Approaches for Learning Constraint Dependency Grammar from Corpora

M. P. Harper, W. Wang, and C. M. White

Electrical & Computer Engineering
Purdue University
West Lafayette, IN 47907-1285

Abstract. This paper evaluates two methods of learning constraint dependency grammars from corpora: one uses the sentences directly and the other uses subgrammar expanded sentences. Learning curves and test set parsing results show that grammars generated directly from sentences have a low degree of parse ambiguity but at a cost of a slow learning rate and less grammar generality. Augmenting these sentences with subgrammars dramatically improves the grammar learning rate and generality with very little increase in parse ambiguity.

1 Introduction

A common trend in natural language processing research is to learn grammars from a corpus of sentences annotated with parse information. This trend has been supported by the availability of treebanks for various languages. One of the most ubiquitous treebanks is the Penn Treebank, which provides context-free grammar (CFG) bracketing for sentences from various corpora including the Wall Street Journal, ATIS, and the Brown Corpus. Because the trees directly represent the grammar rules by the derivations in the parse trees of the bank, it is possible to extract a rule set by simply reading them off the trees. It is also possible to count the frequencies of the rules to support the development of a probabilistic CFG.

In 1998, Johns Hopkins summer workshop [2] investigated learning grammars for the Czech language from the Prague Dependency Treebank [1], which contains sentences with morphologically rich tags assigned to each word with a dependency tree represented using links between each word and its head. Despite the fact that the treebank was available (albeit incomplete), none of the researchers at the workshop constructed Czech grammars by reading dependency rules from the treebank, instead many of them converted the dependencies into CFG rules despite the fact that Czech has crossing-dependencies that are not representable within a CFG framework. In this paper, we develop a method for obtaining grammars from a corpus of sentences labeled with dependencies consistent with Constraint Dependency Grammar (CDG) [3, 6]. We investigate the effect that two types of annotation, six grammar extraction methods, and three additional types of constraint relaxation have on the quality of CDGs learned from corpora.

There is currently no English corpus annotated with CDG grammar information. Hence, for this investigation, we have developed a CDG treebank for

a medium sized domain, DARPA Naval Resource Management [9], which has a vocabulary of around 1,000 different words drawn from queries in the form of wh-questions and yes/no-questions involving naval resources or commands for controlling an interactive database and graphics display. Interviews of naval personnel familiar with naval resource management tasks were used to build an underlying grammar model for this task, which was then used to generate corpora of sentences that were read by a wide variety of speakers in order to generate the standard Resource Management (RM) corpus [9]. RM contains 2,844 distinct sentences, which we annotated in order to obtain a variety of grammars. The DARPA Naval Resource Management task was chosen for several reasons: it provides a good source of both training and testing materials; the sentences have syntactic variety and a definable semantics; the scope of the problem is limited enough to enable the investigation of a wider variety of grammar development techniques than would be possible with larger corpora; it is a domain specific task; and the underlying grammar that we are trying to learn is well defined, which is important for evaluating the learnability of CDG from corpora.

Section 2 defines CDG and the techniques for obtaining CDGs directly from sentence annotations. Section 3 presents the two types of grammar annotation. Section 4 summarizes the empirical evaluations of the various grammars obtained from the annotations and discusses the results.

