Constructing Linguistically Motivated Structures from Statistical Grammars

Ali Basirat
NLP Lab, School of ECE,
College of Engineering,
University of Tehran, Tehran, Iran
a.basirat@srbiau.ac.ir

Heshaam Faili
NLP Lab, School of ECE,
College of Engineering,
University of Tehran, Tehran, Iran
hfaili@ut.ac.ir

Abstract

This paper discusses two Hidden Markov Models (HMM) for linking linguistically motivated XTAG grammar and the automatically extracted LTAG used by MICA parser. The former grammar is a detailed LTAG enriched with feature structures. And the latter one is a huge size LTAG that due to its statistical nature is well suited to be used in statistical approaches. Lack of an efficient parser and sparseness in the supertags set are the main obstacles in using XTAG and MICA grammars respectively. The models were trained by the standard HMM training algorithm, Baum-Welch. To converge the training algorithm to a better local optimum, the initial state of the models also were estimated using two semi-supervised EM-based algorithms. The resulting accuracy of the model (about 91%) shows that the models can provide a satisfactory way for linking these grammars to share their capabilities together.

1 Introduction

Tree Adjoining-Grammar (TAG) is a tree generating system that forms the object language by the set of derived trees (Joshi and Schabez, 1991). This formalism as a Mildly Context Sensitive Grammar is supposed to be powerful enough to model the natural languages (Joshi, 1985).

In the lexicalized case (LTAG), each lexical item of the object language is associated with at least one elementary structure of the grammar called *elementary tree*. Each elementary tree in LTAGs can be considered as a complex description of its anchor that provides a domain of locality over which the anchor can specify syntactic and semantic constraints(Bangalore and Joshi, 1999). Extended domain of locality and factoring of recursion from the domain of dependency are the main key properties of using these grammars (Bangalore and Joshi, 1999).

There are two ways for creating the set of elementary trees (Faili and Basirat, 2010). The first method is the manual crafting of the elementary trees as it was done in the XTAG project (XTAG-Group, 2001). And the alternate one is the automatically extraction of them from some annotated treebanks as it was done in (Xia, 2001; Chen, 2001). The result of the former method is a detailed LTAG that is enriched with semantic representation but suffers from the lack of statistical information. The output of the latter one on the other hand, is a huge size LTAG that suffers from the sparseness problem in the elementary trees set but contains enough statistical information that make it suitable to be used in statistical approaches. The relatively huge size of the automatically extracted elementary trees set is an obstacle in annotating these structures with semantic representation (Chen, 2001).

One of the negative aspects of using LTAGs is their high computational complexity of parsing algorithm, ($O(n^6)$) (Kallmeyer, 2010). Regarding the work presented in (Sarkar, 2007), the factors that affect the parsing complexity of such lexicalized grammars are the number of trees selected by the words in the input sentence and the clausal complexity of the sentence to be parsed. The first factor, named *Syntactic Lexical Ambiguity*, directly addresses *Supertagging*, proposed by Bangalore and Joshi (1999).

Supertagging is a robust partial parsing approach that can be applied for increasing up the speed of LTAG parsing algorithm (Bangalore and Joshi, 1999). In supertagging the flexibility of linguistically motivated lexical descriptions are integrated with the robustness of statistical approaches. The idea is based on extending the no-

Proceedings of Recent Advances in Natural Language Processing, pages 63–69,
Hissar, Bulgaria, 12-14 September 2011.

tion of '*tag*' from the standard *Part Of Speech* to a tag that represents a rich and complex syntactic structure, called *Supertag*. In the lexicalized grammars like LTAGs each elementary structure of the grammar can be considered as a supertag. Supertagging itself is the task of assigning the supertags to each word of the processing sentence. *After* supertagging the only thing that the LTAG parser should do is to attach these selected supertags for creating a forest of derived/derivation trees.

