed.

Error Class	%	Example
POS Tag	19.7%	She adds to/V it , He noises/N him abroad
Missing item in lexicon	43.3%	*used* as an auxiliary V, *calm NP down*
Missing tree	21.2%	*should've, bet NP NP S, regard NP as Adj*
Feature clashes	3%	*My every firm, All money*
Rest	12.8%	*approx, e.g.*

Table 2: Error analysis of TSNLP English corpus

These were sentences with conjoined subject pronouns, where one or both were accusative, e.g. *Her and him succeed.*. We simply removed these sentences from the corpus. Overall, we parsed 61.4% of the 1367 remaining sentences and phrases. The errors were of various types, broken down in Table 2.

The missing lexicon items are obviously the easiest of these to remedy. This class also highlighted the fact that our grammar is heavily slanted toward American English – our grammar does not handle *dare* or *need* as auxiliary verbs, and there were a number of very British particle constructions, e.g. *She misses him out.* The missing trees are slightly harder to address, but the data obtained here is very useful in helping us fill gaps in our grammar. We do not currently handle the class of modal+ *'ve* contractions at all, and this clearly ought to be remedied. The feature clashes are mostly in sequences of determiners, and would need to be looked at more closely to see whether the changes needed to correct them would do more harm than good. One general problem with the corpus is that, because it uses a very restricted lexicon, if there is one problematic lexical item it is likely to appear a large number of times and cause a disproportionate amount of grief. *Used to* appears 33 times and we get all 33 wrong. However, it must be noted that the XTAG grammar has analyses for syntactic phenomena that were not represented in the TSNLP test suite such as sentential subjects and subordinating clauses among others.

As noted by our reviewers, the TSNLP test suite in its current status is not intended as a ready made representative set of test data that can be used for cross system evaluation. We are aware of this and we present the results of our system performance on TSNLP as another data point in our sequence of grammar evaluation experiments. The English grammar has previously been evaluated on ATIS, Wall Street Journal and IBM-Manual data (Srinivas et al., 1996), and found to perform well in these domains.

2 Grammar Organization

The XTAG English grammar currently consists of 768 tree templates, so grammar maintenance is no small task. In general, lexicalizing a TAG creates redundancy because the same trees, modulo their anchor labels, may be associated with many different lexical items. We have eliminated this redundancy by storing only abstract tree templates with uninstantiated anchor labels, and instantiating lexicalized trees on the fly, as words are encountered in the input. Another source of redundancy, however, is the reuse of tree substructures in many different tree templates. For example, most sentential tree templates include a structural fragment corresponding to the phrase-structure rule S → NP VP.

This redundancy poses a problem for grammar maintenance and revision. To consistently implement a change in the grammar, all the relevant trees currently must be edited individually, although we do have an implementation of Becker's metarules (Becker, 1994) which allows us to automate this process to a great extent. For instance, the addition of a new feature equation associated with the structural fragment corresponding to S → NP VP would affect most clausal trees in the grammar. Crucially, one can only manually verify that such an update does not conflict with any other principle already instantiated in the grammar. As the grammar grows, the difficulty of this task grows with it.

Following the idea first proposed in (Vijay-Shankar and Schabes, 1992), we extend the idea of abstraction over lexical anchors. A tree template with an unspecified anchor label subsumes an entire class of lexically specified trees; similarly, we define "meta-templates", or *quasi-trees*, which subsume classes of tree templates. The quasi-trees are specified by partial tree descriptions in a logical language patterned after Rogers and Vijay-Shanker (Rogers and Vijay-Shankar, 1994); we call the partial descriptions *blocks*. Since we are using a feature-based LTAG, our language has also been equipped with descriptive predicates allowing us to specify a tree's feature-structure equations, in addition to its structural characteristics. Each block abstractly de-