2 Constraint Dependency Grammar (CDG)

Constraint Dependency Grammar (CDG) [3, 7] uses constraints to determine the grammatical structure of a sentence, which is represented as labeled dependencies between words. A CDG is defined as a tuple, $\langle \Sigma, R, L, C, T \rangle$, where Σ is a finite set of lexical categories (e.g., determiner), R is a finite set of uniquely named roles or role ids (e.g., governor, need1, need2), L is a finite set of labels (e.g., subject), C is a constraint formula that determines whether an input is grammatical, and T is a table that specifies which roles are supported for each lexical category (e.g., determiners use only the governor role, but verbs use both a governor role and need roles), the set of labels that are supported for each role and lexical category, the domain of feature values for each feature type (the domain of feature type i is denoted F_i), the feature types that are defined for each category in Σ, and the subset of feature values allowed by each category and feature type combination. The number of roles in a CDG is the *degree* of the grammar. For the RM corpus, we chose a degree of four in order to support four roles: `governor`, `need1`, `need2`, and `need3`. Additionally, there are 16 lexical categories (`adj`, `adv`, `conj`, `det`, `mod`, `noun`, `particle`, `predet`, `prep`, `pronoun`, `propernoun`, `verb`, `month`, `cardinal`, `ordinal`, and `comp`), 24 labels, and 13 lexical feature types (`subcat`, `agr`, `case`, `vtype` (e.g., progressive), `mood`, `gap`, `inverted`, `voice`, `behavior` (e.g., mass, count), `type` (e.g., interrogative, relative), `semtype`, `takesdet`, and `conjtype`), each with an appropriate set of feature values. For parsing sentences using CDG, access to a dictionary of word entries is required. Each word is comprised of one or more lexical entries. A lexical entry is made up of one lexical category $\sigma \in \Sigma$ and a single feature value for each feature supported by σ.

In CDG, a parse of a sentence is provided by assigning role values to each of the roles associated with each word such that the constraints in C are satisfied. Each word in a sentence has up to p different roles (most lexical classes need only one or two [4]), with a parse consisting of a maximum of $n * p$ role value assignments, where n is the number of words in the sentence. Consider the parse for *Clear the screen* depicted in the white portion of Figure 1. Each word in the parse has a lexical category and a set of feature values. Also, each word has a governor role (denoted G) that is assence, we also measured the parse ambiguity for each method, which is signed a role value, comprised of a label as well as a modifiee, which indicates the position of the word's governor or head. For example, the role value assigned to the governor role of *the* is `det-3`, where its label `det` indicates its grammatical function and its modifiee `3` is the position of its head *screen*. The need roles (denoted N1, N2, and N3) are used to ensure the requirements of a word are met, as in the case of the verb *clear*, which needs an object (but since the word takes no explicit subject and requires no other complements, the modifiee of the role value assigned to N1 and N3 is set equal to its own position).

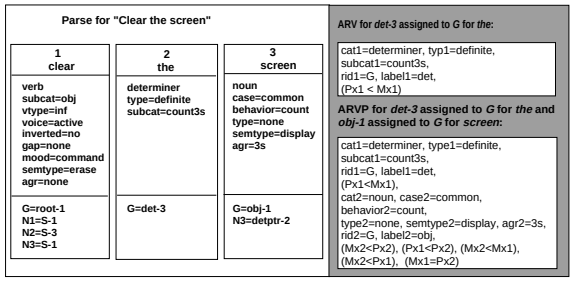

Fig. 1. A CDG parse (see white box) is represented by the assignment of role values to roles associated with a word with a specific lexical category and one feature value per feature. ARVs and ARVPs (see gray box) represent grammatical relations that can be extracted from a sentence's parse.

During parsing, the grammaticality of a sentence in a language defined by a CDG is determined by applying C to the possible role value assignments. Originally C was comprised of a set of hand-written rules specifying which role values (unary constraints) and pairs of role values (binary constraints) were grammatical [7]. In order to derive the constraints directly from CDG annotated sentences, we have developed an algorithm to extract grammar relations using information derived directly from annotated sentences [5]. Using relative position information, unary constraints are represented as a finite set of *abstract role values* (ARVs). Formally, an ARV for a particular grammar $G = \langle \Sigma, R, L, C, T \rangle$ is an element of the set: $\mathcal{A}_1 = \Sigma \times R \times L \times F_1 \times \ldots \times F_k \times \mathrm{UC}$, where k is the number of feature types defined in T, F_i represents the set of feature values for that type, and UC encodes the three possible positional relations between the position (denoted $\mathrm{P}x_1$) and modifiee (denoted $\mathrm{M}x_1$) of a role value assigned to the role of a word ($\mathrm{M}x_1$ and $\mathrm{P}x_1$ can be related by $<$, $>$, or $=$). The gray box

