Research on Language and Computation (2005) 3:83–105
DOI 10.1007/s11168-005-1288-y

Stochastic HPSG Parse Disambiguation Using the Redwoods Corpus

KRISTINA TOUTANOVA, CHRISTOPHER D. MANNING, DAN FLICKINGER[1] and STEPHAN OEPEN[1]
Department of Computer Science, Stanford University, Stanford, CA 94305-9040, U.S.A. (E-mail: {kristina, manning}@cs.stanford.edu); [1]*Center for the Study of Language and Information, Stanford University, Ventura Hall, Stanford, CA 94305-4115, U.S.A. (E-mail: {dan, oe}@csli.stanford.edu)*

Abstract. This article details our experiments on HPSG parse disambiguation, based on the Redwoods treebank. Using existing and novel stochastic models, we evaluate the usefulness of different information sources for disambiguation – lexical, syntactic, and semantic. We perform careful comparisons of generative and discriminative models using equivalent features and show the consistent advantage of discriminatively trained models. Our best system performs at over 76% sentence exact match accuracy.

Key words: HPSG grammars, statistical parsing models

1. Introduction

This article presents probabilistic models that try to select the correct analysis for a sentence based on statistics gathered from the Redwoods HPSG treebank (Oepen et al., 2002, 2004, in press). Head-driven Phrase Structure Grammar (HPSG), the grammar formalism underlying the Redwoods corpus, is a modern constraint-based lexicalist (or "unification") grammar formalism, particularly noted for its concern for broad descriptive adequacy, precise formal specification, and close linking between syntax and semantic interpretation.[1] As a strong competence theory (Kaplan and Bresnan, 1982), there has been considerable emphasis within HPSG on providing a syntactic model that can plausibly support an account of human sentence processing, and in particular, there has been discussion of the *resolution problem* of how the many sources of linguistic and contextual evidence are brought to bear in real time to decide a sentence's interpretation. However, while HPSG has enjoyed great success in developing implementable syntactic theories of broad empirical reach which give precise semantic interpretations, there has been relatively little progress in solving the ambiguity resolution problem – despite the fact that an effective solution is necessary

for successfully applying HPSG in practical NLP systems, for anything but the most restricted domains. In this paper we show how this problem can be effectively approached by using probabilistic models of evidence integration, and how even models that consider only sentence-internal context can be quite effective in solving most structural and interpretive ambiguities. Our emphasis here is on the engineering side, where the goal is simply to disambiguate effectively so as to determine the correct sentence interpretation, but to the extent that these models are successful, they lend support to recent work in psycholinguistics which has also explored probabilistic models of sentence interpretation (MacDonald, 1994; Trueswell, 1996).

Examining the nature of the HPSG parse disambiguation problem, the fine-grained representations found in the Redwoods treebank raise novel issues relative to more traditional treebanks such as the Penn treebank (Marcus et al., 1993), which have been the focus of most past work on probabilistic parsing, e.g. (Collins, 1997; Charniak, 1997). The Redwoods treebank makes available a variety of rich representations. Information in HPSG is represented by a *sign*, a typed feature structure which represents phonological, syntactic, and semantic information about a word or phrase. This information is built up for a sentence compositionally from the signs of sentence parts. We have not used the full HPSG sign in our current models, but rather a number of simpler projections of the sign and how it was composed. Most similar to Penn treebank parse trees are phrase structure trees projected from the sign (Figure 1b), but in this work we have concentrated on use of derivation trees (Figure 1a), which record the combining rule schemas of the HPSG grammar which were used to license the sign by combining initial lexical types.[2] The internal nodes represent, for example, head-complement, head-specifier, and head-adjunct schemas, which were used to license larger signs out of component parts. These derivation trees hence provide significantly different information from conventional phrase structure trees, but have proven to be quite effective for disambiguation. These representations are more fine-grained than those familiar from the Penn treebank: for example, rather than 45 part-of-speech tags and 27 phrasal node labels, we have about 8000 lexical item identifiers, and 70 derivational schemas. The lexical types are more similar to those employed in Lexicalized Tree-Adjoining Grammar work (Srinivas and Joshi, 1999), encoding information such as verbal subcategorization.

Another important difference between the Penn Treebank and the Redwoods corpus is that the former has only an implicit grammar given by the observed trees, while all parses in the Redwoods corpus are licensed by an explicit, and much more constraining, HPSG grammar. In this sense, and because of the similarity between HPSG and LFG, our work is much more similar to parse disambiguation work for LFG grammars (Riezler et al., 2002). Additionally, the (implicit) Penn treebank grammar and the LinGO

ERG (English Resource Grammar) differ in that the Penn treebank often uses quite flat grammatical analyses while the ERG is maximally binary, with extensive use of unary schemas for implementing morphology and type-changing operations. Much common wisdom that has been acquired for building probabilistic models over Penn treebank parse trees is implicitly conditioned on the fact that the flat representations of the Penn treebank trees mean that most important dependencies are represented jointly in a local tree. Thus lessons learned there may not be applicable to our problem (see Collins, 1999 for a careful discussion of this issue). Our results should inform other in progress efforts at constructing HPSGbased treebanks, such as the Polish treebank (Marciniak et al., 1999) and the Bulgarian HPSG treebank (Simov et al., 2002).

Finally, the HPSG signs provide deep semantic representations for sentences: together with the syntactic analyses of constituents, an underspecified minimal recursion semantics (MRS) representation (Copestake et al., 1999) is built up. This semantic information, unavailable in the Penn treebank, may provide a useful source of additional features, at least partially orthogonal to syntactic information, for aiding parse disambiguation. Again, so far we have not used the full MRS structures but rather a semantic dependency tree, which projects a portion of the semantic information.

On the one hand, the richer analyses available have the potential to provide more information to ease parse disambiguation; on the other hand, the finer grain requires finer levels of structure and meaning disambiguation and raises increased data sparsity issues, especially since the corpus available to us is far smaller than the Penn treebank. It is thus unclear a priori how the unique aspects of the HPSG representations will affect performance on the parse disambiguation task. In this work, we have explored building probabilistic models for parse disambiguation using this rich HPSG treebank, assessing the effectiveness of different kinds of information. We present generative and discriminative models using analogous features and compare their performance on the disambiguation task. Among the results that we obtain are:

- Lexical information alone accounts for only half of the parse ambiguity inherent in the corpus, providing an upper bound on parse disambiguation via tagging, which we approach within a few percent. That is, supertagging (Srinivas and Joshi, 1999) alone is not effective in this domain.
- Using multiple sources of information, in particular, adding semantic information, can synergistically improve parse disambiguation performance.
- Conditional models achieve up to 28% error reduction over generative models.

