A Context Free TAG Variant

Ben Swanson
Brown University
Providence, RI
chonger@cs.brown.edu

Elif Yamangil
Harvard University
Cambridge, MA
elif@eecs.harvard.edu

Eugene Charniak
Brown University
Providence, RI
ec@cs.brown.edu

Stuart Shieber
Harvard University
Cambridge, MA
shieber@eecs.harvard.edu

Abstract

We propose a new variant of Tree-Adjoining Grammar that allows adjunction of full wrapping trees but still bears only context-free expressivity. We provide a transformation to context-free form, and a further reduction in probabilistic model size through factorization and pooling of parameters. This collapsed context-free form is used to implement efficient grammar estimation and parsing algorithms. We perform parsing experiments the Penn Treebank and draw comparisons to Tree-Substitution Grammars and between different variations in probabilistic model design. Examination of the most probable derivations reveals examples of the linguistically relevant structure that our variant makes possible.

1 Introduction

While it is widely accepted that natural language is not context-free, practical limitations of existing algorithms motivate Context-Free Grammars (CFGs) as a good balance between modeling power and asymptotic performance (Charniak, 1996). In constituent-based parsing work, the prevailing technique to combat this divide between efficient models and real world data has been to selectively strengthen the dependencies in a CFG by increasing the grammar size through methods such as symbol refinement (Petrov et al., 2006).

Another approach is to employ a more powerful grammatical formalism and devise constraints and transformations that allow use of essential efficient algorithms such as the Inside-Outside algorithm (Lari and Young, 1990) and CYK parsing. Tree-Adjoining Grammar (TAG) is a natural starting point for such methods as it is the canonical member of the mildly context-sensitive family, falling just above CFGs in the hierarchy of formal grammars. TAG has a crucial advantage over CFGs in its ability to represent long distance interactions in the face of the interposing variations that commonly manifest in natural language (Joshi and Schabes, 1997). Consider, for example, the sentences

These pretzels are making me thirsty.
These pretzels are not making me thirsty.
These pretzels that I ate are making me thirsty.

Using a context-free language model with proper phrase bracketing, the connection between the words *pretzels* and *thirsty* must be recorded with three separate patterns, which can lead to poor generalizability and unreliable sparse frequency estimates in probabilistic models. While these problems can be overcome to some extent with large amounts of data, redundant representation of patterns is particularly undesirable for systems that seek to extract coherent and concise information from text.

TAG allows a linguistically motivated treatment of the example sentences above by generating the last two sentences through modification of the first, applying operations corresponding to negation and the use of a subordinate clause. Unfortunately, the added expressive power of TAG comes with $O(n^6)$ time complexity for essential algorithms on sentences of length n, as opposed to $O(n^3)$ for the CFG (Schabes, 1990). This makes TAG infeasible to analyze real world data in a reasonable time frame.

In this paper, we define OSTAG, a new way to constrain TAG in a conceptually simple way so

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 302–310,
Sofia, Bulgaria, August 4-9 2013.

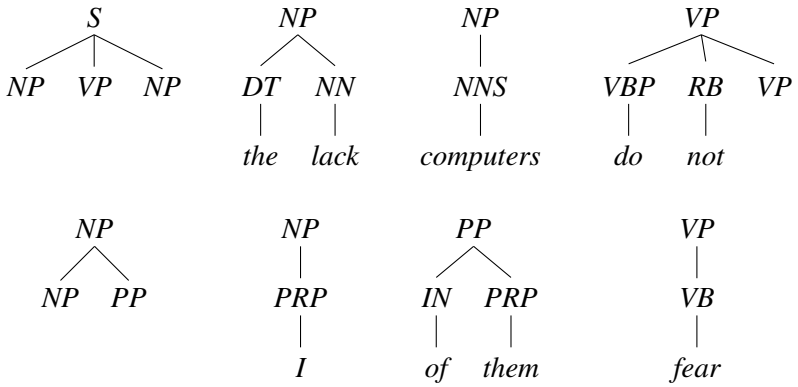


Figure 1: A simple Tree-Substitution Grammar using S as its start symbol. This grammar derives the sentences from a quote of Isaac Asimov's - "I do not fear computers. I fear the lack of them."

that it can be reduced to a CFG, allowing the use of traditional cubic-time algorithms. The reduction is reversible, so that the original TAG derivation can be recovered exactly from the CFG parse. We provide this reduction in detail below and highlight the compression afforded by this TAG variant on synthetic formal languages.