Supertagging as a search problem can be modeled by two major methods, *generative model* and *classification approach* (Bangalore et al., 2005). In the former method the problem is modeled by a *Hidden Markov Model* and in the latter one it is modeled by the discriminant approaches like *SVM* and *Maximum Entropy Estimation*. Applying each of these methods in supertagging is subject to the availability of enough statistical information about the problem. Hence, due to their statistical nature, the automatically extracted LTAGs are more suitable to be used by supertagging algorithm than the manually crafted LTAGs. This characteristic of automatically extracted LTAGs caused the emergence of some powerful statistical parsers like MICA (Bangalore et al., 2009) that works based on the supertagging approach.

The lack of an efficient LTAG parser for manually crafted LTAGs beside the weakness of the automatically extracted LTAGs in representing semantic representation, encouraged us to rectify these deficiencies by making an interface between these grammars. The interface was established between individual elementary trees of each grammars such that any elementary tree of the source LTAG could be mapped onto an elementary tree of the target LTAG. The idea is similar to the *Hidden TAG Model* (Chiang and Rambow, 2006) that links many spoken dialects of a language to benefit from sharing rich resources. Here by relating two different perspectives of a natural language presented in the form of two LTAGs, we are going to share their capabilities together.

The interface was modeled as a sequence tagger that deals with the problem of how to map each supertag sequence of the source LTAG onto a supertag sequence of the target LTAG given the local and non-local information of the source sequence. An unsupervised sequence tagger based on *Hidden Markov Model* (HMM) was proposed that produces a target supertag sequence given a source supertag sequence. The sequence tagger was trained using the standard HMM training algorithm called *Baum-Welch*. Due to this fact that the algorithm convergence is tightly depending on the HMM initial state, the initial state of the HMM also was trained intellectually using an EM-Based semi-supervised bootstrapping algorithm. The solution was applied on the manually crafted English XTAG grammar (XTAG-Group, 2001) as target LTAG and the automatically extracted LTAG used by MICA parser (Bangalore et al., 2009) as source LTAG.

The significance of this work is as follow. First, as a solution for enhancing the parsing efficiency of the XTAG grammar, as it was done by Faili (2009). Second, as a fully automated method for bridging between grammars in order to share their capabilities together.

2 Related Work

Bridging between grammars in order to share their capabilities is considered by some researchers. Improving the parsing quality in the resource-poor languages (Chiang and Rambow, 2006), enriching automatically extracted LTAGs with semantic representation (Chen, 2001; Faili and Basirat, 2010; Faili and Basirat, 2011), increasing the syntactic coverage of lexicalized resources (Dang et al., 2000; Kipper et al., 2000), and finding the overlap between two grammars (Xia and Palmer, 2000) are considered as the most important reasons for performing this task.

In general, the proposed methods for performing such a task could be classified into two major categories. The first category consists of the methods that try to link the grammars using the structural similarities of the grammar's elements regardless of the syntactic environments that the elements may be placed. The approaches proposed in (Chen, 2001), (Xia and Palmer, 2000), and (Ryant and Kipper, 2004) are classified in this category.

The second one consists of the methods that try to make the connection regarding the statistical information of the syntactic environments where the grammar's elements appear on. Chiang and Rambow (2006) by introducing a novel concept, namely *hidden TAG model*, proposed a model analogous to a HMM for linking a resource-rich language to a resource-poor language. In

(Faili and Basirat, 2010; Faili and Basirat, 2011) also a statistical approach based on HMM for linking the automatically extracted LTAG from Penn Treebank (Chen, 2001) and English XTAG grammar (XTAG-Group, 2001) was proposed. Here by introducing two statistical models, we have closely followed the approach presented in (Faili and Basirat, 2011).