– Our models achieve quite high overall parse disambiguation performance, as much as 76.7% exact match parse selection accuracy on ambiguous sentences in the corpus.
– Of the remaining errors, about 62% are real errors, in which the treebank is right and model is wrong. The major part of these errors are due to PP and other modifier attachment ambiguities.

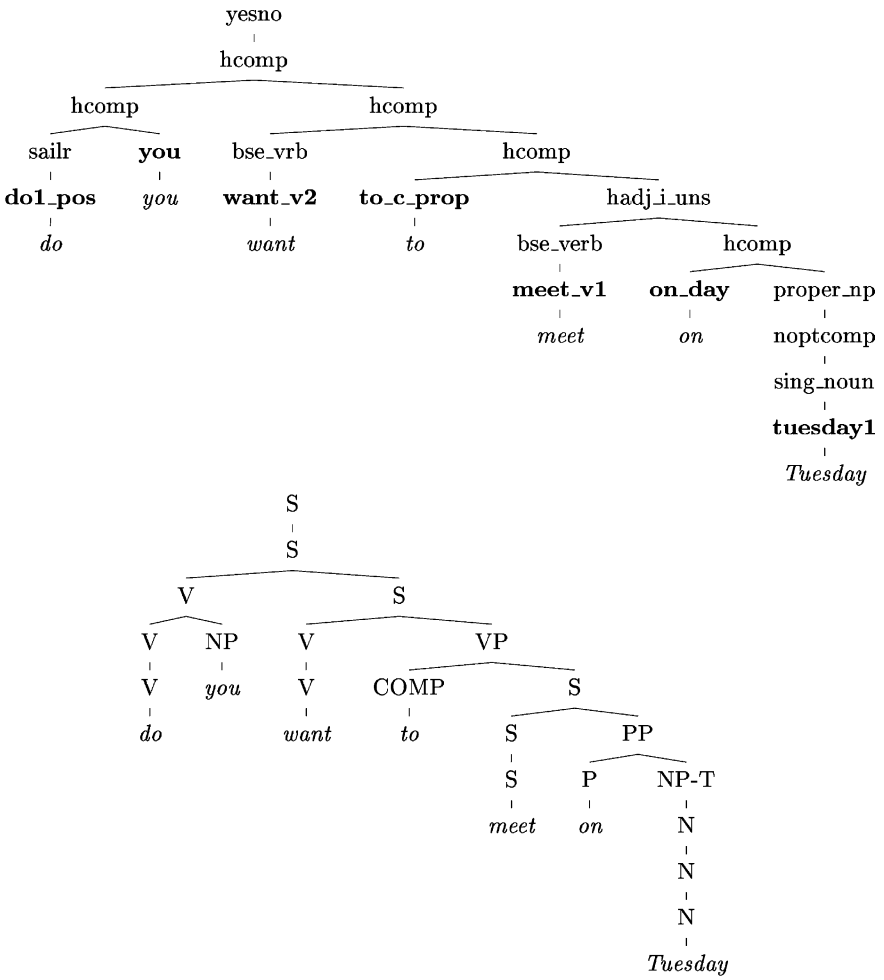
In the sections that follow, we describe the various statistical models we test, provide experimental results on the parse disambiguation task, and provide some preliminary error analysis.

2. Overview of Models

A variety of approaches are possible for building statistical models for parse disambiguation. The Redwoods treebank makes available exhaustive HPSG sign representations for all analyses of sentences. These are large attribute-value matrices which record all aspects of a sentence's syntax and semantics. We have concentrated on using small subsets of these representations. In our initial experiments we built a tagger for the HPSG lexical item identifiers in the treebank, and report results on using the tagger for parse disambiguation. Subsequent models included modeling of tree structures. We have explored training stochastic models using derivation trees, phrase structure trees, and semantic trees (which are approximations to the MRS representation). Figure 1 shows examples of a derivation tree, phrase structure tree and an elementary dependency graph. The learned probabilistic models were used to rank possible parses of unseen test sentences according to the probabilities they assign to them.

Most probabilistic parsing research is based on branching process models (Harris, 1963). The HPSG derivations that the treebank makes available can be viewed as such a branching process, and a stochastic model of the trees can be built as, for instance, a probabilistic context-free grammar (PCFG) model. Abney (1997) notes problems with the soundness of the approach, showing that the distribution of derivations of a unification-based grammar may well not be in the class of PCFG grammars defined using its context-free base. He motivates the use of log-linear models (Agresti, 1990) for parse ranking that Johnson and colleagues further developed (Johnson et al., 1999). Building conditional log-linear models is also expected to improve generalization performance because the criterion being optimized is discriminative (Vapnik, 1998; Ng and Jordan, 2002 Klein and Manning, 2002).

In this work we have experimented with both generative and conditional log-linear models over the same feature sets and we report results achieved using both kinds of models. We examine the performance of five



_4:{
_4:int_rel[SOA e2:_want2_rel]
e2:_want2_rel[ARG1 x4:pron_rel, ARG4 _2:hypo_rel]
_1:def_rel[BV x4:pron_rel]
_2:hypo_rel[SOA e18:_meet_v_rel]
e18:_meet_v_rel[ARG1 x4:pron_rel]
e19:_on_temp_rel[ARG e18:_meet_v_rel, ARG3 x21:dofw_rel]
x21:dofw_rel[NAMED :tue]
_3:def_np_rel[BV x21:dofw_rel] }

Figure 1. Native and derived Redwoods representations for the sentence *Do you want to meet on Tuesday?* – (a) derivation tree using unique rule and lexical item (in bold) identifiers of the source grammar (top), (b) phrase structure tree labelled with user-defined, parameterizable category abbreviations (center), and (c) elementary dependency graph extracted from the MRS meaning representation (bottom).

models: an HMM tagging model, a simple PCFG, a PCFG with ancestor annotation where the number of ancestors was selected automatically, a model of semantic dependencies, and a hybrid model that combines predictions from several of the above models. For these models we also trained corresponding conditional log-linear models using the same information sources as the generative models.

These models will be described in more detail in the next section. We first describe the generative models and after that their corresponding conditional log-linear models.

3. Generative Models

3.1. Tagger

The tagger we implemented is a standard trigram HMM tagger, defining a joint probability distribution over the preterminal sequences and yields of the derivation trees. The preterminals of the derivation trees are the lexical item identifiers. They are displayed in bold in Figure 1a.