We evaluate OSTAG on the familiar task of parsing the Penn Treebank. Using an automatically induced Tree-Substitution Grammar (TSG), we heuristically extract an OSTAG and estimate its parameters from data using models with various reduced probabilistic models of adjunction. We contrast these models and investigate the use of adjunction in the most probable derivations of the test corpus, demonstating the superior modeling performance of OSTAG over TSG.

2 TAG and Variants

Here we provide a short history of the relevant work in related grammar formalisms, leading up to a definition of OSTAG. We start with context-free grammars, the components of which are $\langle N, T, R, S\rangle$, where N and T are the sets of nonterminal and terminal symbols respectively, and S is a distinguished nonterminal, the start symbol. The rules R can be thought of as elementary trees of depth 1, which are combined by substituting a derived tree rooted at a nonterminal X at some leaf node in an elementary tree with a frontier node labeled with that same nonterminal. The derived trees rooted at the start symbol S are taken to be the trees generated by the grammar.

2.1 Tree-Substitution Grammar

By generalizing CFG to allow elementary trees in R to be of depth greater than or equal to 1, we get the Tree-Substitution Grammar. TSG remains in the family of context-free grammars, as can be easily seen by the removal of the internal nodes in all elementary trees; what is left is a CFG that generates the same language. As a reversible alternative that preserves the internal structure, annotation of each internal node with a unique index creates a large number of deterministic CFG rules that record the structure of the original elementary trees. A more compact CFG representation can be obtained by marking each node in each elementary tree with a signature of its subtree. This transform, presented by Goodman (2003), can rein in the grammar constant G, as the crucial CFG algorithms for a sentence of length n have complexity $O(Gn^3)$.

A simple probabilistic model for a TSG is a set of multinomials, one for each nonterminal in N corresponding to its possible substitutions in R. A more flexible model allows a potentially infinite number of substitution rules using a Dirichlet Process (Cohn et al., 2009; Cohn and Blunsom, 2010). This model has proven effective for grammar induction via Markov Chain Monte Carlo (MCMC), in which TSG derivations of the training set are repeatedly sampled to find frequently occurring elementary trees. A straightforward technique for induction of a finite TSG is to perform this non-parametric induction and select the set of rules that appear in at least one sampled derivation at one or several of the final iterations.

2.2 Tree-Adjoining Grammar

Tree-adjoining grammar (TAG) (Joshi, 1985; Joshi, 1987; Joshi and Schabes, 1997) is an extension of TSG defined by a tuple $\langle N, T, R, A, S\rangle$, and differs from TSG only in the addition of a

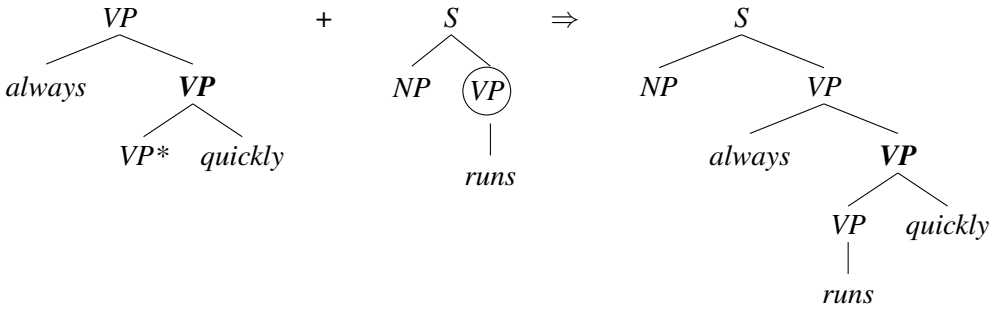


Figure 2: The adjunction operation combines the auxiliary tree (left) with the elementary tree (middle) to form a new derivation (right). The adjunction site is circled, and the foot node of the auxiliary tree is denoted with an asterisk. The OSTAG constraint would disallow further adjunction at the bold ***VP*** node only, as it is along the spine of the auxiliary tree.

set of auxiliary trees A and the adjunction operation that governs their use. An auxiliary tree α is an elementary tree containing a single distinguished nonterminal leaf, the foot node, with the same symbol as the root of α. An auxiliary tree with root and foot node X can be adjoined into an internal node of an elementary tree labeled with X by splicing the auxiliary tree in at that internal node, as pictured in Figure 2. We refer to the path between the root and foot nodes in an auxiliary tree as the *spine* of the tree.