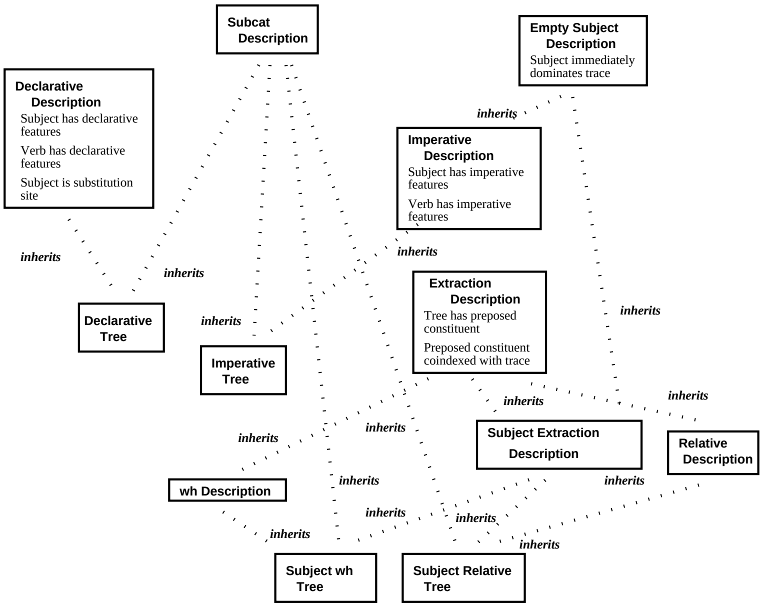


Figure 1: Tree are generated by combining partial tree description

scribes all trees incorporating the partial structure it represents.

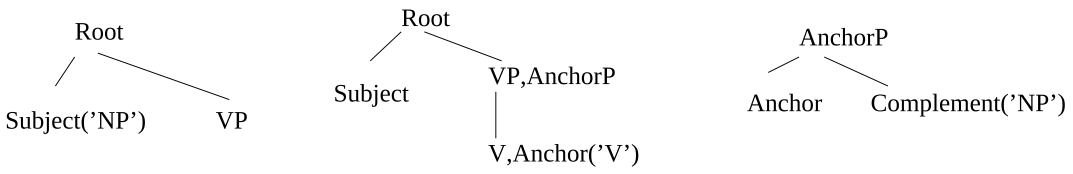
An elementary tree template is expressed as a conjunction of blocks. The blocks are organized as an inheritance lattice, so that descriptive redundancy is localized within an individual block. Within this description lattice, we isolate two sub-lattices which form more or less independent dimensions: the subcategorization sub-lattice and the sub-lattice of descriptions of "transformations" on base subcategorization frames, such as wh-question formation and imperative mood. The subcategorization sub-lattice is further divided into four fairly orthogonal subparts: (1) the set of blocks describing the syntactic subject, (2) those for the main anchor(s), (3) those describing complements and (4) those for structure below a complement.

Similar approaches have been pursued for a large French LTAG by (Candito, 1996) and for the XTAG English grammar by (Becker, 1994). Following the ideas set forth in (Vijay-Shankar and Schabes, 1992), Candito constructs a description hierarchy in much the same way as the present work, albeit for a smaller range of constructions than what exists in the XTAG grammar. Becker's meta-rules can also been seen as partial descriptions, wherein the inputs and outputs of the meta-rules are sisters in a description hierarchy and the parent is the common structure shared by both. However, there is still redundancy across meta-rules whose inputs apply to the same partial descriptions. For instance, the subject wh- extraction and subject relative metarules would be specified independently and both refer to an NP in subject position of a clause.

2.1 Hierarchical Organization of the Current English Grammar

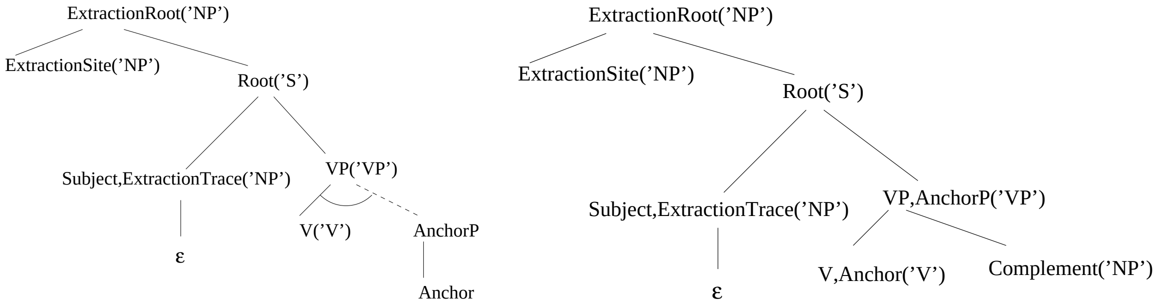
For the XTAG English grammar, we use the hierarchy to build the tree templates at compile time. In maintaining the grammar, however, only the abstract descriptions need ever be manipulated; the larger sets of tree templates and actual trees which they subsume are computed deterministically from these high-level descriptions, as given in Figure 1.