of Figure 1 shows an example of an ARV obtained from the parsed sentence. Similarly binary constraints are represented as a finite set of abstract role value pairs (ARVPs) which are members of the domain $\mathcal{A}_2 = \Sigma \times R \times L \times F_1 \times \ldots \times F_k \times \Sigma \times R \times L \times F_1 \times \ldots \times F_k \times \mathrm{BC}$, where BC encodes for a pair of role values the positional relations among their positions ($\mathrm{P}x_1$ and $\mathrm{P}x_2$) and modifiees ($\mathrm{M}x_1$ and $\mathrm{M}x_2$) (there are six ways to pair the position and modifiee of two role values, each of which can be related in three different ways). The gray box of Figure 1 also shows an example of an ARVP obtained from the parsed sentence.

We have developed several methods to derive C directly from the annotated sentences of a corpus. ARVs include the role, the label of its assigned role value, the category and feature values of its word, a UC relation, and the modifiee's lexical category and feature values, which we call *modifiee constraints*. Modifiee constraints are used for ARVs regardless of the extraction method because their use does not change grammar coverage and they help to improve parse times by eliminating ungrammatical role values during the less costly early stages of parsing [4]. Because the ARVP space is larger than the ARV space, using all of the information associated with all pairs of role values could generate a very large, overly specific grammar. Hence, six variations for extracting ARVPs were examined:

1. **Full Mod:** contains lexical category, role, label, modifiee, and feature information for all pairs of role values from annotated sentences, as well as modifiee constraints. For a role value pair in a sentence to be valid during parsing with this grammar, it must match an extracted ARVP.
2. **Full:** like **Full Mod** without modifiee constraints.
3. **Feature Mod:** contains all grammar relations between all pairs of role values, but it considers feature and modifiee constraints only for pairs that are directly related by a modifiee link (i.e., one of $\mathrm{P}x_1 = \mathrm{M}x_2$, $\mathrm{P}x_2 = \mathrm{M}x_1$, or $\mathrm{M}x_1 = \mathrm{M}x_2$ is true). During parsing, if a role value pair is related by a modifiee link, then it must match a corresponding ARVP with full feature and modifiee information; otherwise, it must match an ARVP, ignoring feature and modifiee constraint information.
4. **Feature:** like **Feature Mod** without modifiee constraints.
5. **Direct Mod:** stores grammar relations only for those pairs of role values that are directly related by a modifiee link. During parsing, if a role value pair is related by such a link, then a corresponding ARVP must appear in the grammar for it to be allowed; otherwise, the pair is allowed.
6. **Direct:** like **Direct Mod** without modifiee constraints.

Clearly, the Full grammars use all of the information available in a pair of role values when parsing a sentence; whereas, the other variants relax constraints by selectively ignoring some of that information. There are a number of benefits of the ARV/ARVP representation of C. Because the ARV and ARVP spaces are finite sets that can be partitioned into positive and negative ARVs and ARVPs, respectively, a grammar's constraints can be expressed by enumerating the positive ARVs and ARVPs from labeled sentences. Also, C is PAC-learnable from positive examples [8, 11]. ARV/ARVP constraints can be enforced by using a fast hash table lookup to see if the role value (or pair of role values) is allowed (rather

than propagating thousands of constraints). And finally, the ARV/ARVP representation supports the rapid development of a CDG from annotated training sentences.

One issue that requires further investigation is how well can a CDG be learned from a corpus of sentences, that is, is it possible to learn a grammar that generalizes to new sentences without a high degree of spurious ambiguity. In this paper, we will investigate the learning curves and the generality of the six types of grammars resulting from two types of grammar annotation: direct sentence annotation and annotation of sentences enriched with invocations of subgrammars. We will also evaluate the quality of corpus-based grammars given several types of constraint relaxation.

3 Grammar Annotation

To acquire the grammar rules for the RM corpus, we created two CDG treebanks for the sentences in the corpus, one containing annotated sentences and the other containing annotated strings comprised of words and subgrammar invocations. This work was carried out in two steps. First, the sentences were annotated by language experts using the CDG annotation tool SENATOR [12] together with an active learning, selective sampling procedure similar to [10]. This method supports the rapid development of a grammar by obtaining a grammar from a subset of sentences to be annotated and then using that grammar to identify sentences not covered by the grammar. In the end, we were able to simply verify the parses for 1,073 sentences that were parsed but not annotated by an expert using the SENATOR tool.

