Nonparametric Bayesian Inference and Efficient Parsing for Tree-adjoining Grammars

Elif Yamangil and **Stuart M. Shieber**
Harvard University
Cambridge, Massachusetts, USA
{elif, shieber}@seas.harvard.edu

Abstract

In the line of research extending statistical parsing to more expressive grammar formalisms, we demonstrate for the first time the use of tree-adjoining grammars (TAG). We present a Bayesian nonparametric model for estimating a probabilistic TAG from a parsed corpus, along with novel block sampling methods and approximation transformations for TAG that allow efficient parsing. Our work shows performance improvements on the Penn Treebank and finds more compact yet linguistically rich representations of the data, but more importantly provides techniques in grammar transformation and statistical inference that make practical the use of these more expressive systems, thereby enabling further experimentation along these lines.

1 Introduction

There is a deep tension in statistical modeling of grammatical structure between providing good expressivity — to allow accurate modeling of the data with sparse grammars — and low complexity — making induction of the grammars (say, from a treebank) and parsing of novel sentences computationally practical. Tree-substitution grammars (TSG), by expanding the domain of locality of context-free grammars (CFG), can achieve better expressivity, and the ability to model more contextual dependencies; the payoff would be better modeling of the data or smaller (sparser) models or both. For instance, constructions that go across levels, like the predicate-argument structure of a verb and its arguments can be modeled by TSGs (Goodman, 2003).

Recent work that incorporated Dirichlet process (DP) nonparametric models into TSGs has provided an efficient solution to the daunting model selection problem of segmenting training data trees into appropriate elementary fragments to form the grammar (Cohn et al., 2009; Post and Gildea, 2009). The elementary trees combined in a TSG are, intuitively, primitives of the language, yet certain linguistic phenomena (notably various forms of modification) "split them up", preventing their reuse, leading to less sparse grammars than might be ideal (Yamangil and Shieber, 2012; Chiang, 2000; Resnik, 1992).

TSGs are a special case of the more flexible grammar formalism of tree adjoining grammar (TAG) (Joshi et al., 1975). TAG augments TSG with an *adjunction operator* and a set of *auxiliary trees* in addition to the substitution operator and initial trees of TSG, allowing for "splicing in" of syntactic fragments within trees. This functionality allows for better modeling of linguistic phenomena such as the distinction between modifiers and arguments (Joshi et al., 1975; XTAG Research Group, 2001). Unfortunately, TAG's expressivity comes at the cost of greatly increased complexity. Parsing complexity for unconstrained TAG scales as $O(n^6)$, impractical as compared to CFG and TSG's $O(n^3)$. In addition, the model selection problem for TAG is significantly more complicated than for TSG since one must reason about many more combinatorial options with two types of derivation operators. This has led researchers to resort to manual (Doran et al., 1997) or heuristic techniques. For example, one can consider "outsourcing" the auxiliary trees (Shieber, 2007), use template rules and a very small number of grammar categories (Hwa, 1998), or rely on head-words and force lexicalization in order to constrain the problem (Xia et al., 2001; Chiang,

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 597–603,
Sofia, Bulgaria, August 4-9 2013.

2000; Carreras et al., 2008). However a solution has not been put forward by which a model that maximizes a principled probabilistic objective is sought after.

Recent work by Cohn and Blunsom (2010) argued that under highly expressive grammars such as TSGs where exponentially many derivations may be hypothesized of the data, local Gibbs sampling is insufficient for effective inference and global blocked sampling strategies will be necessary. For TAG, this problem is only more severe due to its mild context-sensitivity and even richer combinatorial nature. Therefore in previous work, Shindo et al. (2011) and Yamangil and Shieber (2012) used tree-insertion grammar (TIG) as a kind of expressive compromise between TSG and TAG, as a substrate on which to build nonparametric inference. However TIG has the constraint of disallowing wrapping adjunction (coordination between material that falls to the left and right of the point of adjunction, such as parentheticals and quotations) as well as left adjunction along the spine of a right auxiliary tree and vice versa.

In this work we formulate a blocked sampling strategy for TAG that is effective and efficient, and prove its superiority against the local Gibbs sampling approach. We show via nonparametric inference that TAG, which contains TSG as a subset, is a better model for treebank data than TSG and leads to improved parsing performance. TAG achieves this by using more compact grammars than TSG and by providing the ability to make finer-grained linguistic distinctions. We explain how our parameter refinement scheme for TAG allows for cubic-time CFG parsing, which is just as efficient as TSG parsing. Our presentation assumes familiarity with prior work on block sampling of TSG and TIG (Cohn and Blunsom, 2010; Shindo et al., 2011; Yamangil and Shieber, 2012).