As mentioned above, the added power afforded by adjunction comes at a serious price in time complexity. As such, probabilistic modeling for TAG in its original form is uncommon. However, a large effort in non-probabilistic grammar induction has been performed through manual annotation with the XTAG project(Doran et al., 1994).

2.3 Tree Insertion Grammar

Tree Insertion Grammars (TIGs) are a longstanding compromise between the intuitive expressivity of TAG and the algorithmic simplicity of CFGs. Schabes and Waters (1995) showed that by restricting the form of the auxiliary trees in A and the set of auxiliary trees that may adjoin at particular nodes, a TAG generates only context-free languages. The TIG restriction on auxiliary trees states that the foot node must occur as either the leftmost or rightmost leaf node. This introduces an important distinction between left, right, and wrapping auxiliary trees, of which only the first two are allowed in TIG. Furthermore, TIG disallows adjunction of left auxiliary trees on the spines of right auxiliary trees, and vice versa. This is to prevent the construction of wrapping auxiliary trees, whose removal is essential for the simplified complexity of TIG.

Several probabilistic models have been proposed for TIG. While earlier approaches such as Hwa (1998) and Chiang (2000) relied on hueristic induction methods, they were nevertheless sucessful at parsing. Later approaches (Shindo et al., 2011; Yamangil and Shieber, 2012) were able to extend the non-parametric modeling of TSGs to TIG, providing methods for both modeling and grammar induction.

2.4 OSTAG

Our new TAG variant is extremely simple. We allow arbitrary initial and auxiliary trees, and place only one restriction on adjunction: we disallow adjunction at any node on the spine of an auxiliary tree below the root (though we discuss relaxing that constraint in Section 4.2). We refer to this variant as Off Spine TAG (OSTAG) and note that it allows the use of full wrapping rules, which are forbidden in TIG. This targeted blocking of recursion has similar motivations and benefits to the approximation of CFGs with regular languages (Mohri and jan Nederhof, 2000).

The following sections discuss in detail the context-free nature of OSTAG and alternative probabilistic models for its equivalent CFG form. We propose a simple but empirically effective heuristic for grammar induction for our experiments on Penn Treebank data.

3 Transformation to CFG

To demonstrate that OSTAG has only context-free power, we provide a reduction to context-free grammar. Given an OSTAG $\langle N, T, R, A, S\rangle$, we define the set $\mathcal{N}$ of nodes of the corresponding CFG to be pairs of a tree in R or A together with an

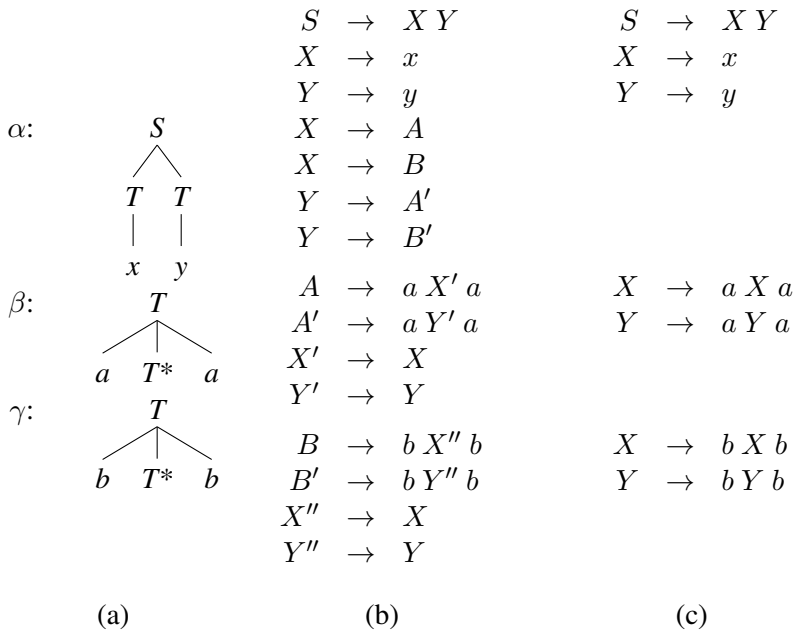


Figure 3: (a) OSTAG for the language wxw^Rvyv^R where $w, v \in \{a|b\}+$ and R reverses a string. (b) A CFG for the same language, which of necessity must distinguish between nonterminals X and Y playing the role of T in the OSTAG. (c) Simplified CFG, conflating nonterminals, but which must still distinguish between X and Y.