Next, the annotated sentences were modified using a tool to replace strings of words (and their corresponding annotations) with subgrammar invocations. For example, the sentence *Count ships on April one* is more general if *on April one* is replaced with an invocation of a subgrammar representing the structure of all legal dates. Note that the subgrammar is made up of annotations of phrases like those replaced by the invocation. An annotated sentence containing subgrammar invocations is equivalent to a potentially large set of sentences, permitting all possible relations between the words in the sentence and the subgrammar to be learned at one time. Our intuition is that expanding sentences with subgrammars offers a paradigm that should result in more efficient learning of grammar rules.

4 Evaluating the Grammars

A set of experiments is conducted to compare the quality of the grammars induced directly from sentences to grammars extracted from subgrammar expanded sentences. For this comparison, we used the 2,844 RM sentences together with an additional 246 sentences added to represent conjunction phenomena we wished to model, giving a corpus size of 3,090 sentences for each type of annotation.

4.1 Grammar Quality Given Annotation Set and Extraction Method

In order to compare the two grammar annotation sets, we extracted the six CDG extraction variations from sentence annotations and from subgrammar expanded

sentence annotations for RM. We hypothesize that learning grammars from a corpus of subgrammar expanded sentences will improve grammar generality dramatically. However, this generality could come at the cost of increased spurious ambiguity, and thus decreased grammar precision.

We first compare the sizes of each grammar variation for each type of grammar annotation. As can be seen in Table 1, augmenting sentence annotations with subgrammars increases grammar size regardless of the extraction method. The Full Mod variation grows much more dramatically in size compared to the other extraction methods. Additionally, the grammars with modifiee constraints grow more quickly than their counterparts without those constraints. These results show that by expanding sentences with subgrammars, we are able to learn more rules. The additional learned rules should translate into better grammar generality.

Table 1. The number of ARVs and ARVPs extracted for each of the grammar extraction methods using all features and all roles given each type of grammar annotation (i.e., sentences (denoted Sentence) and subgrammar expanded sentences (denoted Expanded)).

Extraction	Sentence	Expanded	Increase
Full Mod	143,033	398,562	178.65%
Full	84,921	142,253	67.51%
Feature Mod	25,648	41,459	61.65%
Feature	18,639	24,789	33.00%
Direct Mod	21,468	36,005	67.71%
Direct	14,459	19,335	33.72%
ARVs	4,508	5,551	23.14%

Next, we examined the learning rates of sentence grammars and subgrammar expanded sentence grammars for the RM corpus, which are depicted in Figure 2. For this experiment, we randomly selected sets of sentences representing $10, 20, \ldots, 100$ percent of the corpus, sampling three times for each percentage. The same training sets were used to evaluate each type of annotation; the only difference was in whether or not the sentences contained subgrammar invocations. The grammars extracted directly from sentences do not learn the rules of the grammar as quickly as the subgrammar expanded sentences. Furthermore, the six sentence grammar variations show greater differences in their learning rates compared to the subgrammar expanded sentence grammar variations. This suggests that expanding sentences with subgrammars enables a grammar to be learned more rapidly than using sentences directly, even though the number of rules to learn is larger.

Next, we used the same random training sets described above to evaluate the generality of the grammars learned using $10, 20, \ldots, 90$ percent of the corpus as the training set and the unseen sentences (the remaining $90, 80, \ldots, 10$ percent of the corpus, respectively) as the testing set. We measured the average percentage of the test sentences successfully parsed for each training set size, and the results are shown in Figure 3. Clearly, the subgrammar expanded annotations