Trigram probabilities are smoothed by linear interpolation with lower-order models, using Witten–Bell smoothing with a varying parameter d (Witten and Bell, 1991). The general formulation for deleted interpolation based on linear context subsets is:

$$\begin{aligned}\tilde{P}(y|x_1,\ldots,x_n) = {} & \lambda_{x_1,\ldots,x_n}\hat{P}(y|x_1,\ldots,x_n) \\ & +(1-\lambda_{x_1,\ldots,x_n})\tilde{P}(y|x_1,\ldots,x_{n-1})\end{aligned}$$

and the Witten–Bell method we used defines the interpolation weights as

$$\lambda(x_1,\ldots,x_n) = \frac{c(x_1,\ldots,x_n)}{c(x_1,\ldots,x_n)+d\times|y:c(y,x_1,\ldots,x_n)>0|}.$$

The lexical item identifiers shown in Figure 1 are organized into about 500 lexical types, which are themselves placed in the HPSG type hierarchy. The lexical types are not shown in the figure. They are the direct supertypes of the lexical items. For example, the lexical type of **meet_v1** in the figure is *v_unerg_le*, and the lexical type of **want_v2** is *v_subj_equi_le*. Our tagging model does not take advantage of the lexical types or the type hierarchy in which they are organized and we plan to pursue incorporating this information in future models.

3.2. PCFG models over derivation trees

The PCFG models define probability distributions over the trees of derivational types corresponding to the HPSG analyses of sentences. A PCFG

model has parameters $\theta_{i,j}$ for each rule $A_i \rightarrow \alpha_j$ in the corresponding context-free grammar ($\theta_{i,j} = P(\alpha_j|A_i)$).[3] In our application, the nonterminals in the PCFG A_i are schemas of the HPSG grammar used to build the parses (such as HEAD-COMPL or HEAD-ADJ). We set the parameters to maximize the likelihood of the set of derivation trees for the preferred parses of the sentences in a training set. In further discussion we will refer to this simple PCFG model as PCFG-1P.

PCFG models can be made better if the rule applications are conditioned to capture sufficient context. For example, grandparent annotation for PCFGs has been shown to significantly improve parsing accuracy (Charniak and Carroll, 1994; Johnson, 1998). One feature of the LinGO ERG is that it is binarized and thus it is even more important to make probabilistic models aware of a wider context. We implemented several models that condition additionally on the parent, grandparent, etc. of the current node. Model PCFG-2P uses the current node's parent and has parameters $\theta_{<i_1,i_2>,j}$ for each rule $A_{i_1} : A_{i_2} \rightarrow \alpha_j$ in the derivation trees. Here A_{i_2} denotes the label of a node, and A_{i_1} the label of its parent. Similarly model PCFG-3P conditions on the node's parent and grandparent. For estimation of the local expansion probabilities, these models use linear interpolation of estimates based on linear subsets of the conditioning context. The interpolation coefficients were obtained using Witten–Bell smoothing as for the tagger.

An interesting issue is how many levels of parenting are optimal and how to learn that automatically.

We implemented an extended PCFG that conditions each node's expansion on up to five of its ancestors in the derivation tree. Our method of ancestor selection is similar to learning context-specific independencies in Bayesian networks (Friedman and Goldszmidt, 1996). In particular, we use decision tree representation of the distribution $P(\alpha_j|context)$ where $context$ contains five ancestors. We experimented with growing the decision tree according to an MDL criterion and gain ratio and the growing algorithm did not make a noticeable difference. The final probability estimates were linear interpolations of relative frequency estimates in a decision tree leaf and all nodes on the path to the root, as in Magerman (1995). The interpolation coefficients were again estimated using Witten–Bell smoothing. We will refer to the PCFG model with ancestor information as PCFG-A.

3.3. PCFG Models over semantic dependency trees

We also learned PCFG-style models over trees of semantic dependencies extracted from the HPSG signs. These semantic models served as an early experiment in using semantic information for disambiguation. We intend as work progresses to build stochastic models over the elementary dependency

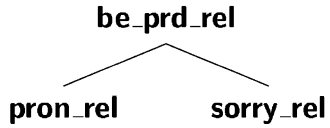


Figure 2. Semantic dependency tree for the sentence: *I am sorry*.

graphs extracted from MRS meaning representations shown in Figure 1 but for the moment keep to tree representations. The semantic trees mirror the derivation trees. They were obtained in the following manner: Each node in the derivation tree was annotated with its key semantic relation (Copestake et al., 1999). Consequently the annotated tree was flattened so that all dependents of a semantic relation occur at the same level of the tree as its direct descendants. Figure 2 shows the semantic dependency tree for the sentence: *I am sorry.*

The probability of a semantic dependency tree was estimated as a product of the probabilities of local trees, such as the one shown in Figure 2. For these trees the expansion of a node is viewed as consisting of separate trials for each dependent. Any conditional dependencies among children of a node can be captured by expanding the history. The probability of generating a dependent in the semantic dependency trees is estimated given a context of five conditioning features. These were, the parent of the node (the parent is the head and the node is the dependent), the direction (left of right), the number of dependents already generated in the surface string between the head and the dependent, the grandparent label, and the label of the immediately preceding dependent. This model is a slight modification of the model over semantic dependency trees described in (Toutanova and Manning, 2002).

More specifically, the model for generation of semantic dependents to the left and right is as follows: first the left dependents are generated from right to left given the head, its parent, right sister, and the number of dependents to the left that have already been generated. After that, the right dependents are generated from left to right, given the head, its parent, left sister and number of dependents to the right that have already been generated. We also add stop symbols at the ends to the left and right. This model is very similar to the markovized rule models in Collins (1997). For example, the joint probability of the dependents of *be_prd_rel* in the above example would be:

$$P(pron\_rel|be\_prd\_rel, left, 0, top, none) \times$$
$$P(stop|be\_prd\_rel, left, 0, top, pron\_rel) \times$$

$$P(sorry\_rel|be\_prd\_rel, right, 0, top, none) \times$$
$$P(stop|be\_prd\_rel, right, 1, top, sorry\_rel)$$

The amount of conditioning context for this phase was chosen automatically similarly to PCFG-A, using a decision tree growing algorithm. Final probability estimates were obtained at the decision tree leaves using Witten–Bell smoothing as for the other models. In further discussion we will refer to the model of semantic dependencies as PCFG-Sem.

3.4. Model combination

We explored combining the predictions from the PCFG-A model, the tagger, and PCFG-Sem. The combined model computes the scores of analyses as linear combinations of the log-probabilities assigned to the analyses by the individual models. Since some of the factors participating in the tagger also participate in the PCFG-A model, in the combined model we used only the trigram tag sequence probabilities from the tagger. These are the transition probabilities of the HMM tagging model.