3 HMM-based LTAG mapping

The task of mapping a MICA elementary tree sequence onto an appropriate XTAG elementary tree sequence could be formulated as below:

> *Given a sequence of MICA elementary trees $T = (t_1, \ldots, t_n)$ assigned to sentence $S = (w_1, \ldots, w_n)$ by MICA, tag each element of T with an elementary tree $t'_i \in$ XTAG Grammar such that the likelihood of $T' = (t'_1, \ldots, t'_n)$ given T and S be maximized.*

This problem directly addresses a Hidden Markov Model (HMM) that relates a MICA elementary tree sequence as an observation sequence to the most probable XTAG elementary tree sequence as a hidden state path. Given such a model, the *Viterbi* algorithm can be used for finding the most probable hidden state path that generates the observation sequence. The rest of this part deals with the problem modeling using HMM.

3.1 Problem Modeling Using HMM

Regarding the existence gap between XTAG and MICA grammars (Chen, 2001), two possible mapping models were proposed. The M-1 model simply ignores this gap. It assumes every syntactic structure in the MICA grammar has at least one corresponding element in the XTAG grammar. In this case, each hidden state is exactly corresponded to a XTAG elementary tree. The MICA supertags also are considered as the observation symbols. Given any XTAG elementary tree t'_i and t'_j, the state transition matrix ($A = [a_{i,j}]$) contains the probability of seeing t'_j after t'_i in a sequence of XTAG elementary trees. For each MICA elementary tree t_j and XTAG elementary tree t'_i the observation probability matrix ($B = [b_{i,j}]$) also contains the probability $P(t_j|t'_i)$.

On the other hand, the alternate model, M-2, tries to model the relation between the grammars with respect to the existence gap between them. In this model it is assumed that there are some syntactic structures in the MICA grammar that are not supported by the XTAG grammar. The main difference between M-1 and M-2 is in their hidden states. In addition to the hidden states used in M-1, a new symbolic state, namely *UNKNOWN*, is added to the M-2 hidden states set. This new state is the representative of all syntactic structures that are modeled by MICA grammar but not by XTAG grammar.

3.2 Training

Both of the M-1 and M-2 models were trained by the Baum-Welch algorithm. As the other HMM training algorithm, Baum-Welch algorithm also cannot find the global optimum of the search space. This weakness is inherited from the HMM in which does not provide any clear solution to use any extra information of the problem. In this case, the initial state of the training algorithm provides a way to use a part of environment's knowledge that can largely cover the mentioned weakness (Rabiner, 1989).

To lead the training algorithm to a better solution two methods was peoposed for estimating the initial state of the models. Next part, introduces these algorithms.

3.3 Initialization

The initial state of the models has been trained using two novel semi-supervised EM-based training algorithms. The algorithms work based on the available set of MICA and XTAG elementary tree sequences achieved from parsing a set of English sentences namely *Initialization Data Base (IDB)*.

In the M-1 model, IDB must be selected so that all of its sentences can be modeled in both of XTAG and MICA grammars. This constraint is due to the M-1 assumption about the problem.

In M-2 the only constraint over the IDB sentences is that the sentences must be modeled in the MICA's grammar. In this case, IDB can be partitioned into two parts. The sentences that can be modeled by XTAG grammar, *Parsable Initialization dataset (PI)*, and the sentences that cannot be modeled by the XTAG grammar, *NotParsable Initialization dataset (NPI)*. The partitioning enables the model to consider the existence gap between the grammars.

3.3.1 Initializing M-1

Let C and C' be two sets of elementary tree sequences achieved from parsing IDB using MICA and XTAG parsers, respectively. Due to the statistical nature of MICA parser, for any sentence $S_i \in$ IDB, C contains a set of scored elementary tree sequences. Nevertheless, C' contains an ambiguity set of elementary tree sequences without any clear way to disambiguate it.

Given C and C', the simplest and most intuitive way for estimating the initial values of the HMM is MLE. Nevertheless, performing this application is subject to disambiguating the output of the XTAG parser stored in C'. This problem addresses a function that assigns a real value to each member of C' as shown in eq. 1.

$$\omega: C' \rightarrow \mathbb{R} \tag{1}$$

Given such a weighting function ω, the probability of transition ($S_i \rightarrow S_j$) in hidden states can be estimated by taking weighted count from all bigrams (S_i, S_j) in C' and normalizing by the sum of all bigrams (S_i, S_k) that share the same first elements. A similar method also can be used for computing the probabilities presented in the observation matrix (B) and Π.