2 Probabilistic Model

In the basic nonparametric TSG model, there is an independent DP for every grammar category (such as $c = \text{NP}$), each of which uses a base distribution P_0 that generates an initial tree by making stepwise decisions and concentration parameter α_c that controls the level of sparsity (size) of the generated grammars: $G_c \sim \text{DP}(\alpha_c, P_0(\cdot \mid c))$ We extend this model by adding specialized DPs for auxiliary trees $G_c^{\text{aux}} \sim \text{DP}(\alpha_c^{\text{aux}}, P_0^{\text{aux}}(\cdot \mid c))$ Therefore, we have an exchangeable process for generating auxiliary tree a_j given $j-1$ auxiliary trees previously generated

$$p(a_j \mid \mathbf{a}_{<j}) = \frac{n_{c,a_j} + \alpha_c^{\text{aux}} P_0^{\text{aux}}(a_j \mid c)}{j - 1 + \alpha_c^{\text{aux}}} \qquad (1)$$

as for initial trees in TSG (Cohn et al., 2009).

We must define base distributions for initial trees and auxiliary trees. P_0 generates an initial tree with root label c by sampling rules from a CFG $\tilde{P}$ and making a binary decision at every node generated whether to leave it as a frontier node or further expand (with probability β_c) (Cohn et al., 2009). Similarly, our P_0^{aux} generates an auxiliary tree with root label c by sampling a CFG rule from $\tilde{P}$, flipping an unbiased coin to decide the direction of the spine (if more than a unique child was generated), making a binary decision at the spine whether to leave it as a foot node or further expand (with probability γ_c), and recurring into P_0 or P_0^{aux} appropriately for the off-spine and spinal children respectively.

We glue these two processes together via a set of adjunction parameters μ_c. In any derivation for every node labeled c that is not a frontier node or the root or foot node of an auxiliary tree, we determine the number (perhaps zero) of *simultaneous adjunctions* (Schabes and Shieber, 1994) by sampling a Geometric(μ_c) variable; thus k simultaneous adjunctions would have probability $(\mu_c)^k(1-\mu_c)$. Since we already provide simultaneous adjunction we disallow adjunction at the root of auxiliary trees.

3 Inference

Given this model, our inference task is to explore posterior derivations underlying the data. Since TAG derivations are highly structured objects, we design a blocked Metropolis-Hastings sampler that samples derivations per entire parse trees all at once in a joint fashion (Cohn and Blunsom, 2010; Shindo et al., 2011; Yamangil and Shieber, 2012). As in previous work, we use a Goodman-transformed TAG as our proposal distribution (Goodman, 2003) that incorporates additional CFG rules to account for the possibility of backing off to the infinite base distribution P_0^{aux}, and use the parsing algorithm described by Shieber et al. (1995) for computing inside probabilities under this TAG model.

The algorithm is illustrated in Table 1 along with Figure 1. Inside probabilities are computed in a bottom-up fashion and a TAG derivation is sampled top-down (Johnson et al., 2007). The

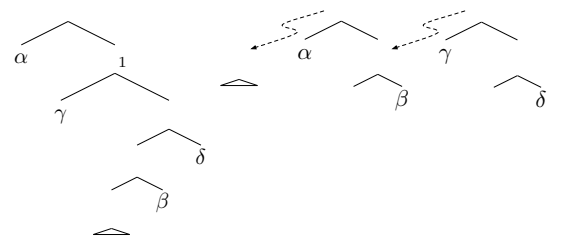

Figure 1: Example used for illustrating blocked sampling with TAG. On the left hand side we have a partial training tree where we highlight the particular nodes (with node labels $0, 1, 2, 3, 4$) that the sampling algorithm traverses in post-order. On the right hand side is the TAG grammar fragment that is used to parse these particular nodes: one initial tree and two *wrapping* auxiliary trees where one adjoins into the *spine* of the other for full generality of our illustration. Grammar nodes are labeled with their Goodman indices (letters i, j, k, l, m). Greek letters $\alpha, \beta, \gamma, \delta$ denote entire subtrees. We assume that a subtree in an auxiliary tree (e.g., α) parses the same subtree in a training tree.

sampler visits every node of the tree in post-order ($O(n)$ operations, n being the number of nodes), visits every node below it as a potential foot (another $O(n)$ operations), visits every mid-node in the path between the original node and the potential foot (if spine-adjunction is allowed) ($O(\log n)$ operations), and forms the appropriate chart items. The complexity is $O(n^2 \log n)$ if spine-adjunction is allowed, $O(n^2)$ otherwise.