address (Gorn number (Gorn, 1965)) in that tree. We take the nonterminals of the target CFG grammar to be nodes or pairs of nodes, elements of the set $\mathcal{N} + \mathcal{N} \times \mathcal{N}$. We notate the pairs of nodes with a kind of "applicative" notation. Given two nodes η and η', we notate a target nonterminal as $\eta(\eta')$.

Now for each tree τ and each interior node η in τ that is not on the spine of τ, with children $\eta_1, \ldots, \eta_k$, we add a context-free rule to the grammar

$$\eta \rightarrow \eta_1 \cdots \eta_k \tag{1}$$

and if interior node η is on the spine of τ with η_s the child node also on the spine of τ (that is, dominating the foot node of τ) and η' is a node (in any tree) where τ is adjoinable, we add a rule

$$\eta(\eta') \rightarrow \eta_1 \cdots \eta_s(\eta') \cdots \eta_k \quad . \tag{2}$$

Rules of type (1) handle the expansion of a node not on the spine of an auxiliary tree and rules of type (2) a spinal node.

In addition, to initiate adjunction at any node η' where a tree τ with root η is adjoinable, we use a rule

$$\eta' \rightarrow \eta(\eta') \tag{3}$$

and for the foot node η_f of τ, we use a rule

$$\eta_f(\eta) \rightarrow \eta \tag{4}$$

The OSTAG constraint follows immediately from the structure of the rules of type (2). Any child spine node η_s manifests as a CFG nonterminal $\eta_s(\eta')$. If child spine nodes themselves allowed adjunction, we would need a type (3) rule of the form $\eta_s(\eta') \rightarrow \eta_s(\eta')(\eta'')$. This rule itself would feed adjunction, requiring further stacking of nodes, and an infinite set of CFG nonterminals and rules. This echoes exactly the stacking found in the LIG reduction of TAG .

To handle substitution, any frontier node η that allows substitution of a tree rooted with node η' engenders a rule

$$\eta \rightarrow \eta' \tag{5}$$

This transformation is reversible, which is to say that each parse tree derived with this CFG implies exactly one OSTAG derivation, with substitutions and adjunctions coded by rules of type (5) and (3) respectively. Depending on the definition of a TAG derivation, however, the converse is not necessarily true. This arises from the spurious ambiguity between adjunction at a substitution site (before applying a type (5) rule) versus the same adjunction at the root of the substituted initial tree (after applying a type (5) rule). These choices lead to different derivations in CFG form, but their TAG derivations can be considered conceptually

identical. To avoid double-counting derivations, which can adversely effect probabilistic modeling, type (3) and type (4) rules in which the side with the unapplied symbol is a nonterminal leaf can be omitted.

3.1 Example

The grammar of Figure 3(a) can be converted to a CFG by this method. We indicate for each CFG rule its type as defined above the production arrow. All types are used save type (5), as substitution is not employed in this example. For the initial tree α, we have the following generated rules (with nodes notated by the tree name and a Gorn number subscript):

$$\begin{array}{lcl} \alpha_\epsilon & \xrightarrow{1} & \alpha_1\,\alpha_2 \\ \alpha_1 & \xrightarrow{1} & x \\ \alpha_2 & \xrightarrow{1} & y \end{array} \qquad \begin{array}{lcl} \alpha_1 & \xrightarrow{3} & \beta_\epsilon(\alpha_1) \\ \alpha_1 & \xrightarrow{3} & \gamma_\epsilon(\alpha_1) \\ \alpha_2 & \xrightarrow{3} & \beta_\epsilon(\alpha_2) \\ \alpha_2 & \xrightarrow{3} & \gamma_\epsilon(\alpha_2) \end{array}$$