Given C" = ω(C') and C, we define function Λ for generating the HMM λ using the aforementioned MLE (eq. 2).

$$\Lambda: C" \times C \rightarrow \lambda \tag{2}$$

The main problem here is to find an appropriate function ω. Function ω was estimated using a semi-supervised EM-based method. The algorithm takes the C and C' as input and attempts to estimate some values for function ω such that the objective function $\mathfrak{I}$ presented in eq. 3 is being maximized. Function $\mathfrak{I}$ shows the likelihood of observing C given the HMM λ achieved by Λ.

$$\mathfrak{I} = P(C|\lambda = \Lambda(C", C)) \tag{3}$$

In the EM formulation, the E-step was defined as the computing the value of λ using Λ. In M-step the algorithm attempts to update the ω regarding the earlier model resulted from E-step. Eq. 4 shows how to estimate the value of ω for a XTAG elementary tree sequences $T' \in C'$. In this equation, ξ shows the set of XTAG elementary tree sequences in C' that are generated from the sentence S, the generator of T'. $T_i \in C$ also represents the *ith* MICA elementary tree sequence in ξ. The index n shows the total number of sequences in C generated from S (| ξ |).

$$\omega(T') = \frac{\sum_{i=1}^{n} P(T_i)P(T_i, T'|\lambda)}{|\ \xi\ |} \tag{4}$$

3.3.2 Initializing M-2

In this part also for the sake of simplicity, C_{MP}, C_{XP} and C_{MNP} are used as the supertagging result of *PI* in MICA grammar, *PI* in XTAG grammar, and *NPI* in MICA grammar, respectively. Unlike the M-1 that uses all of the MICA elementary tree sequences resulted from parsing a sentence in IDB, here only the most probable MICA elementary tree sequence was used. So, related to each sentence in PI and NPI, we have a single MICA elementary tree sequence in C_{MP} and C_{MNP} respectively.

In this model, in addition to computing ω, applying MLE is subject to generating the set of elementary tree sequences for the sentences in dataset *NPI*. We name this set of elementary tree sequences C_{XNP}. Each sequence in C_{XNP} consists of XTAG elementary trees and have to contain at least one *UNKNOWN* symbol regarding this fact that NPI contains the sentences that couldn't be modeled in XTAG grammars. Given the paired sets (C_{MNP}, C_{XNP}) and (C_{MP}, C_{XP}) and an appropriate weighting function ω as shown in eq. 5, the initial values of HMM can be estimated using the mentioned MLE method.

$$\omega: C_{XNP} \cup C_{XP} \rightarrow \mathbb{R} \tag{5}$$

The ω was estimated using a *semi-supervised boot strapping EM-based* algorithm. Like the initialization algorithm proposed in sec. 3.3.1, this algorithm also has an iterative nature that tries to estimate some values for ω (hence for HMM parameters) in a greedy manner. The objective function in this phase is to maximize the likelihood of observing MICA supertag sequences in $C_{MNP} \cup C_{MP}$ (eq. 6). In the heart of the algorithm, the C_{XNP} is bootstrapped by applying a customized version of Viterbi algorithm on the C_{MNP} using the earlier value of HMM.

$$\mathfrak{I} = P(C_{MP} \cup C_{MNP}\ |\ \lambda) \tag{6}$$

The algorithm consists of four main stages as below:

1. *Pre Initializing*: Initializing the HMM parameters with out considering UNKNOWN hidden state.

2. *Bootstrapping*: Bootstrapping C_{XNP} by annotating C_{MNP} with hidden states labels.

3. *Updating*: Estimating the new value of HMM using Maximum Likelihood Estimation (MLE) on the paired sequences (C_{MNP}, C_{XNP}) and (C_{MP}, C_{XP})

4. *Termination*: Until the termination criterion is not satisfied go to step 2.