4 Parameter Refinement

During inference, adjunction probabilities are treated simplistically to facilitate convergence. Only two parameters guide adjunction: μ_c, the probability of adjunction; and $p(a_j \mid \mathbf{a}_{<j}, c)$ (see Equation 1), the probability of the particular auxiliary tree being adjoined given that there is an adjunction. In all of this treatment, c, the context of an adjunction, is the grammar category label such as S or NP, instead of a unique identifier for the node at which the adjunction occurs as was originally the case in probabilistic TAG literature. However it is possible to experiment with further refinement schemes at parsing time. Once the sampler converges on a grammar, we can re-estimate its adjunction probabilities. Using the $O(n^6)$ parsing algorithm (Shieber et al., 1995) we experimented with various refinements schemes — ranging from full node identifiers, to Goodman index identifiers of the subtree below the adjunction (Hwa, 1998), to simple grammar category labels — and find that using Goodman index identifiers as c is the best performing option.

Chart item	Why made?	Inside probability
N_i[4]	By assumption.	–
N_k[3-4]	N_*[4] and β	$(1-\mu_c) \times \pi(\beta)$
N_m[2-3]	N_*[3] and δ	$(1-\mu_c) \times \pi(\delta)$
N_l[1-3]	γ and N_m[2-3]	$(1-\mu_c) \times \pi(\gamma)$ $\times\pi(\mathrm{N}_m[2\text{-}3])$
N_{aux}[1-3]	N_l[1-3]	$n_{c,a_l}/(n_c + \alpha_c^{aux})$ $\times\pi(\mathrm{N}_l[1\text{-}3])$
N_k[1-4]	N_{aux}[1-3] and N_k[3-4]	$\mu_c \times \pi(\mathrm{N}_{\text{aux}}[1\text{-}3])$ $\times\pi(\mathrm{N}_k[3\text{-}4])$
N_j[0-4]	α and N_k[1-4]	$(1-\mu_c) \times \pi(\alpha)$ $\times\pi(\mathrm{N}_k[1\text{-}4])$
N_{aux}[0-4]	N_j[0-4]	$n_{c,a_j}/(n_c + \alpha_c^{aux})$ $\times\pi(\mathrm{N}_j[0\text{-}4])$
N_i[0]	N_{aux}[0-4] and N_i[4]	$\mu_c \times \pi(\mathrm{N}_{\text{aux}}[0\text{-}4])$ $\times\pi(\mathrm{N}_i[4])$

Table 1: Computation of inside probabilities for TAG sampling. We create two types of chart items: (1) **per-node**, e.g., $\mathrm{N}_i[\nu]$ denoting the probability of starting at an initial subtree that has Goodman index i and generating the subtree rooted at node ν, and (2) **per-path**, e.g., $\mathrm{N}_j[\nu\text{-}\eta]$ denoting the probability of starting at an auxiliary subtree that has Goodman index j and generating the subtree rooted at ν minus the subtree rooted at η. Above, c denotes the context of adjunction, which is the nonterminal label of the node of adjunction (here, N), μ_c is the probability of adjunction, $n_{c,a}$ is the count of the auxiliary tree a, and $n_c = \sum_a n_{c,a}$ is total number of adjunctions at context c. The function $\pi(\cdot)$ retrieves the inside probability corresponding to an item.

Interestingly, this particular refinement scheme also allows for fast cubic-time parsing, which we achieve by approximating the TAG by a TSG with little loss of coverage (*no* loss of coverage under special conditions which we find that are often satisfied) and negligible increase in grammar size, as discussed in the next section.

5 Cubic-time parsing

MCMC training results in a list of sufficient statistics of the final derivation that the TAG sampler converges upon after a number of iterations. Basically, these are the list of initial and auxiliary trees, their cumulative counts over the training data, and their adjunction statistics. An adjunction statistic is listed as follows. If α is any elementary tree, and β is an auxiliary tree that adjoins n times at node ν of α that is uniquely reachable at path p, we write $\alpha \overset{p}{\leftarrow} \beta$ (n times). We denote ν alternatively as $\alpha[p]$.