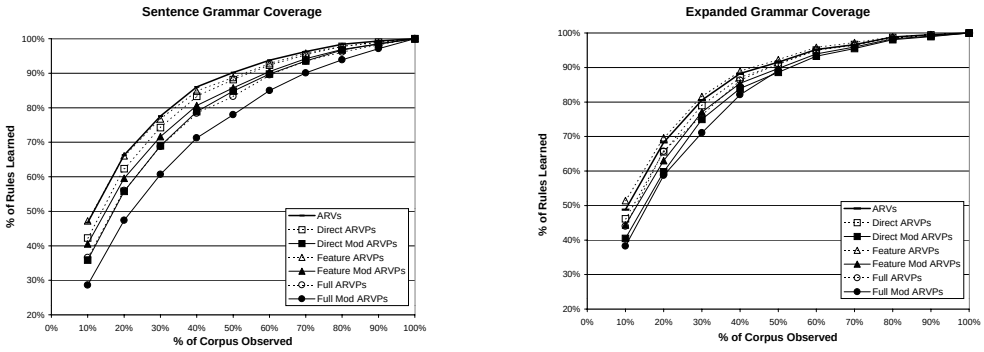

Fig. 2. Grammar coverage for sentence grammars and subgramar expanded sentence grammars.

generalize much better to the unseen test set than the sentence only annotations. Furthermore, the differences between the six grammar extraction methods are reduced by using subgrammars.

To further evaluate the quality of the sentence and subgrammar expanded sentence grammars, we tested them on a set of 4,946 sentences (distinct from the 3,090 RM training sentences) that were generated to represent the full range of possible sentences for the Resource Management Task given the underlying grammar model, which we call the **Domain Sentences** for future reference. The coverage of each grammar is shown in columns two and three of Table 2. The subgrammar expanded sentence grammars all parsed over 90% of the test sentences regardless of the extraction method. In contrast, the sentence grammars had coverages ranging from a low of 33.83% for Full Mod to a high of 76.67% for Direct. The coverage of a hand-written conventional CDG designed to cover the RM corpus sentences was 98.63%, which is comparable to the coverage attained by all but one of the subgrammar expanded corpus-based grammars. It appears that using 3,090 annotated sentences alone is insufficient for obtaining a grammar that uses a full set of grammar features with a coverage of the conventional CDG. The sentence grammars with modifiee constraints are clearly less general than their counterparts without them, and the Full grammars generalize quite poorly. The Feature and Direct sentence grammars are more general because they relax constraints when two role values are not directly related by a modifiee link.

Although the CDGs extracted from subgrammar expanded sentences have a much greater generality than any of the sentence CDGs, they may have achieved that generality at the cost of increased parse ambiguity. Hence, we also measured the parse ambiguity for each method, which is shown in columns four and five of Table 2. Although ambiguity increases as a result of using subgrammar expanded sentence annotations, the increase is quite minor for most of the grammar extraction methods, except for Direct. Note that the average parse ambiguity

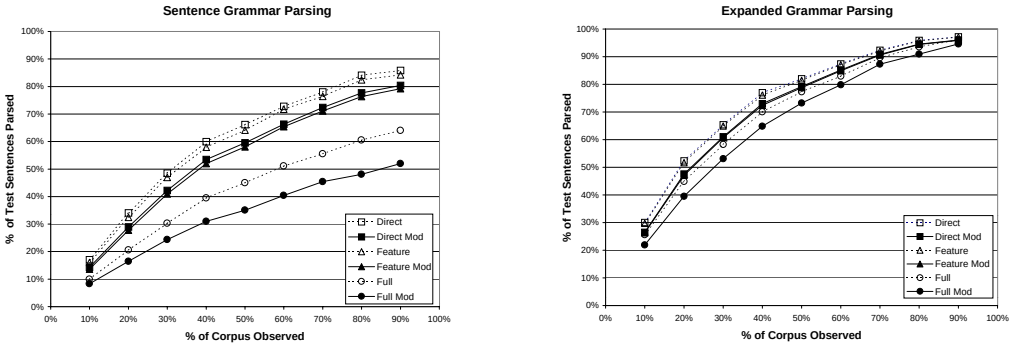

Fig. 3. Grammar generality of sentence grammars and subgramar expanded sentence grammars to unseen sentences.

for the Direct variant is still smaller than the average parse ambiguity for the hand-written CDG, which was 3.52.