In the rest of this part we will express each phase in detail.

***Pre Initializing*:** In this step, it tries to estimate the HMM parameters from the related sequences in (C_{MP}, C_{XP}) using the MLE. Applying the MLE over these sets gives some approximations about the probabilities presented in the HMM parameters except the probabilities related to UNKNOWN hidden state. The weighting function used in this phase gives a uniform distribution of probability to each member of C_{MP} that are generated from same sentence.

The probabilities related to UNKNOWN hidden states also could be estimated using some heuristics over the existence gap between the grammars. For instance, the amount of uncertainty involved in the HMM parameters resulted by the MLE is a criterion for estimating the probabilities related to the UNKNOWN.

***Bootstrapping*:** In this phase it tries to annotate each MICA supertag sequence in C_{MNP} with a set of hidden state paths given the earlier value of HMM. To do this, a modified version of Viterbi algorithm, namely *Forced Viterbi*, was used. The algorithm looks for the hidden state paths that have the highest consistency with the earlier HMM and pass through UNKNOWN hidden state.

Before applying Forced Viterbi over C_{MNP}, we need some assumptions about the source elementary trees that are more likely to be corresponded to UNKNOWN. A simple solution for making such assumption is feasible via taking a differential between C_{MNP} and C_{MP}, and looking for the n-grams in the former that are not presented in the latter. The result of this process is a set of n-grams of MICA elementary trees, namely *Gap-set*, that their related n-gram in the original sentence couldn't be modeled in the XTAG grammar. For any n-gram in Gap-Set that is observed in a MICA elementary tree sequence member of C_{MNP}, by considering all conditions that the UNKNOWN can be assigned to the elementary trees of the observed n-gram, the Forced Viterbi algorithm will generate 2^n XTAG elementary tree sequences.

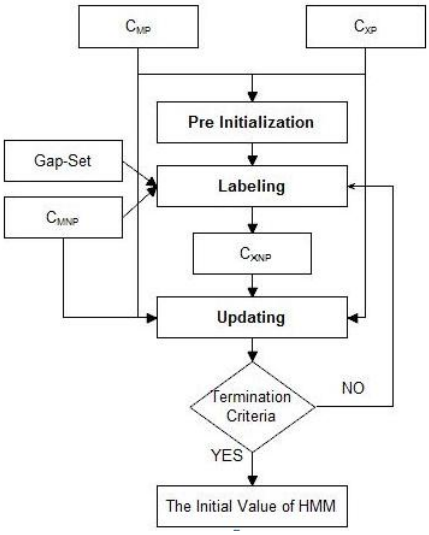

Figure 1: The HMM Initialization algorithm used in M-2

Updating: In this step, the HMM parameters will be updated regarding the paired sets (C_{MNP}, C_{XNP}) and (C_{MP}, C_{XP}). Having these paired sets and a scoring function ω, the HMM parameters can be updated using the mentioned MLE method.

For each XTAG elementary tree sequence $T' \in C_{XP} \cup C_{XNP}$ and its related MICA elementary tree sequence $T_i \in C_{MP} \cup C_{MNP}$, the scoring function ω can be defined as shown in eq. 7. ξ in this equation refers to the set of XTAG elementary trees that are generated from the same sentence and $T' \in \xi$.

$$\omega(T') = \frac{P(T', T \mid \lambda)}{\mid \xi \mid} \tag{7}$$

Fig. 1 gives an outline over the HMM initialization algorithm. Observing same values for the probability presented in eq. 6 or exceeding the predefined maximum number of iterations are two candidates to be used as termination criteria.