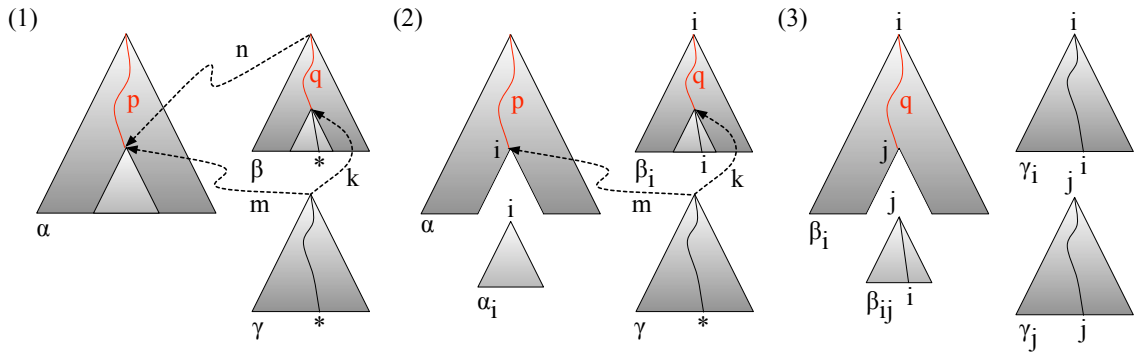

Figure 2: TAG to TSG transformation algorithm. By removing adjunctions in the correct order we end up with a larger yet adjunction-free TSG.

Now imagine that we end up with a small grammar that consists of one initial tree α and two auxiliary trees β and γ, and the following adjunctions occurring between them

$$\alpha \xleftarrow{p} \beta \text{ } (n \text{ times})$$
$$\alpha \xleftarrow{p} \gamma \text{ } (m \text{ times})$$
$$\beta \xleftarrow{q} \gamma \text{ } (k \text{ times})$$

as shown in Figure 2. Assume that α itself occurs $l > n + m$ times in total so that there is nonzero probability of no adjunction anywhere within α. Also assume that the node uniquely identified by $\alpha[p]$ has Goodman index i, which we denote as $i = G(\alpha[p])$.

The general idea of this TAG-TSG approximation is that, for any auxiliary tree that adjoins at a node ν with Goodman index i, we create an initial tree out of it where the root and foot nodes of the auxiliary tree are both replaced by i. Further, we split the subtree rooted at ν from its parent and rename the substitution site that is newly created at ν as i as well. (See Figure 2.) We can separate the foot subtree from the rest of the initial tree since it is completely remembered by any adjoined auxiliary trees due to the nature of our refinement scheme. However this method fails for adjunctions that occur at *spinal* nodes of auxiliary trees that have foot nodes below them since we would not know in which order to do the initial tree creation. However when the *spine-adjunction relation* is amenable to a *topological sort* (as is the case in Figure 2), we can apply the method by going in this order and doing some extra bookkeeping: updating the list of Goodman indices and redirecting adjunctions as we go along. When there is no such topological sort, we can approximate the TAG by heuristically dropping low-frequency adjunctions that introduce cycles.[1]

The algorithm is illustrated in Figure 2. In (1) we see the original TAG grammar and its adjunctions (n, m, k are adjunction counts). Note that the adjunction relation has a topological sort of α, β, γ. We process auxiliary trees in this order and iteratively remove their adjunctions by creating specialized initial tree duplicates. In (2) we first visit β, which has adjunctions into α at the node denoted $\alpha[p]$ where p is the unique path from the root to this node. We retrieve the Goodman index of this node $i = G(\alpha[p])$, split the subtree rooted at this node as a new initial tree α_i, relabel its root as i, and rename the newly-created substitution site at $\alpha[p]$ as i. Since β has only this adjunction, we replace it with initial tree version β_i where root/foot labels of β are replaced with i, and update all adjunctions into β as being into β_i. In (3) we visit γ which now has adjunctions into α and β_i. For the $\alpha[p]$ adjunction we create γ_i the same way we created β_i but this time we cannot remove γ as it still has an adjunction into β_i. We retrieve the Goodman index of the node of adjunction $j = G(\beta_i[q])$, split the subtree rooted at this node as new initial tree β_{ij}, relabel its root as j, and rename the newly-created substitution site at $\beta_i[q]$ as j. Since γ now has only this adjunction left, we remove it by also creating initial tree version γ_j where root/foot labels of γ are replaced with j. At this point we have an adjunction-free TSG with elementary trees (and counts) $\alpha(l), \alpha_i(l), \beta_i(n), \beta_{ij}(n), \gamma_i(m), \gamma_j(k)$ where l is the count of initial tree α. These counts, when they are normalized, lead to the appropriate adjunc-

[1] We found that, on average, about half of our grammars have a topological sort of their spine-adjunctions. (On average fewer than 100 spine adjunctions even exist.) When no such sort exists, only a few low-frequency adjunctions have to be removed to eliminate cycles.