More specifically, for a tree t,

$$Score(t) = \log(P_{PCFG\text{-}A}(t)) + \lambda_1 \log(P_{TRIG}(tags(t))) + \lambda_2 \log(P_{PCFG\text{-}Sem}(t)),$$

where $P_{TRIG}(tags(t))$ is the probability of the sequence of preterminals $t_1, \ldots, t_n$ in t according to a trigram tag model:

$$P_{TRIG}(t_1 \ldots t_n) = \prod_{i=1}^{n} P(t_i|t_{i-1}, t_{i-2})$$

with appropriate treatment of boundaries. The trigram probabilities are smoothed as for the HMM tagger. The combination weights λ_1 and λ_2 were not fitted extensively. The performance of the model was stable under changes of the value of λ_1 in the range 0.2 to 1, whereas the performance of the combination went down if λ_2 was set to a value above 0.5. We report results using values $\lambda_1 = 0.5$ and $\lambda_2 = 0.4$.

4. Conditional Log-linear Models

A conditional log-linear model for estimating the probability of an HPSG analysis given a sentence has a set of features $\{f_1, \ldots, f_m\}$ defined over analyses and a set of corresponding weights $\{\lambda_1, \ldots, \lambda_m\}$ for them. In this work we have defined features over derivation trees and semantic trees as described for the branching process models.

For a sentence s with possible analyses $t_1, \ldots, t_k$, the conditional probability for analysis t_i is given by

$$P(t_i|s) = \frac{\exp \sum_{j=1,\ldots,m} f_j(t_i)\lambda_j}{\sum_{i'=1,\ldots,k} \exp \sum_{j=1,\ldots,m} f_j(t_{i'})\lambda_j}.$$

As described by Johnson et al. (1999), we trained the model by maximizing the conditional likelihood of the preferred analyses and using a Gaussian prior for smoothing (Chen and Rosenfeld, 1999). In particular, the objective being maximized was the following:

$$L(D, \Lambda) = \sum_{i=1,\ldots,n} \log(p(t_1|s_i)) - \frac{1}{2\sigma^2} \sum_{j=1,\ldots,m} \lambda_j^2.$$

Here D is the training data set and i ranges over all sentences; t_1 denotes the correct analysis for a sentence; σ is the standard deviation of the Gaussian prior. We experimented with several values for σ and small values were better. In the following experiments, $\sigma = 1$ was used for all models.

We used the conjugate gradient method for optimization. In our experiments including only features with observed frequency more than a threshold was not advantageous and therefore we include all occurring features.

In Riezler et al. (2002) similar log-linear models are used, but only partial annotation of correct analyses is available and therefore they optimize the conditional likelihood of consistent parses. The features (properties) they use include counts of c-structure subtrees, nodes, and f-structure attributes. More complex features indicating attachment, branching, and (non) parallelism of coordinations are also included. Lexical features are obtained using a clustering model.

Our log-linear models have features exactly corresponding to the five generative models described in the previous section. The following subsections give more detail on the correspondence between the generative and discriminative models.

4.1. Tagger

The conditional tagging modes LTrigram includes features for all lexical item trigrams, bigrams and unigrams.

4.2. Models over derivation trees

The conditional log-linear model LPCFG-1P, corresponding to PCFG-1P has one feature for each expansion of each nonterminal in the derivation trees $A_i \rightarrow \alpha_j$. For a derivation tree t, it has as value the number of times

this expansion occurs in the tree. For every expansion $A_i \rightarrow \alpha_j$, PCFG-1P has a parameter $\theta_{ij} = P(\alpha_j|A_i)$, and LPCFG-1P has a parameter λ_{ij}. The number of parameters is thus the same in PCFG-1P and LPCFG-1P. The probabilities that both models assign to a parse tree t are proportional to the product of parameters corresponding to occurring productions. The difference is that the weights λ_{ij} are estimated by maximum conditional likelihood and not relative frequency (maximum joint likelihood).

The model LPCFG-A corresponds to the generative model PCFG-A. The features of LPCFG-A were defined using the same decision trees induced for PCFG-A. A feature was added for every path in the decision tree (both to a leaf and internal node), and every expansion occurring at that node. The feature will be active at a node n in a derivation tree, if the feature values specified by this decision tree path are the same for n, and if the expansion at n is the same as for the feature. The feature values for a derivation tree are sums of the feature values at local trees. Thus LPCFG-A uses a generative component for feature selection and is not purely discriminative in construction. The correspondence between LPCFG-A and PCFG-A is not as direct as that between LPCFG-1P and PCFG-1P. PCFG-A uses relative frequency estimates of expansion probabilities at the decision tree leaves and internal nodes, obtaining final estimates via linear interpolation. LPCFG-A also has feature parameters for leaves and internal nodes, but it multiplies the parameters.

4.3. Models over semantic dependency trees

The model LPCFG-Sem corresponds to PCFG-Sem and its features were defined using decision trees induced by PCFG-Sem in the same was as LPCFG-A was defined using PCFG-A.

4.4. Model combination

The combination LCombined is a log-linear model including the features of LPCFG-A, LPCFG-Sem, and LTagger. Therefore the method of combination is different for the conditional and generative models. The generative models were combined by taking a weighted log-sum of the probabilities they assign to trees. The conditional models were instead combined by collecting all features into one model. This would be similar to the combination in the generative models case if we could view the semantic trees, derivation trees, and tag sequences as independent portions of the analyses.

5. Experimental Results

We report parse disambiguation results on the dataset described in Table I. The table lists the characteristics of the 3rd Growth of the Redwoods treebank (Oepan et al., 2004, in press). It is the most recent growth, using the new October 2002 version of the ERG. The sentences listed here have exactly one preferred analysis and are not marked as ungrammatical. We have listed statistics for all sentences (ambiguous and unambiguous), and ambiguous only. In testing, we only consider ambiguous sentences, while unambiguous ones may be used in training. A previous version of the corpus, the 1st Growth, was used in the experiments reported in the papers (Oepen et al., 2002; Toutanova et al., 2002, 2003b). The 3rd Growth of Redwoods is much more ambiguous than the previous version because of grammar changes and inclusion of highly ambiguous sentences that were initially excluded.

To illustrate the distribution of ambiguity levels in the corpus, as well as the related distribution of number of words per sentence, Figure 3 shows histograms of the number of analyses per sentence in (*b*) and the percentage of sentences by sentence length in (*a*).