For the auxiliary trees β and γ we have:

$$\begin{array}{lcl} \beta_\epsilon(\alpha_1) & \xrightarrow{2} & a\,\beta_1(\alpha_1)\,a \\ \beta_\epsilon(\alpha_2) & \xrightarrow{2} & a\,\beta_1(\alpha_2)\,a \\ \beta_1(\alpha_1) & \xrightarrow{4} & \alpha_1 \\ \beta_1(\alpha_2) & \xrightarrow{4} & \alpha_2 \end{array}$$

$$\begin{array}{lcl} \gamma_\epsilon(\alpha_1) & \xrightarrow{2} & b\,\gamma_1(\alpha_1)\,b \\ \gamma_\epsilon(\alpha_2) & \xrightarrow{2} & b\,\gamma_1(\alpha_2)\,b \\ \gamma_1(\alpha_1) & \xrightarrow{4} & \alpha_1 \\ \gamma_1(\alpha_2) & \xrightarrow{4} & \alpha_2 \end{array}$$

The grammar of Figure 3(b) is simply a renaming of this grammar.

4 Applications

4.1 Compact grammars

The OSTAG framework provides some leverage in expressing particular context-free languages more compactly than a CFG or even a TSG can. As an example, consider the language of bracketed palindromes

$$Pal = \left\{ a_i\, w\, a_i\, w^R\, a_i \;\middle|\; \begin{array}{l} 1 \le i \le k \\ w \in \{b_j \mid 1 \le j \le m\}^* \end{array} \right.$$

containing strings like $a_2\, b_1 b_3\, a_2\, b_3 b_1\, a_2$. Any TSG for this language must include as substrings some subpalindrome constituents for long enough strings. Whatever nonterminal covers such a string, it must be specific to the a index within it, and must introduce at least one pair of bs as well. Thus, there are at least m such nonterminals, each introducing at least k rules, requiring at least km rules overall. The simplest such grammar, expressed as a CFG, is in Figure 4(a). The ability to use adjunction allows expression of the same language as an OSTAG with $k + m$ elementary trees (Figure 4(b)). This example shows that an OSTAG can be quadratically smaller than the corresponding TSG or CFG.

4.2 Extensions

The technique in OSTAG can be extended to expand its expressiveness without increasing generative capacity.

First, OSTAG allows zero adjunctions on each node on the spine below the root of an auxiliary tree, but any non-zero finite bound on the number of adjunctions allowed on-spine would similarly limit generative capacity. The tradeoff is in the grammar constant of the effective probabilistic CFG; an extension that allows k levels of on spine adjunction has a grammar constant that is $O(|\mathcal{N}|^{(k+2)})$.

Second, the OSTAG form of adjunction is consistent with the TIG form. That is, we can extend OSTAG by allowing on-spine adjunction of left- or right-auxiliary trees in keeping with the TIG constraints without increasing generative capacity.

4.3 Probabilistic OSTAG

One major motivation for adherence to a context-free grammar formalism is the ability to employ algorithms designed for probabilistic CFGs such as the CYK algorithm for parsing or the Inside-Outside algorithm for grammar estimation. In this section we present a probabilistic model for an OSTAG grammar in PCFG form that can be used in such algorithms, and show that many parameters of this PCFG can be pooled or set equal to one and ignored. References to rules of types (1-5) below refer to the CFG transformation rules defined in Section 3. While in the preceeding discussion we used Gorn numbers for clarity, our discussion applies equally well for the Goodman transform discussed above, in which each node is labeled with a signature of its subtree. This simply redefines η in the CFG reduction described in Section 3 to be a subtree indicator, and dramatically reduces redundancy in the generated grammar.