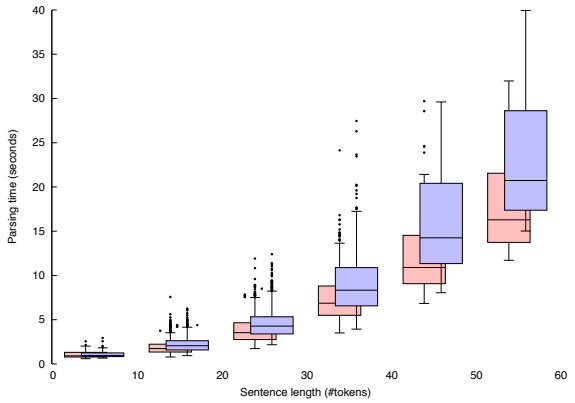

Figure 3: Nonparametric TAG (blue) parsing is efficient and incurs only a small increase in parsing time compared to nonparametric TSG (red).

tion probability refinement scheme of $\mu_c \times p(a_j \mid \mathbf{a}_{<j}, c)$ where c is the Goodman index.

Although this algorithm increases grammar size, the sparsity of the nonparametric solution ensures that the increase is almost negligible: on average the final Goodman-transformed CFG has 173.9K rules for TSG, 189.2K for TAG. Figure 3 demonstrates the comparable Viterbi parsing times for TSG and TAG.

6 Evaluation

We use the standard Penn treebank methodology of training on sections 2–21 and testing on section 23. All our data is head-binarized, all hyperparameters are resampled under appropriate vague gamma and beta priors. Samplers are run 1000 iterations each; all reported numbers are averages over 5 runs. For simplicity, parsing results are based on the maximum probability derivation (Viterbi algorithm).

In Table 4, we compare TAG inference schemes and TSG. $\text{TAG}_{\text{Gibbs}}$ operates by locally adding/removing potential adjunctions, similar to Cohn et al. (2009). TAG' is the $O(n^2)$ algorithm that disallows spine adjunction. We see that TAG' has the best parsing performance, while TAG provides the most compact representation.

model	*F measure*	*# initial trees*	*# auxiliary trees*
TSG	84.15	69.5K	-
$\text{TAG}_{\text{Gibbs}}$	82.47	69.9K	1.7K
TAG'	**84.87**	66.4K	1.5K
TAG	84.82	**66.4K**	**1.4K**

Figure 4: EVALB results. Note that the Gibbs sampler for TAG has poor performance and provides no grammar compaction due to its lack of convergence.

label	*#adj (spine adj)*	*ave. depth*	*#lex. trees*	*#left trees*	*#right trees*	*#wrap trees*
$\overline{\text{VP}}$	4532 (23)	1.06	45	22	65	0
$\overline{\text{NP}}$	2891 (46)	1.71	68	94	13	1
$\overline{\text{NN}}$	2160 (3)	1.08	85	16	110	0
$\overline{\text{NNP}}$	1478 (2)	1.12	90	19	90	0
$\overline{\text{NNS}}$	1217 (1)	1.10	43	9	60	0
$\overline{\text{VBN}}$	1121 (1)	1.05	6	18	0	0
$\overline{\text{VBD}}$	976 (0)	1.0	16	25	0	0
NP	937 (0)	3.0	1	5	0	0
$\overline{\text{VB}}$	870 (0)	1.02	14	31	4	0
$\overline{\text{S}}$	823 (11)	1.48	42	36	35	3
total	23320 (118)	1.25	824	743	683	9

Table 2: Grammar analysis for an estimated TAG, categorized by label. Only the most common top 10 are shown, binarization variables are denoted with overline. A total number of 98 wrapping adjunctions (9 unique wrapping trees) and 118 spine adjunctions occur.

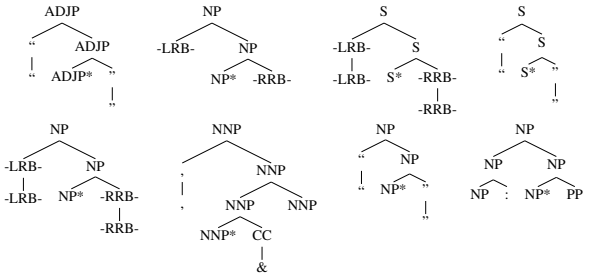

Figure 5: Example wrapping trees from estimated TAGs.