All models were trained and tested using 10-fold cross-validation. Each of the 10 folds was formed deterministically, by starting from sentence i, and placing every 10th sentence in the test set. Thus the union of the 10 test sets, for $i = 1, \ldots, 10$ is the complete corpus and they do not overlap. The unambiguous sentences were discarded from the test sets. The generative models use the unambiguous sentences for training, but the conditional log-linear models do not (the unambiguous sentences contribute a constant to the log-likelihood).

Accuracy results denote the percentage of test sentences for which the highest ranked analysis was the correct one. Note that this corresponds to getting the sentence analysis completely right, and is a much more stringent criterion than evaluating the percentage of labelled constituents or dependencies that are correct, as is more commonly done in statistical parsing work. Often the models give the same score to several different parses. In these cases, when a model ranks a set of m parses highest with equal scores

Table I. Annotated corpora used in experiments: the columns are, from left to right, the total number of sentences, average length, and average structural ambiguity

Sentences		Length	Struct ambiguity
All	6876	8.0	44.5
Ambiguous	5266	9.1	57.8

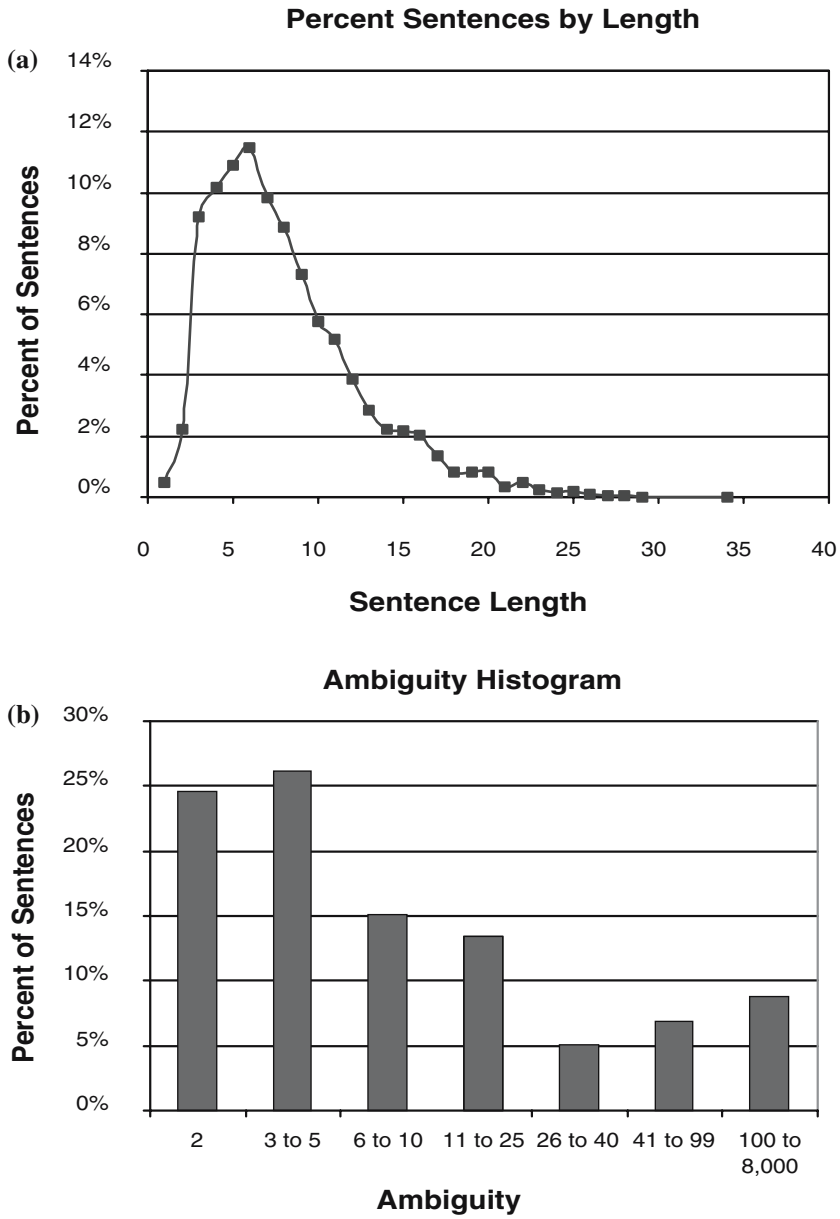


Figure 3. Percentage of sentences by length (*a*) and percentage of sentences by ambiguity level ranges (*b*) for the 3rd Growth of the Redwoods corpus.

Table II. Performance of generative models for the parse selection task (exact match accuracy on ambiguous sentences)

	Method	Accuracy
Random		22.7
Tagger	Trigram	42.1
	Perfect	48.8
PCFG	PCFG-1P	61.6
	PCFG-A	71.0
	PCFG-Sem	62.8
	Combined	73.2

and one of those parses is the preferred parse in the treebank, we compute the accuracy on this sentence as $1/m$. For comparison, a baseline showing the expected performance of choosing parses randomly according to a uniform distribution is included.

Table II shows the accuracy of parse selection using the generative models described in section 3. The results in Table II indicate that high-disambiguation accuracy can be achieved using simple statistical models. The HMM tagger does not perform well on the task by itself in comparison with other models that have more information about the parse. For comparison, we present the performance of a hypothetical clairvoyant tagger that knows the true tag sequence and scores highest the parses that have the correct preterminal sequence. The performance of the perfect tagger shows that, informally speaking, roughly half of the information necessary to disambiguate parses is available in the lexical tags.

Using ancestor information in the PCFG models improved parse ranking accuracy significantly over a simple model PCFG-1P – PCFG-A achieved 24% error reduction from PCFG-1P. The PCFG-Sem model has respectable accuracy but does not by itself work as well as PCFG-A. The performance of model combination shows that the information they explore is somewhat complementary. The tagger adds left-context information to the PCFG-A model (in a crude way) and the PCFG-Sem model provides semantic information.

Table III shows the accuracy of parse selection using the conditional log-linear models. We see that higher accuracy is achieved by the discriminative models. The difference between the generative and conditional log-linear models is largest for the PCFG-1P model and its corresponding LPCFG-1P model (28% error reduction). The difference between the generative and conditional log-linear models for the trigram tagger is small

Table III. Performance of conditional log-linear models for the parse selection task (exact match accuracy on ambiguous sentences)

	Method	Accuracy
Random		22.7
CTagger	Trigram	43.2
	Perfect	48.8
LPCFG	LPCFG-1P	72.4
	LPCFG-A	75.9
	LPCFG-Sem	65.4
	LCombined	76.7

and this result is in agreement with similar results in the literature comparing HMM and conditional log-linear models for part of speech tagging (Klein and Manning, 2002). Overall the gain from using conditional log-linear models for the final combined model is a 13% error reduction from the generative model.