4 Numerical Results

4.1 Experiments Description

To evaluate the accuracy of the proposed models, the models have been initialized and trained with three real world data sets including ATIS , IBM Manual and Wall Street Journal (WSJ) corpora. Some parts of these datasets were randomly selected and divided into three distinct sections as initialization dataset (IDB), training dataset (TRDB) and testing dataset (TSDB). Table 4.1

No. Sentences				
	IDB_{M-1}	IDB_{M-2}	TRDB	TSDB
ATIS	904	991	1280	18
IBM	3463	4473	9742	102
WSJ	11913	16871	21709	197
No. words				
ATIS	7726	9734	16917	209
IBM	32840	46833	154668	1547
WSJ	102355	155879	221337	2029

Table 1: Statistical information about initialization, training and testing datasets used in M-1 and M-2

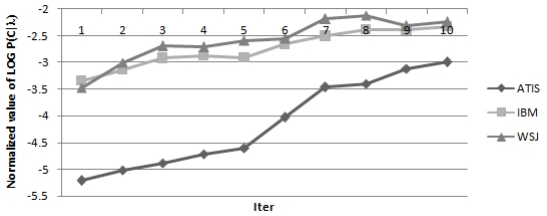

Figure 2: The values of the objective function presented in eq. 8 while initializing the M-1

shows some statistics about the datasets used in initialization, training and testing the models.

4.2 Initializing

The results of applying each of the initializing methods M-1 and M-2 over the IDBs are presented in figure 2 and 3 respectively. These figures show the value of Θ presented in eq. 8. 'O' in this equation refers to all MICA elementary tree sequences used in the algorithms. The observed progress in the likelihood of observing the MICA elementary tree sequences is an evidence on the successful of the algorithms.

$$\Theta = \frac{\sum_{T_i \in O} \log P(T_i \mid \lambda)}{\mid O \mid} \tag{8}$$

As these show, while the values resulted from M-2 are strictly ascending in a logarithmic manner, increasing in the values resulted from M-1 has no specific, predictable manner. It is due to the objective function shown in eq. 3 in which doesn't consider the score values of each MICA elementary tree sequences in C. In fact, related to any sentence in each IDB, C contains many scored MICA elementary tree sequences used in initializing algorithm but in the value of the objective function.

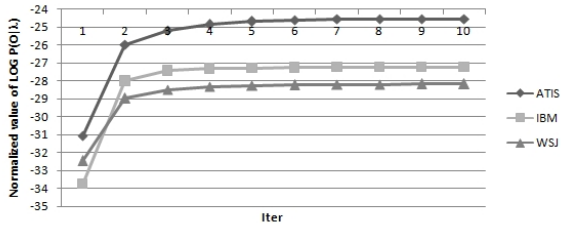

Figure 3: The values of the objective function presented in eq. 8 while initializing the M-2

	M-1	M-2	Base Line
ATIS	59.83%	80.00%	78.30%
IBM	79.55%	88.30%	88.70%
WSJ	87.75%	91.50%	88.96%

Table 2: The result of the tagging accuracy on the test sets

4.3 Models Evaluation

The models were evaluated in two ways, *tagging accuracy* and *parsed sequences*. The first criterion originally introduced in (Faili and Basirat, 2011), enables us to evaluate the models as XTAG supertaggers. The latter one also, provide a way to evaluate them when combining with a LTAG parser. In *parsed sequences* the main focus is on the number of resulted XTAG sequences that their constituents elementary trees can be attached to each other regarding the standard operations defined in TAG formalism, *Substitution* and *Adjunction*.

Due to the lack of a gold annotated corpus, the tagging accuracy has been done manually. Table 4.3 shows the result of the tagging accuracy over the mentioned test sets (TSDBs). The base line here is the result of tagging accuracy reported in (Faili and Basirat, 2011). As it can be seen, M-2 gives the best accuracy in comparison to the M-1 and the base line.

The result of the alternate criterion, parsed sentences, is given in table 4.3. As it shows, here also the M-2 gives a better response in compare to the M-1. An important point that should be noted is that, not all of the sentences in the test sets are covered by the XTAG grammar. In fact, our experiments showed that of all sentences in each of the ATIS-TSDB, IBM-TSDB, and WSJ-TSDB, all but 6%, 13% and 24% of them could be parsed by XTAG parser respectively.