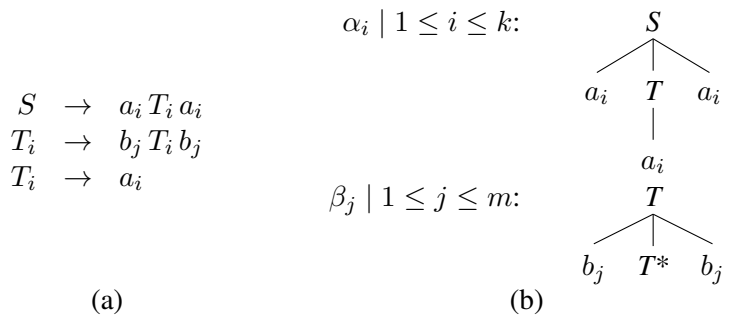


Figure 4: A CFG (a) and more compact OSTAG (b) for the language *Pal*

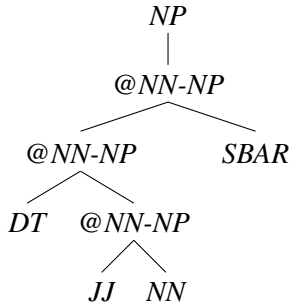
The first practical consideration is that CFG rules of type (2) are deterministic, and as such we need only record the rule itself and no associated parameter. Furthermore, these rules employ a template in which the stored symbol appears in the left-hand side and in exactly one symbol on the right-hand side where the spine of the auxiliary tree proceeds. One deterministic rule exists for this template applied to each η, and so we may record only the template. In order to perform CYK or IO, it is not even necessary to record the index in the right-hand side where the spine continues; these algorithms fill a chart bottom up and we can simply propagate the stored nonterminal up in the chart.

CFG rules of type (4) are also deterministic and do not require parameters. In these cases it is not necessary to record the rules, as they all have exactly the same form. All that is required is a check that a given symbol is adjoinable, which is true for all symbols except nonterminal leaves and applied symbols. Rules of type (5) are necessary to capture the probability of substitution and so we will require a parameter for each.

At first glance, it would seem that due to the identical domain of the left-hand sides of rules of types (1) and (3) a parameter is required for each such rule. To avoid this we propose the following factorization for the probabilistic expansion of an off spine node. First, a decision is made as to whether a type (1) or (3) rule will be used; this corresponds to deciding if adjunction will or will not take place at the node. If adjunction is rejected, then there is only one type (1) rule available, and so parameterization of type (1) rules is unnecessary. If we decide on adjunction, one of the available type (3) rules is chosen from a multinomial. By conditioning the probability of adjunction on varying amounts of information about the node, alternative models can easily be defined.

5 Experiments

As a proof of concept, we investigate OSTAG in the context of the classic Penn Treebank statistical parsing setup; training on section 2-21 and testing on section 23. For preprocessing, words that occur only once in the training data are mapped to the unknown categories employed in the parser of Petrov et al. (2006). We also applied the annotation from Klein and Manning (2003) that appends "-U" to each nonterminal node with a single child, drastically reducing the presence of looping unary chains. This allows the use of a coarse to fine parsing strategy (Charniak et al., 2006) in which a sentence is first parsed with the Maximum Likelihood PCFG and only constituents whose probability exceeds a cutoff of 10^{-4} are allowed in the OSTAG chart. Designed to facilitate sister adjunction, we define our binarization scheme by example in which the added nodes, indicated by @, record both the parent and head child of the rule.



A compact TSG can be obtained automatically using the MCMC grammar induction technique of Cohn and Blunsom (2010), retaining all TSG rules that appear in at least one derivation in after 1000 iterations of sampling. We use EM to estimate the parameters of this grammar on sections 2-21, and use this as our baseline.

To generate a set of TAG rules, we consider each rule in our baseline TSG and find all possi-

	All	40	#Adj	#Wrap
TSG	85.00	86.08	–	–
TSG′	85.12	86.21	–	–
OSTAG^1	85.42	86.43	1336	52
OSTAG^2	85.54	86.56	1952	44
OSTAG^3	85.86	86.84	3585	41

Figure 5: Parsing F-Score for the models under comparison for both the full test set and sentences of length 40 or less. For the OSTAG models, we list the number of adjunctions in the MPD of the full test set, as well as the number of wrapping adjunctions.

ble auxiliary root and foot node pairs it contains. For each such root/foot pair, we include the TAG rule implied by removal of the structure above the root and below the foot. We also include the TSG rule left behind when the adjunction of this auxiliary tree is removed. To be sure that experimental gains are not due to this increased number of TSG initial trees, we calculate parameters using EM for this expanded TSG and use it as a second baseline (TSG′). With our full set of initial and auxiliary trees, we use EM and the PCFG reduction described above to estimate the parameters of an OSTAG.