The parse disambiguation accuracy achieved by these models is quite high. However, in evaluating this level of performance we need to take into account the low-ambiguity rate of our corpus and the short sentence length. To assess the influence of ambiguity rate on the parse disambiguation accuracy of our model, we computed average accuracy of the best model LCombined as a function of the number of possible analyses per sentence. Figure 4 shows the breakdown of accuracy for several sentence categories.

The figure displays average accuracy for sentences in the specified ambiguity ranges. As expected, we can see that the accuracy degrades with increased ambiguity. The accuracy is 94.6% for sentences with two possible analyses and 40% for sentences with more than 100 parses.

We can gain insight into the performance of our best log-linear model with the current number of features and training data size by looking at learning curves for the model. Figure 5 shows the accuracy of the model LCombined when using fractions of growing size from the training data.

The accuracy numbers shown are the average of 10-fold cross-validation as before. We can see that the log-linear model has enough features for the available training set sizes and achieves very high accuracy on the training set. The gap between training and test set accuracy is very large, especially for a small training set size. We can conclude that for the available training size, the model is overfitting the training data and it could do better if we

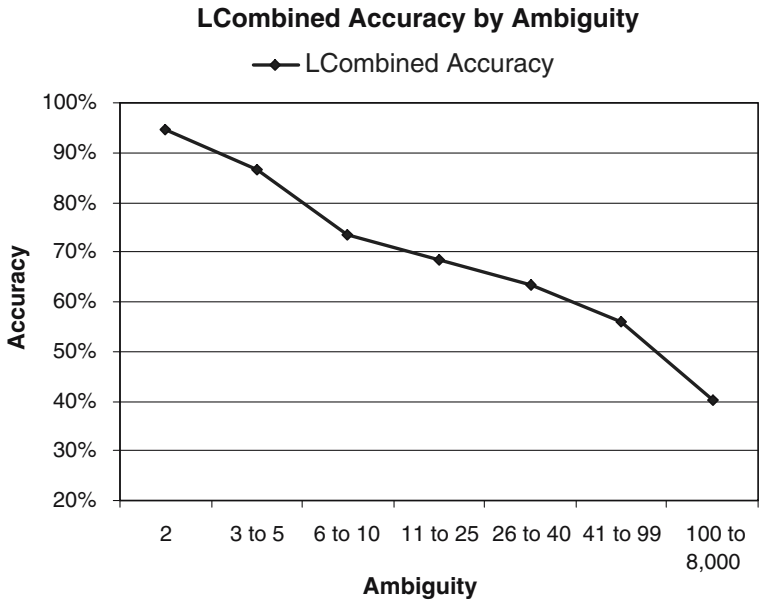


Figure 4. Parse ranking accuracy of LCombined by number of possible parses.

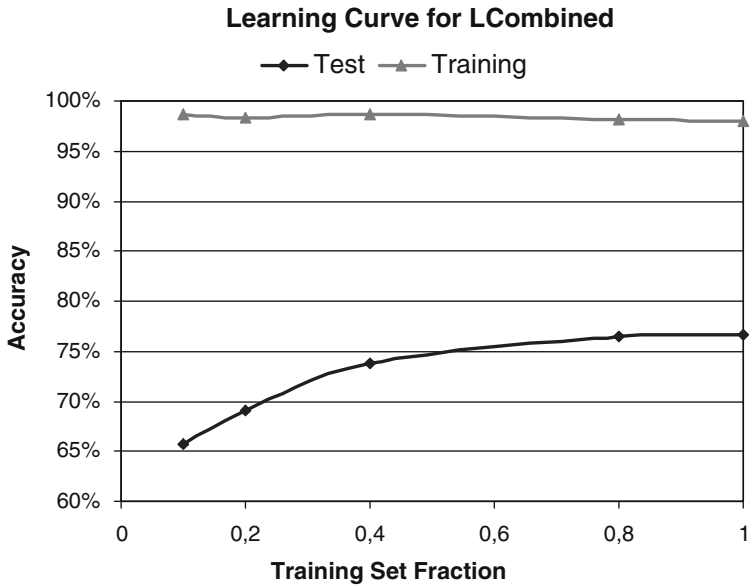


Figure 5. Learning curve for the model LCombined.

Table IV. Accuracy of generative models trained using derivation trees and phrase structure trees

Model	Derivation Trees	PS Trees	Combined
PCFG-1P	61.6	48.5	62.1
PCFG-2P	68.4	60.5	67.8
PCFG-3P	70.7	65.3	71.1

incorporated other non-sparse features, for example by using features from the HPSG signs.

It is interesting to compare models over derivation trees and phrase structure trees. Our experiments suggest that the information provided by the labels in the derivation trees is more helpful for disambiguation. Table IV shows accuracy results for equivalent models using the two different tree representations. We note that ancestor annotation is even more helpful for phrase structure trees, and that performance is lower for models trained on phrase structure trees. If we combine the log-probabilities assigned by models over derivation trees and phrase structure trees, similarly to the combination of models in Combined,[4] we obtain a model with slightly higher accuracy for PCFG-1P and PCFG-3P. For PCFG-2P, the combination is slightly weaker than the model over derivation trees only.

Based on our experiments, we can make the following observations:

- Overall it is surprising that the PCFG-1P/A and CPCFG-1P/A models over derivation trees work so well given the nature of node labels which are schema names and do not in general contain information about the phrasal types of the constituents.
- The current semantic models PCFG-Sem and LPCFG-Sem do not give us large performance gains. Perhaps this is due to data sparsity at the current size of the corpus, or the limitations of the semantic representation as semantic dependency trees rather than MRS structures.
- The conditional model LPCFG-Sem does not do much better than the joint PCFG-Sem model. This might be justified by the fact that although the LPCFG-Sem model will have a lower asymptotic error rate, it may not be approached due to the sparsity of the training data at the level of semantic relations (Ng amd Jordan, 2002).
- The overfitting effect of adding a large number of lexical features is stronger for the conditional model thus making it harder to improve generalization performance and making careful feature selection increasingly important.

6. Error Analysis

We performed a more extensive error analysis for the model LPCFG-3P on all errors in one part of the data.[5] Model LPCFG-3P is a conditional log-linear model over derivation trees, using as local features the node, its parent, its grandparent, and the expansion. Its overall accuracy was 74.9%. Since the annotation consistency of the 3rd Growth was improved, we hoped that the fraction of errors due to wrong annotation would diminish compared to the 1st Growth (Toutanova et al., 2002), and this was indeed the case.