Table 2. Percentage of Domain Sentences parsed and average parse ambiguity for each grammar extraction method and each type of grammar annotation (i.e., sentences (denoted Sentence) and subgrammar expanded sentences (denoted Expanded).

	% Domain Sentences Parsed		Average Parse Ambiguity	
Extraction	**Sentence**	**Expanded**	**Sentence**	**Expanded**
Full Mod	33.83%	94.74%	1.00	1.01
Full	53.30%	98.16%	1.01	1.04
Feature Mod	64.68%	97.86%	1.02	1.05
Feature	74.61%	99.17%	1.08	1.52
Direct Mod	66.05%	97.94%	1.17	1.25
Direct	76.67%	99.20%	1.35	2.49
Conventional	98.63%		3.52	

The subgrammar expanded sentence grammars provide better coverage of the grammar with a lower percentage of the corpus observed and generalize better to unseen test sentences than their corresponding sentence grammars, most with only a minor increase in ambiguity. The success of this grammar development methodology suggests that it would be effective for developing other domain-specific grammars (e.g., radiology reports) for speech recognition tasks and for intelligently increasing the size of a training set for developing a high quality stochastic CDG grammar. Even though clearly the best grammars are obtained by making the extra effort to identify and use subgrammars within sentences in order to derive grammars with sufficient coverage, this approach requires more effort than simply learning grammars from sentences.

4.2 Grammar Quality Given Constraint Relaxation

Because annotating sentences is more convenient than identifying and using subgrammars in the annotation process, we also investigate the effectiveness of more direct methods for improving the coverage of the grammars extracted directly from sentences. Each of the methods we investigate here involves constraint relaxation in that we selectively ignore some information available in the annotated sentences when deriving the grammar (note that the Feature and Direct grammars are already partially relaxed). An important question concerns whether these methods can increase generality without increasing grammar ambiguity exorbitantly. We will investigate the properties of grammars that result from several types of constraint relaxation:

1. **Relax Grammar Degree:** We will measure the generality and ambiguity of grammars that use only the governor role (as in traditional dependency grammars).
2. **Relax Specific Features:** If all features within the annotations are ignored when extracting a grammar, that grammar would be extremely general but also highly ambiguous. One of the most undertrained features in our corpus is semantics, so we examine the generality of grammars obtained from annotations ignoring semantics rather than one or more of the other features.
3. **Relax Grammar Degree and Specific Features:** We will also measure the generality of grammars that use only the governor role and ignore the semantics feature.
4. **Relax All Features in Certain Cases:** Recall that Feature and Direct grammars ignore all features when matching a pair of role values that are not directly linked together by a modifiee relation. However since $\mathrm{M}x_1 = \mathrm{M}x_2$ is fairly indirect, we will investigate the effect of ignoring features when only this modifiee relation is true.

Additional grammar relaxations could have been investigated; however, as the reader will observe, the above methods are sufficient to obtain extremely high levels of parse ambiguity for several of our grammar extraction methods.

For each relaxation, we extracted grammars using all six grammar extraction methods and both types of annotation and then counted the number of ARVs and ARVPs. Since the number of ARVs is constant across all grammar extraction methods, we report the number of ARVs separately in Table 3. Each grammar relaxation method reduces the the number of ARVs in the grammar relative to its baseline grammar except for the method that eliminates $\mathrm{M}x_1{=}\mathrm{M}x_2$ as a direct relation for ARVPs (which does not affect the number of ARVs). Figure 4 depicts the number of ARVPs in the baseline grammars described in the previous subsection compared to the relaxed forms of those grammars. The sizes of the six grammars extracted from subgrammar expanded sentence annotations are depicted using solid lines, while the sizes of the six extracted directly from sentence annotations are depicted using dashed lines. The sizes of the baseline grammars are depicted for each annotation method using bold lines (bold dashed for the sentence only grammar). Notice that the number of ARVPs is reduced

significantly by most of the relaxation methods except for the method that relaxes Mx_1=Mx_2, which has very little impact on the number of ARVPs in the Full grammar variations.

Table 3. The number of ARVs extracted for each grammar extraction given the type of annotation and indicated relaxation.