Consider, for example, the description of the relative clause tree for transitive verbs which contains four blocks: one specifying that its subject is extracted, one that the subject is an NP, one that the main anchor is a verb, and one that the complement is an NP. These blocks correspond to the quasi-trees (partially specified trees) shown in Figure 2 and 3(1) and when combined will generate the elementary tree in Figure 3(2). For the sake of simplicity, feature equations are not shown. In these figures, solid lines and dashed lines denote the parent and dominance relations respectively; each node has a label, enclosed in parentheses, and at least one name. Multiple names for the same node are separated by commas such as *VP, AnchorP* in Figure 2(2). The arc in Figure 3(1) indicates that the precedence order of V and AnchorP is unspecified. (In small clauses, the main anchor is a preposition,



(1) Subject_is_NP (2) Main_anchor_is_Verb (3)Complement_is_NP

Figure 2: Subcategorization quasi-trees



(1) quasi-tree for relative clause (2) tree generated from the four quasi-trees

Figure 3: Quasi-tree for subject extraction in relative clause, and tree generated by combining it with the 3 quasi-trees in Figure 2

adjective or noun, not a verb, so AnchorP and VP are not always the same node.)

Our lexical organization tool is implemented in Prolog, and contains blocks which account for 85% of the current English grammar. By the time of the workshop, the remainder of the grammar will also be implemented. There is also an interface to the Prolog module, and a visualization tool for displaying portions of the description lattice.

2.2 A tool for grammar examination

Being able to specify the grammar in a high-level description language has obvious advantages for maintenance and updating of the grammar, in that changes need only be made in one place and are automatically percolated appropriately throughout the grammar. We expect to reap additional benefits from this approach when developing a grammar for another language. Beyond these issues of efficiency and consistency, this approach also gives us a unique perspective on the existing grammar as a whole. Defining hierarchical blocks for the grammar both necessitates and facilitates an examination of the linguistic assumptions that have been made with regard to feature specification and tree-family definition. This can be very useful for gaining a overview of the theory that is being implemented and exposing gaps that have not yet been explained. Because of the organic way in which the grammar was built over the years, we have always suspected that there might exist a fair amount of inconsistency either within the feature structures, or within the tree families. The effort in organizing the lexicon has so far turned up very few non-linguistically motivated inconsistencies, which is a gratifying validation of the constraints imposed by the LTAG formalism.

Our work in tree organization has allowed us to characterize three principal types of exceptions in the XTAG English grammar: (1) a class of trees is missing from the grammar, though this class would be expected from allowing the descriptive blocks to combine freely (for example, a sentential subject with a verb anchor and a PP complement); (2) within a class of trees, some member is missing, though an analogous member is present in another class (extraction of the clausal complement of a noun-anchored predicative); (3) one tree in a class can be generated by combining quite general descriptions, but there is an exceptional piece of structure or feature equation (the ergative alternation of transitive verbs). While these may sometimes reflect known syntactic generalizations (e.g. extraction islands, as with the example in (2)), they may also reflect inconsistencies which have arisen over the lengthy time-course of grammar development and need to be corrected. As previously noted, the latter have so far been quite limited in number and significance.

Our approach makes it incumbent on us to seek principled explanations for these irregularities, since they must be explicitly encoded in the description hierarchy. Without the description hierarchy, there would be no need to reconcile these differences, since they would be entirely independent pieces of a flat grammar.