For a total of 165 sentences, the model made an error in parse selection. The error analysis suggests the following breakdown:

- For about 26% of errors, the annotation in the treebank was wrong.
- For about 12% of the errors, both the treebank and the model were wrong.
- About 62% of the errors were real errors and we could hope to get them right.

The number of annotation errors is down to 26% from the previously reported 50% figure in (Toutanova et al., 2002). This shows the quality of the treebank is indeed improved. Correspondingly, the percent of real errors is up from 20 to 62%.

A more detailed break-down of the real errors (103 out of 165), in which the treebank was right and the model was wrong, follows:

- 27 are PP-attachment errors.
- 21 are errors in choosing the correct lexical item.
- 15 are other modifier attachment errors.
- 13 are coordination errors.
- 9 are errors in the complement/adjunct distinction.
- 18 are other errors.

The types of most common errors are similar to the ones observed in Penn Treebank parsing. Since the Redwoods treebank makes finer grained distinctions, there are additional error types. The two most frequently occurring types of errors are PP attachment and lexical item selection.

The PP attachment errors seem to be addressable by better use of semantic or lexical information as other researchers have proposed, e.g., (Hindle and Rooth, 1991; Collins and Brooks, 1995). Most of the time low attachment is correct as has been observed for other treebanks and the model does seem to prefer low attachment fairly strongly. But we do not at present have special features to model low or high attachment and in future models we plan to add this information.

An example of an error of this sort where the correct attachment is high is for the sentence *"I do not like to go anywhere on Sundays"*, where the model chose to attach the PP *on Sundays* to *anywhere* rather than to *go*.

For this case the low attachment to *anywhere* should be strongly dispreferred if there were sufficient lexical information.

Another interesting case of a PP attachment error is for the sentence *"I will put you in my schedule for March sixteenth at one o'clock"*. The correct attachment for the PP *at one o'clock* is low, as a modifier of *March sixteenth*, but the model chose to attach it high to *put* in the meaning that the putting in the schedule event would happen at one o'clock and not the meeting. Again here semantic collocation information would be useful as for example knowing that people do not usually talk about entering information in their schedules at a particular time.

The second largest type of errors are cases where the lexical item for a word was not chosen correctly. An example of this is for the sentence *"Yeah, that is about all"*. The model selected the meaning of *about* as a preposition, whereas the preferred analysis of *about* in this case should be as a degree specifier. In addition to being very common as a degree specifier in our corpus domain, *about* is also very common in the collocation *about all*. So again lexical information should be useful. Another similar case is the sentence *"But we are getting real close to the holidays"*. The model did not select the correct meaning of *real* here as an adverb but chose the meaning of *real* as an adjective which could be a possible meaning in this sentence in fairy-tales but quite improbable in the domain of appointment scheduling.

Another amusing lexical error was for the sentence *"You said you were getting in Tuesday night"*. The model selected the rare meaning of *in* as an abbreviation for *Indiana*.[6] This is not semantically plausible in this sentence and domain as people should not normally get states.

In summary we think that more lexical information will help resolve attachment and lexical ambiguities despite possible problems of data sparseness. We can expect that increasing the corpus size will be helpful to obtain better word-specific statistics for our current models. Automatic clustering or exploring existing lexical hierarchies could also improve our modeling of semantic preferences. Since our current experiments suggest that there are not very big gains from the semantic dependencies model, further research is necessary to resolve this conflict of intuition and results.

7. Conclusions and Future Work

This article has detailed our initial experiments on HPSG parse disambiguation using statistical models. We demonstrated the usefulness of building models over derivation trees of HPSG analyses and showed how they can be supplemented with semantic and lexical item sequence information for significant accuracy improvement. A particularly useful feature of our experi-

ments is that they show paired comparisons of generative and conditional models with exactly the same features. While there is considerable evidence from various domains on the value of discriminative models, and they have been successfully used for parsing, there is a lack of extant results showing comparisons between otherwise identical parsing models, and carefully showing the value of the conditional models, as we have done. Use of conditional models always delivered a very useful performance boost, but an interesting result was that the conditional model gave the greatest value on the simplest model, where the independence assumptions of the generative model were most greatly violated, and where most value could be gained by not just weighting features according to their relative frequency.

The work presented here should be viewed as only a first round of experiments: we have barely dug into the rich syntactic and semantic information available in the Redwoods treebank. In future work we intend to build more complex probabilistic models capable of using more information from within a sentence, and indeed from prior discourse information, for prediction. One particular avenue of interest is effectively using semantic representations for disambiguation: we have so far only gotten quite limited value from our semantic dependency trees, and hope to get more value from the full MRS semantic representations. This requires more complex modeling, because of the non-tree structure of these graphs, but can be suitably handled within the conditional log-linear model framework we have already been using. We also plan to explore finding and using other features within the full HPSG sign which are useful features for disambiguation. Such simple features as syntactic category, clause finiteness and agreement presumably have considerable value, and through the feature equality principles of HPSG(such as the Head Feature Principle and the Nonlocal Feature Principle), they could be accessed in the sign where they are relevant, rather than being imperfectly captured as long distance information within the derivation tree. Some experiments exploring these ideas appear in Toutanova et al. (2003a).

It is in many ways a quite surprising result that our system can determine the correct parse in over 76% of cases, without using any discourse context or world knowledge beyond statistics latent in the corpus. While in part this reflects the fact that the domain of scheduling dialogs within the Redwoods corpus is fairly straightforward, it nevertheless also shows the great leverage on the ambiguity resolution problem that can be derived from the combination of rich precise grammars together with probabilistic frameworks for evidence combination and adjudication. We eagerly anticipate the next generation of more complex models that integrate many other sources of relevant information, and which hence provide an even more satisfactory solution to the resolution problem.

Acknowledgements

We particularly wish to thank Stuart Shieber for his work during the early stages of this project, in particular for doing the initial implementation and evaluation of a trigram tagging model. Additionally, we would like to thank Emily Bender, Ivan Sag, Thorsten Brants, Timothy Baldwin, and the other participants in the Redwoods project for fruitful discussion, the audience at the Treebanks and Linguistic Theories 2002 workshop in Sozopol for valuable questions and comments, and Miles Osborne and Jason Baldridge for more recent discussions of the models and revised Redwoods corpus. This work was supported by a CSLI internal seed grant, the Edinburgh-Stanford Link programme ROSIE project R36763, funded by Scottish Enterprise, and an IBM Faculty Partnership Award to the second author.