Relaxation	Sentence	Expanded
Original	4,508	5,551
Governor	2,019	2,264
No Semantics	2,413	3,070
No Semantics & Gov	1,029	1,217
Relax Mx_1=Mx_2	4,508	5,551

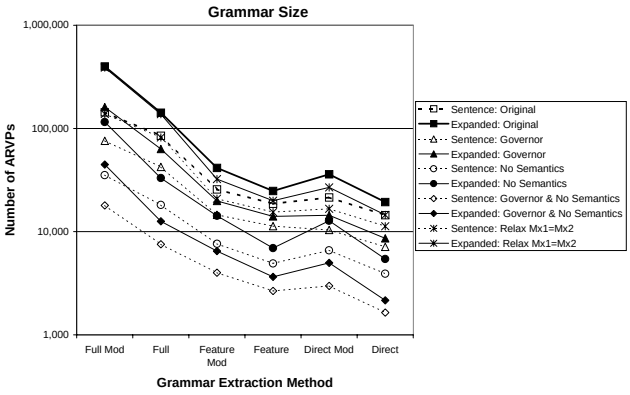

Fig. 4. Number of ARVPs for each the grammar extraction method given the type of annotation and indicated relaxation.

Next we evaluate the generality of the relaxed grammars by using them to parse the Domain Sentences. The coverage of each grammar is shown in Figure 5 and the ambiguity in Figure 6. Both the expanded and sentence grammars have increased generality given each type of relaxation, with the best across-the-board improvement occurring when only the governor role is used together with relaxing the semantic feature. However, this generality comes at the cost of a very high parse ambiguity, especially for the Direct, Direct Mod, and Feature grammars. For both the sentence and expanded grammars, relaxing to a single role improves the grammars without modifiee constraints the most. This may result from the fact that modifiee information provides an alternative way of capturing the information contained in a multiple role encoding of a parse. Also, note that the relaxation of Mx_1 = Mx_2 does not affect the coverage of the Full grammar variations and only minimally affects the other grammars with modifiee constraints. The most beneficial single relaxation for generality involves ignoring semantics, with the Feature grammar offering the best trade-off of generality

and ambiguity for a sentence grammar. The baseline expanded grammars have a higher generality and lower parse ambiguity than the corresponding relaxed sentence grammars; however, even the expanded grammars are improved slightly by the relaxations. On the other hand, the expanded grammars suffer a greater degree of ambiguity from the relaxation process for only a very small increase in generality.

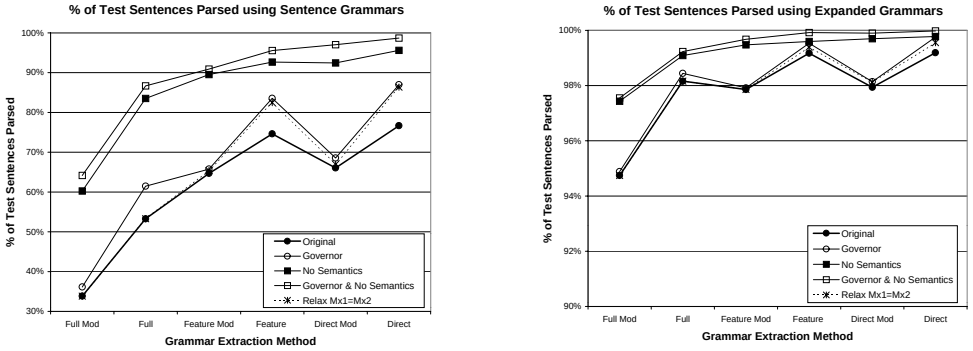

Fig. 5. Percentage of **Domain Sentences** parsed for each grammar extraction type given the type of annotation and indicated relaxation.