7 Conclusion

We described a nonparametric Bayesian inference scheme for estimating TAG grammars and showed the power of TAG formalism over TSG for returning rich, generalizable, yet compact representations of data. The nonparametric inference scheme presents a principled way of addressing the difficult model selection problem with TAG. Our sampler has near quadratic-time efficiency, and our parsing approach remains context-free allowing for fast cubic-time parsing, so that our overall parsing framework is highly scalable.[2]

There are a number of extensions of this work: Experimenting with automatically induced adjunction refinements as well as incorporating substitution refinements can benefit Bayesian TAG (Shindo et al., 2012; Petrov et al., 2006). We are also planning to investigate TAG for more context-sensitive languages, and synchronous TAG for machine translation.

[2] An extensive report of our algorithms and experiments will be provided in the PhD thesis of the first author (Yamangil, 2013). Our code will be made publicly available at `code.seas.harvard.edu/~elif`.

References

Xavier Carreras, Michael Collins, and Terry Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 456–463, Morristown, NJ, USA. Association for Computational Linguistics.

Trevor Cohn and Phil Blunsom. 2010. Blocked inference in Bayesian tree substitution grammars. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 225–230, Stroudsburg, PA, USA. Association for Computational Linguistics.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556, Morristown, NJ, USA. Association for Computational Linguistics.

Christine Doran, Beth Hockey, Philip Hopely, Joseph Rosenzweig, Anoop Sarkar, B. Srinivas, Fei Xia, Alexis Nasr, and Owen Rambow. 1997. Maintaining the forest and burning out the underbrush in xtag. In *Proceedings of the ENVGRAM Workshop*.

Joshua Goodman. 2003. Efficient parsing of DOP with PCFG-reductions. In Rens Bod, Remko Scha, and Khalil Sima'an, editors, *Data-Oriented Parsing*. CSLI Publications, Stanford, CA.

Rebecca Hwa. 1998. An empirical evaluation of probabilistic lexicalized tree insertion grammars. In *Proceedings of the 17th international conference on Computational linguistics - Volume 1*, pages 557–563, Morristown, NJ, USA. Association for Computational Linguistics.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 139–146, Rochester, New York, April. Association for Computational Linguistics.

Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July. Association for Computational Linguistics.

Matt Post and Daniel Gildea. 2009. Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48, Suntec, Singapore, August. Association for Computational Linguistics.

Philip Resnik. 1992. Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. In *Proceedings of the 14th conference on Computational linguistics - Volume 2*, COLING '92, pages 418–424, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yves Schabes and Stuart M. Shieber. 1994. An alternative conception of tree-adjoining derivation. *Computational Linguistics*, 20(1):91–124. Also available as cmp-lg/9404001.

Stuart M. Shieber, Yves Schabes, and Fernando C. N. Pereira. 1995. Principles and implementation of deductive parsing. *J. Log. Program.*, 24(1&2):3–36.

Stuart M. Shieber. 2007. Probabilistic synchronous tree-adjoining grammars for machine translation: The argument from bilingual dictionaries. In Dekai Wu and David Chiang, editors, *Proceedings of the Workshop on Syntax and Structure in Statistical Translation*, Rochester, New York, 26 April.

Hiroyuki Shindo, Akinori Fujino, and Masaaki Nagata. 2011. Insertion operator for Bayesian tree substitution grammars. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 206–211, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hiroyuki Shindo, Yusuke Miyao, Akinori Fujino, and Masaaki Nagata. 2012. Bayesian symbol-refined tree substitution grammars for syntactic parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–448, Jeju Island, Korea, July. Association for Computational Linguistics.

Fei Xia, Chung-hye Han, Martha Palmer, and Aravind Joshi. 2001. Automatically extracting and comparing lexicalized grammars for different languages. In *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 2*, IJCAI'01, pages 1321–1326, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

XTAG Research Group. 2001. A lexicalized tree adjoining grammar for English. Technical Report IRCS-01-03, IRCS, University of Pennsylvania.

Elif Yamangil and Stuart Shieber. 2012. Estimating compact yet rich tree insertion grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 110–114, Jeju Island, Korea, July. Association for Computational Linguistics.

Elif Yamangil. 2013. *Rich Linguistic Structure from Large-Scale Web Data*. Ph.D. thesis, Harvard University. Forthcoming.