3 Tailoring XTAG to the Weather Domain

While it is certainly interesting to develop a wide-coverage grammar for its own sake, it is clear that for any practical application the grammar will have to be tailored to the particular domain. Our overarching goal in building the English grammar was to make it broad enough and general enough that tailoring would be a matter of extracting the desired subset of the lexicon and/or the tree database. In this section, we will discuss and evaluate various approaches to specializing a large grammar, and then will discuss our effort at specializing the XTAG English grammar for a weather-message domain.

3.1 General Considerations

In considering how one might specialize a grammar, we make the following basic assumptions: that a sub-language exists; that it can be identified; that there is training data (usually unannotated) available; that default mechanisms will be adequate for handling over-specialization (since we know training data will not perfectly reflect the genre) and that the smaller grammar combined with defaults will still be more efficient than the large grammar.

Based on these assumptions, the first choice is whether to do full parsing at all in the final application. If the domain contains a large number of fragments, it might be preferable to use a partial parsing approach, in which case development of a sub-grammar will be less crucial. Supertagging (Joshi and Srinivas, 1994) is one such approach; once the supertagger is trained for the domain, it could be used in place of the full parser. If, however, it is determined that full parsing is practicable for the domain, there are still a number of considerations in deriving the sub-grammar.

In the ideal situation, there would already be a corrected parsed corpus (treebank), which can be used for crafting a sub-grammar for the domain. This is exceptionally unlikely, and in the more common case, training data will have to be constructed, either manually or automatically. In a lexicalized grammar like LTAG, this turns out to be quite manageable, since there are distinct representations which encode syntactic structures. We can use a statistical approach, such as supertagging, to make a first pass at assigning the correct structures to each word, and then hand-correct them to derive the relevant set of structures. In non-lexicalized grammars, this process would be much more difficult, because there is no straightforward way to associate structures with lexical word and to identify the rules to be eliminated. If it is impossible to create training data by any other method, the full grammar can be applied and then the output corrected to create a treebank of the training data. Needless to say, this is a tedious, time-consuming and computationally expensive task. Alternatively, a domain expert could provide a list of grammatical phenomena needing to be handled, and this list used to extract the sub-grammar.

Once the training data has been processed by one of these methods, the sub-grammar is extracted based on the elementary objects in the grammar required to handle all of the syntactic phenomena identified in the training set. This could mean extracting precisely the constructions used in the training set, or generalizing from them. A lexical hierarchy such as that described in Section 2 can be used for this process, with generalization performed along either of the hierarchy dimensions. The expansion could be done by general principles (add all trees of a certain subcat frame if any are present), or could be done based on performance of the sub-grammar on held-out training data.

Most domains have a rich terminological vocabulary, which if not taken into account can cause prohibitive ambiguity in parsing and interpretation. Identifying and demarcating domain specific terminology is helpful for all of these approaches, since the terms can then be treated as single tokens. This can been done either manually or automatically (Daille, 1994; Jacquemin and Royaut, 1994).

Once the sub-grammar has been finalized, strategies for recovering from failure to parse should be developed. One simple strategy is to fall back to the large/whole grammar. A more sophisticated strategy would be to back off using a lexical hierarchy in the same way it was used for generalizing from the training set.

3.2 Specializing to the Weather Domain

The domain we chose to test out these strategies was weather reports, provided to us by CoGenTex.[4] The sentences tend to be quite long (an average of 20 to-

[4] Thanks to the Contrastive Syntax Project, Linguistics Department of the University of Montreal, for the use of their weather synopsis corpus.

kens/sentence) and complex, and included a large amount of domain specific terminology in addition to many geographical names. To identify the domain specific terms, we are using a hand-collected list, but we are currently working with Beatrice Daille (Daille, 1994) to collect them automatically. Collapsing these terms reduced the length of the test sentences by 22%. Example 1 is illustrative of the type of sentences and the terminology in this domain. We split the development data into a training set (99 sentences) and a test set (50 sentences).

(1) Skies were beginning to clear over [western New-Brunswick] and [western Nova Scotia] early this morning as [drier air] pushed into the district from the west.