Notes

[1] For an introduction to HPSG, see (Pollard and Sag, 1994; Sag and Wasow, 1999).
[2] This derivation tree is also the fundamental data stored in the Redwoods treebank, since the full sign can be reconstructed from it by reference to the grammar.
[3] For an introduction to PCFG grammars see, for example, the text by Manning and Schütze (1999).
[4] We used fixed interpolation weights $\lambda_1 = 0.7$ and $\lambda_2 = 0.3$ for derivation and phrase structure trees.
[5] The section corresponding to one of the treebanked Verbmobil CDs – CD32.
[6] Note that these sentences are a transcription of spoken dialogues so capitalization information is not reliably available in the data.

References

Abney, S. P. (1997) Stochastic Attribute-Value Grammars. *Computational Linguistics*, 23, pp. 597–618.

Agresti A. (1990) *Categorical Data Analysis*. Wiley, New York.

Charniak E. (1997) Statistical Parsing with a Context-Free Grammar and Word Statistics. In *Proceedings of the 14th National Conference on Artifical Intelligence*. Providence, RI, pp. 598–603.

Charniak E., Carroll G. (1994) Context-Sensitive Statistics for Improved Grammatical Language Models. In *Proceedings of the 12th National Conference on Artificial Intelligence*. Srattle, WA, pp. 742–747.

Chen S., Rosenfeld R. (1999) A Gaussian Prior for Smoothing Maximum Entropy Models. Technical Report CMUCS-99-108, Carnegie Mellon.

Collins M. (1999) Head-Driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania.

Collins M., Brooks J. (1995) Prepositional Attachment Through a Backed-off Model. In Yarovsky D. and Church K. (eds.), *Proceeding of the 3rd Workshop on Very Large Corpora*. Somerset, New Jersey, pp. 27–38, Association for Computational Linguistics.

Collins M. J.(1997) Three Generative, Lexicalised Models For Statistical Parsing. In *Proceedings of the 35th Meeting of the Association for Computational Linguistics and the 7th Conference of the European Chapter of the ACL.* Madrid, Spain, pp. 16–23.

Copestake A., Flickinger D. P., Sag I. A., Pollard, C. (1999) Minimal Recursion Semantics. An Indroduction. Ms., Stanford University.

Friedman N., Goldszmidt M.(1996) Learning Bayesian Network with Local Structure. In *Proceeding of the 12th Conference on Uncertainty in Artifical Intelligence.*

Harris T. E. (1963) *The Theory of Branching Processes.* Springer, Berlin, Germany.

Hindle D., Rooth M. (1991) Structural Ambiguity and Lexical Relations. In *Proceedings of the 29th Meeting of the Association for Computational Linguisitics.* pp. 229–236.

Johnson M. (1998) PCFG Models of Linguistic Tree Representations. *Computational Linguisitics*, 24, pp. 613–632.

Johnson M., Geman, S., Canon, S., Chi, Z., Riezler, S. (1999) Estimators for Stochastic 'Unification-based' Grammars. In *Proceeding of the 37th Meeting of the Association for Computational Linguistics.* College Park, MD, pp. 535–541.

Kaplan R. M., Bresnan J. (1982) Lexical-Functional Grammar: A Formal System for Grammatical Representation. In Bresnan J. (ed.), *The Mental Representation of Grammatical Relations.* Cambridge, MA: MIT Press, pp. 173–281.

Klein D., Manning C. D. (2002) Conditional Structure Versus Conditional Estimation in NLP Models. In *EMNLP* 2002.

MacDonald M. C. (1994) Probabilistic Constraints and Syntactic Ambiguity Resolution. *Language and Cognitive Processes*, 9, pp. 157–201.

Magerman D. M. (1995) Statistical Decision-Tree Models for Parsing. In *Proceeding of the 33rd Meeting of the Association for Computational Linguistics.*

Manning C.D., Schütze. (1999) *Foundations of Statistical Natural Language Processing.* Cambridge, MA: MIT Press.

Marciniak M., Mykowiecka A., Przepiórkowski A., Kupść A. (1999) Construction of an HPSG Treebank for Polish. In *Journée ATALA, 18–19 juin, Corpus annotés pour la syntaxe.* Paris, pp. 97–105.

Marcus M. P., Santorini B., Marcinkiewicz M. A. (1993) Bulding a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19, pp. 313–330.

Ng A., Jordan M. (2002) On Discriminative Vs. Generative Classifiers: A comparison of logistic regression and Naive Bayes. In *NIPS 14.*

Oepen S., Flickinger D., Toutanova K., Manning C. D. (2004) LinGO Redwoods. A Rich and Dynamic Treebank for HPSG. *Journal of Language and Computation.*

Oepen S., Toutanova K., Shieber S., Manning C., Flickinger D., Brants T. (2002) The LinGo Redwoods Treebank: Motivation and Preliminary applications. In *COLING 19*.

Pollard C., Sag I. A. (1994) *Head-Driven Phrase Structure Grammar.* University of Chicago Press, Chicago.

Riezler S., King T. H., Kaplan R. M., Crouch R., Maxwell J. T., III, Johnson M.(2002) Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics.*

Sag I. A., Wasow T.(1999) *Syntactic Theory: A Formal Introduction*. CSLI Publications, Stanford, CA.

Simov K., Osenova P., Slavcheva M., Kolkovka S., Balabanova E., Doikoff D., Ivanova K., Simov A., Kouylekov M. (2002) Building a Linguistically Interpreted Corpus of Bulgarian : The BulTreeBank. In *Proceedings of LREC.* Canary Islands, Spain, pp. 1729–1736.

Srinivas B., Joshi A. K. (1999) Supertagging: An Approach to Almost Parsing. *Computational Linguistics*, 25, pp. 237–265.

Toutanova K., Manning C., Oepen S., Flickinger D.(2003a) Parse Selection on the Redwoods Corpus : 3rd Growth Results. CS Technical Report, Stanford University.

Toutanova K., Manning C. D. (2002) Feature Selection for a Rich HPSG Grammar Using Decision Trees. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*.

Toutanova K., Manning C. D., Flickinger D., Oepen S. (2002) Parse Disambiguation for a Rich HPSG grammar. In *Treebanks and Linguistic Theories.* Sozopol, Bulgaria.

Toutanova K., Mitchell M., Manning C. (2003b) Optimizing Local Probability Models for Satistical Parsing. In *Proceeding of the 14th European Conference on Machine Learing (ECML).* Dubrovnik, Croatia.

Trueswell J. C. (1996) The role of lexical frequency in syntactic ambiguity resolution. *Journal of Memory and Language*, 35, pp. 566–585.

Vapnik V. N. (1998) *Statistical Learning Theory.* Wiley, New York.

Witten I. H., Bell T. C. (1991) The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Trans. Inform. Theory*, 37(4) pp. 1085–1094.