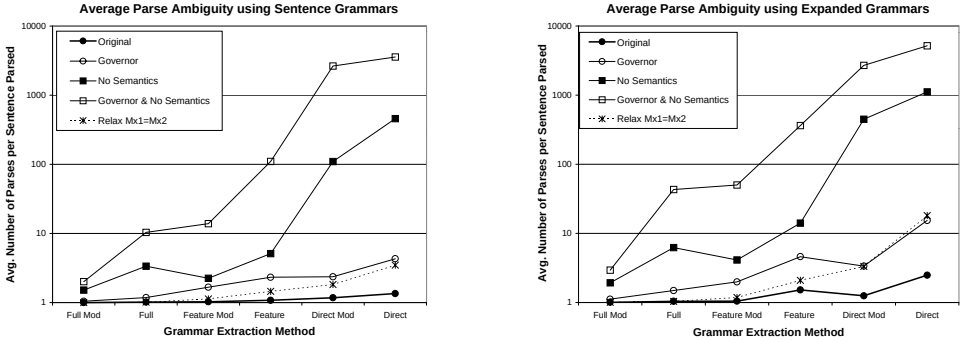

Fig. 6. Parse ambiguity of **Domain Sentences** for each grammar extraction type given the type of annotation and indicated relaxation.

5 Conclusion and Acknowledgements

In this paper, we have investigated the effect that two types of annotation, six grammar extraction methods, and three additional types of constraint relaxation have on the quality of CDGs learned from corpora. Empirical tests

suggest that grammars generated directly from sentences have a low degree of parse ambiguity, a slow learning rate, and poor grammar generality. Grammars derived from subgrammar expanded sentence annotations show a dramatically improved learning rate and generality with very little increase in parse ambiguity. Relaxation of sentence grammars on degree, semantics, and direct link types increases the generality but at a cost of a higher degree of ambiguity. Relaxing the subgrammar expanded sentence grammars dramatically boosts parse ambiguity with a very small increase in generality. This suggests that subgrammar expanded sentence grammars well represent the language of the domain.

This research was supported by Intel, Purdue Research Foundation, and the National Science Foundation under Grant No. IRI 97-04358, CDA 96-17388, and #9980054-BCS.

References

1. J. Hajic. Building a syntactically annotated corpus: The prague dependency treekbank. In *Issues of Valency and Meaning (Festschrift for Jarmila Panevova)*, pages 106–132. Carolina, Charles University, Prague, 1998.
2. J. Hajic, E. Brill, M. Collins, B. Hladka, D. Jones, C. Kuo, L. Ramshaw, O. Schwartz, C. Tillmann, and D. Zeman. Core natural language processing technology applicable to multiple languages – Workshop '98. Technical report, Johns Hopkins Univ., Center for Language and Speech Processing, Baltimore, MD, 1998.
3. M. P. Harper and R. A. Helzerman. Extensions to constraint dependency parsing for spoken language processing. *Computer Speech and Language*, 9:187–234, 1995.
4. M. P. Harper, S. A. Hockema, and C. M. White. Enhanced constraint dependency grammar parsers. In *Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing*, 1999.
5. M. P. Harper, C. M. White, W. Wang, M. T. Johnson, and R. A. Helzerman. Effectiveness of corpus-induced dependency grammars for post-processing speech. In *Proceedings of the 1st Annual Meeting of the North American Association for Computational Linguistics*, 2000.
6. H. Maruyama. Constraint Dependency Grammar. Technical Report #RT0044, IBM, Tokyo, Japan, 1990.
7. H. Maruyama. Structural disambiguation with constraint propagation. In *The Proceedings of the Annual Meeting of Association for Computational Linguistics*, pages 31–38, 1990.
8. B. Natarajan. On learning sets and functions. *Machine Learning*, 4(1), 1989.
9. P. J. Price, W. Fischer, J. Bernstein, and D. Pallett. A database for continuous speech recognition in a 1000-word domain. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 651–654, 1988.
10. Cynthia A. Thompson, Mary Elaine Califf, and Raymond J. Mooney. Active learning for natural language parsing and information extraction. In *Proceedings of ICML-99*, pages 406–414, July 27–30 1999.
11. L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
12. Christopher M. White. *Rapid Grammar Development and Parsing: Constraint Dependency Grammars with Abstract Role Values.* PhD thesis, 2000. Purdue University, School of Electrical and Computer Engineering, West Lafayette, IN.