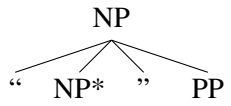
We investigate three models for the probability of adjunction at a node. The first uses a conservative number of parameters, with a Bernoulli variable for each symbol (OSTAG^1). The second employs more parameters, conditioning on both the node's symbol and the symbol of its leftmost child (OSTAG^2).The third is highly parameterized but most prone to data sparsity, with a separate Bernoulli distribution for each Goodman index η (OSTAG^3). We report results for Most Probable Derivation (MPD) parses of section 23 in Figure 5.

Our results show that OSTAG outperforms both baselines. Furthermore, the various parameterizations of adjunction with OSTAG indicate that, at least in the case of the Penn Treebank, the finer grained modeling of a full table of adjunction probabilities for each Goodman index OSTAG^3 overcomes the danger of sparse data estimates. Not only does such a model lead to better parsing performance, but it uses adjunction more extensively than its more lightly parameterized alternatives. While different representations make direct comparison inappropriate, the OSTAG results lie in the same range as previous work with statistical TIG on this task, such as Chiang (2000) (86.00) and Shindo et al. (2011) (85.03).

The OSTAG constraint can be relaxed as described in Section 4.2 to allow any finite number of on-spine adjunctions without sacrificing context-free form. However, the increase to the grammar constant quickly makes parsing with such models an arduous task. To determine the effect of such a relaxation, we allow a single level of on-spine adjunction using the adjunction model of OSTAG^1, and estimate this model with EM on the training data. We parse sentences of length 40 or less in section 23 and observe that on-spine adjunction is never used in the MPD parses. This suggests that the OSTAG constraint is reasonable, at least for the domain of English news text.

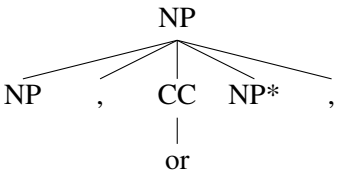
We performed further examination of the MPD using OSTAG for each of the sentences in the test corpus. As an artifact of the English language, the majority have their foot node on the left spine and would also be usable by TIG, and so we discuss the instances of wrapping auxiliary trees in these derivations that are uniquely available to OSTAG. We remove binarization for clarity and denote the foot node with an asterisk.

A frequent use of wrapping adjunction is to coordinate symbols such as quotes, parentheses, and dashes on both sides of a noun phrase. One common wrapping auxiliary tree in our experiments is



This is used frequently in the news text of the Wall Street Journal for reported speech when avoiding a full quotation. This sentence is an example of the way the rule is employed, using what Joshi and Schabes (1997) referred to as “factoring recursion from linguistic constraints” with TAG. Note that replacing the quoted noun phrase and its following prepositional phrase with the noun phrase itself yields a valid sentence, in line with the linguistic theory underlying TAG.

Another frequent wrapping rule, shown below, allows direct coordination between the contents of an appositive with the rest of the sentence.



This is a valuable ability, as it is common to use an appositive to provide context or explanation for a proper noun. As our information on proper nouns will most likely be very sparse, the appositive may be more reliably connected to the rest of the sentence. An example of this from one of the sentences in which this rule appears in the MPD is the phrase "since the market fell 156.83, or 8 %, a week after Black Monday". The wrapping rule allows us to coordinate the verb "fell" with the pattern "X %" instead of 156.83, which is mapped to an unknown word category.

These rules highlight the linguistic intuitions that back TAG; if their adjunction were undone, the remaining derivation would be a valid sentence that simply lacks the modifying structure of the auxiliary tree. However, the MPD parses reveal that not all useful adjunctions conform to this paradigm, and that left-auxiliary trees that are not used for sister adjunction are susceptible to this behavior. The most common such tree is used to create noun phrases such as

P&G's share of [the Japanese market]
the House's repeal of [a law]
Apple's family of [Macintosh Computers]
Canada's output of [crude oil]

by adjoining the shared unbracketed syntax onto the NP dominating the bracketed text. If adjunction is taken to model modification, this rule drastically changes the semantics of the unmodified sentence. Furthermore, in some cases removing the adjunction can leave a grammatically incorrect sentence, as in the third example where the noun phrase changes plurality.

While our grammar induction method is a crude (but effective) heuristic, we can still highlight the qualities of the more important auxiliary trees by examining aggregate statistics over the MPD parses, shown in Figure 6. The use of left-auxiliary trees for sister adjunction is a clear trend, as is the predominant use of right-auxiliary trees for the complementary set of "regular" adjunctions, which is to be expected in a right branching language such as English. The statistics also reflect the importance of substitution in right-auxiliary trees, as they must capture the wide variety of right branching modifiers of the English language.