We primarily pursued the full-parsing approach, but explored partial parsing to a more limited extent as well. Since we did not have access to parsed training data, we tried several of the approaches discussed above for creating the small grammar. Parsing with the full grammar was impractical and inefficient. We also attempted to parse the training sentences using a sub-grammar, created with the aid of a domain expert who identified relevant syntactic constructions. We used this information as input to the lexical organization tool to extract a sub-lattice of the grammar hierarchy (along both the subcat and transformational dimensions). However, initial experiments suggest this first pass sub-grammar was still too large, and that more radical pruning of the large grammar would be required.

The most effective strategy for us was to use the supertagger to create an annotated training corpus. The supertagger (which had been trained on 200,000 words of correctly supertagged WSJ data) performed at about 87%. We then manually corrected the erroneous supertags, and prepared a subgrammar using the word/POS-tag/supertag triples from the weather training corpus. Using this subgrammar, we setup the task to parse the 50 test sentences, backing off to the full grammar. As of the time of submission of this paper, we were still parsing these sentences. Although the sentences which could be parsed by the sub-grammar were assigned a parse very quickly, overall, we did not see the anticipated speed up that we expected. We suspect that backing off to the full grammar is not the best way to go, and are working on ways to back off using the lexical inheritance hierarchy.

There are a number of directions for future work suggested by these initial experiments. With regard to partial parsing, we retrained the supertagger on the 100 training sentences (1416 tokens). This supertagger performed at 78%, a considerable decrease from the WSJ-trained supertagger, but respectable given the small training set. Some of the errors produced by the WSJ-trained supertagger were idiosyncratic to the newswire domain, so we plan to explore strategies for combining the information from the WSJ domain with the weather report domain, analogous to techniques used in the speech domain.

References

Becker, T. 1994. Patterns in metarules. In *Proceedings of the 3rd TAG+ Conference*, Paris, France.

Candito, Marie-Helene. 1996. A principle-based hierarchical representation of ltags. In *Proceedings of COLING-96*, Copenhagen, Denmark, August.

Church, Kenneth Ward. 1988. A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In *2nd Applied Natural Language Processing Conference*, Austin, Texas.

Daille, Beatrice. 1994. Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. In *The Balancing Act Workshop: Combining Symbolic and Statistical Approaches to Language.*

Egedi, Dania and Patrick Martin. 1994. A Freely Available Syntactic Lexicon for English. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, Nara, Japan, August.

Jacquemin, C. and J. Royaut. 1994. Retrieving terms and their variants in a lexicalised unification-based framework. In *Proceedings of SIGIR94*, pages 132–141.

Joshi, Aravind K. and B. Srinivas. 1994. Disambiguation of Super Parts of Speech (or Supertags): Almost Parsing. In *Proceedings of the* 17^{th} *International Conference on Computational Linguistics (COLING '94)*, Kyoto, Japan, August.

Karp, Daniel, Yves Schabes, Martin Zaidel, and Dania Egedi. 1992. A Freely Available Wide Coverage Morphological Analyzer for English. In *Proceedings of the* 15^{th} *International Conference on Computational Linguistics (COLING '92)*, Nantes, France, August.

Lehmann, Sabine, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Hervé Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. 1996. TSNLP — Test Suites for Natural Language Processing. In *Proceedings of COLING 1996*, Copenhagen.

Rogers, J. and Vijay-Shankar. 1994. Obtaining trees from their descriptions: An application to tree adjoining grammars. *Computational Intelligence*, 10(4).

Soong, Frank K. and Eng-Fong Huang. 1990. Fast Tree-Trellis Search for Finding the N-Best Sentence Hypothesis in Continuous Speech Recognition. *Journal of Acoustic Society, AM.*, May.

Srinivas, B., Christine Doran, Beth Ann Hockey, and Aravind Joshi. 1996. An approach to robust partial parsing and evaluation metrics. In *Proceedings of the Workshop on Robust Parsing at European Summer School in Logic, Language and Information*, Prague, August.

Vijay-Shankar and Y. Schabes. 1992. Sharing in lexicalized tree adjoining grammar. In *Proceedings of COLING-92*, Nantes, France, August.

XTAG-Group, The. 1995. A Lexicalized Tree Adjoining Grammar for English. Technical Report IRCS 95-03, University of Pennsylvania.