	M-1	M-2
ATIS	5%	33%
IBM	12.74%	43.10%
WSJ	50.25%	57%

Table 3: Number of the parsed sentences

5 Conclusion

Two Hidden Markov Models (HMM) were proposed to make a bridge between the linguistic view of the English XTAG grammar and the statistical nature of the LTAG used by MICA parser (Bangalore et al., 2009). The models were trained by the standard HMM training algorithm, Baum-Welch. The initial state of the models also were estimated using two semi-supervised EM-based algorithms. The models can be used to combine the statistical approaches with the grammar engineering.

References

Srinivas Bangalore and Arvanid K. Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25(2):237–266.

S. Bangalore, P. Haffner, and G. Emami. 2005. Factoring global inference by enriching local representations. Technical report, AT&T Labs - Reserach.

Srinivas Bangalore, P. Boulllier, A. Nasr, O. Rambow, and B. Sagot. 2009. Mica: A probabilistic dependency parser based on tree insertion grammar. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

John Chen. 2001. *Toward Efficient Statistical Parsing Using Lexicalized Grammatical Information*. Ph.D. thesis, University of Delaware.

David Chiang and Owen Rambow. 2006. The hidden tag model: Synchronous grammars for parsing resource-poor languages. Proceedings of the 8th International Workshop on Tree Adjoining Grammar and Related Formalisms, July.

H. Dang, K. Kipper, and Martha Palmer. 2000. Integrating compositional semantic into a verb lexicon. In *In Proceedings of the Eighteenth International Conference on Computational Linguistic (COLING-2000)*.

Heshaam Faili and Ali Basirat. 2010. Augmenting the automated extracted tree adjoining grammars by semantic representation. In *6th IEEE International Conference on Natural Language Processing and Knowledge Engineering (IEEE NLP-KE'10)*, Beijing.

Heshaam Faili and Ali Basirat. 2011. An unsupervised approach for linking automatically extracted and manually crafted ltags. In *12th International conference on Intelligent Text Processing and Computational Linguistics (CICLing-2011)*, Tokyo, February.

Heshaam Faili. 2009. From partial toward full parsing. In *Recent Advances In Natural Language Processing (RANLP)*.

Arvanid K. Joshi and Yves Schabez. 1991. Tree adjoining grammars and lexicalized grammars. Technical Report MS-CIS 91–22, Department of Computer & Information Science, University of Pennsylvania.

Arvanid K. Joshi. 1985. How much context-sensitivity is necessary for characterizing structural descriptions? *Natural Language Processing: Theoretical, Computational, and Psychological Perspectives*, pages 206–250. New York, NY: Cambridge University Press.

Laura Kallmeyer. 2010. *Parsing Beyond Context-Free Grammars*, volume 0 of *Cognitive Technologies*. Springer.

K. Kipper, H. Dang, and Martha Palmer. 2000. Class-based construction of verb lexicon. In *In Proceedings of Seventh nation Conference on Artificial Intelligence (AAAI-2000)*.

Lawrence R. Rabiner. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Neville Ryant and Karin Kipper. 2004. Assigning xtag trees to verbnet. TAG+7: Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms, May 20-22.

Anoop Sarkar. 2007. Combining supertagging and lexicalized tree-adjoining grammar parsing. In Srinivas Bangalore and Aravind Joshi, editors, Complexity of Lexical Descriptions and its Relevance to Natural Language Processing: A Supertagging Approach.

Fia Xia and Martha Palmer. 2000. Evaluating the coverage of ltags on annotated corpora. the Workshop on Using Evaluation within HLT Programs: Results and Trends, May 30.

Fia Xia. 2001. *Automatic grammar generation from two different perspectives*. Ph.D. thesis, University of Pennsylvania.

XTAG-Group. 2001. A lexicalized tree adjoining grammar for english. Technical Report IRCS 01–03, Institute for Research in Cognitive Science, University of Pennsylvania.