	All	Wrap	Right	Left
Total	3585 (1374)	41 (26)	1698 (518)	1846 (830)
Sister	2851 (1180)	17 (11)	1109 (400)	1725 (769)
Lex	2244 (990)	28 (19)	894 (299)	1322 (672)
FLex	1028 (558)	7 (2)	835 (472)	186 (84)

Figure 6: Statistics for MPD auxiliary trees using OSTAG[3]. The columns indicate type of auxiliary tree and the rows correspond respectively to the full set found in the MPD, those that perform sister adjunction, those that are lexicalized, and those that are fully lexicalized. Each cell shows the number of tokens followed by the number of types of auxiliary tree that fit its conditions.

6 Conclusion

The OSTAG variant of Tree-Adjoining Grammar is a simple weakly context-free formalism that still provides for all types of adjunction and is a bit more concise than TSG (quadratically so). OSTAG can be reversibly transformed into CFG form, allowing the use of a wide range of well studied techniques in statistical parsing.

OSTAG provides an alternative to TIG as a context-free TAG variant that offers wrapping adjunction in exchange for recursive left/right spine adjunction. It would be interesting to apply both OSTAG and TIG to different languages to determine where the constraints of one or the other are more or less appropriate. Another possibility is the combination of OSTAG with TIG, which would strictly expand the abilities of both approaches.

The most important direction of future work for OSTAG is the development of a principled grammar induction model, perhaps using the same techniques that have been successfully applied to TSG and TIG. In order to motivate this and other related research, we release our implementation of EM and CYK parsing for OSTAG[1]. Our system performs the CFG transform described above and optionally employs coarse to fine pruning and relaxed (finite) limits on the number of spine adjunctions. As a TSG is simply a TAG without adjunction rules, our parser can easily be used as a TSG estimator and parser as well.

[1] bllip.cs.brown.edu/download/bucketparser.tar

References

Eugene Charniak, Mark Johnson, Micha Elsner, Joseph L. Austerweil, David Ellis, Isaac Haxton, Catherine Hill, R. Shrivaths, Jeremy Moore, Michael Pozar, and Theresa Vu. 2006. Multilevel coarse-to-fine PCFG parsing. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Eugene Charniak. 1996. Tree-bank grammars. In *Association for the Advancement of Artificial Intelligence*, pages 1031–1036.

David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. Association for Computational Linguistics.

Trevor Cohn and Phil Blunsom. 2010. Blocked inference in bayesian tree substitution grammars. pages 225–230. Association for Computational Linguistics.

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

Christy Doran, Dania Egedi, Beth Ann Hockey, Bangalore Srinivas, and Martin Zaidel. 1994. XTAG system: a wide coverage grammar for English. pages 922–928. Association for Computational Linguistics.

J. Goodman. 2003. Efficient parsing of DOP with PCFG-reductions. *Bod et al. 2003*.

Saul Gorn. 1965. Explicit definitions and linguistic dominoes. In *Systems and Computer Science*, pages 77–115.

Rebecca Hwa. 1998. An empirical evaluation of probabilistic lexicalized tree insertion grammars. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 557–563. Association for Computational Linguistics.

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69–124. Springer.

Aravind K Joshi. 1985. *Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions?* University of Pennsylvania.

Aravind K Joshi. 1987. An introduction to tree adjoining grammars. *Mathematics of Language*, pages 87–115.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. pages 423–430. Association for Computational Linguistics.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, pages 35–56.

Mehryar Mohri and Mark jan Nederhof. 2000. Regular approximation of context-free grammars through transformation. In *Robustness in language and speech technology*.

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 the 44th annual meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics.

Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, (4):479–513.

Yves Schabes. 1990. *Mathematical and computational aspects of lexicalized grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.

Hiroyuki Shindo, Akinori Fujino, and Masaaki Nagata. 2011. Insertion operator for bayesian tree substitution grammars. pages 206–211. Association for Computational Linguistics.

Elif Yamangil and Stuart M. Shieber. 2012. Estimating compact yet rich tree insertion grammars. pages 110–114. Association for Computational Linguistics.