Rebecca Nesson, Stuart M. Shieber, and Alexander Rush. 2006. Induction of Probabilistic Synchronous Tree-Insertion Grammars for Machine Translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA 2006)*, Boston, Massachusetts, 8–12 August.

Induction of Probabilistic Synchronous Tree-Insertion Grammars for Machine Translation

Rebecca Nesson, Stuart Shieber, and Alexander Rush
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
{nesson,shieber,rush}@eecs.harvard.edu

Abstract

The more expressive and flexible a base formalism for machine translation is, the less efficient parsing of it will be. However, even among formalisms with the same parse complexity, some formalisms better realize the desired characteristics for machine translation formalisms than others. We introduce a particular formalism, probabilistic synchronous tree-insertion grammar (PSTIG) that we argue satisfies the desiderata optimally within the class of formalisms that can be parsed no less efficiently than context-free grammars and demonstrate that it outperforms state-of-the-art word-based and phrase-based finite-state translation models on training and test data taken from the EuroParl corpus (Koehn, 2005). We then argue that a higher level of translation quality can be achieved by hybridizing our induced model with elementary structures produced using supervised techniques such as those of Groves et al. (2004).

1 Introduction

In this paper we identify a base formalism, probabilistic synchronous tree-insertion grammar (PSTIG), for a statistical machine translation system that we propose:

1. maximizes, within its efficiency class, the quality of the MT system induced unsupervised from aligned sentence pairs; and
2. is suitable for hybridization with linguistically-motivated elementary structures (constructed manually, or obtained automatically with or without supervision).

We begin with an argument from first principles for the choice of PSTIG as a base formalism for syntax-aware statistical machine translation (SMT). We then present our implementation of a system that induces a PSTIG unsupervised from data and show that it outperforms a state-of-the-art phrase-based SMT system in both automatic and human evaluation. We conclude by proposing a method for hybridizing our system to include linguistically-motivated elementary structures that draws on recent results in Data-Oriented Parsing.

In Section 2 we place our work in the context of similar work in the field and argue for the efficacy of choosing a base formalism in the class of formalisms that can be processed no less efficiently than context-free grammars. In Section 3, we present a formalism, probabilistic synchronous tree-insertion grammar (PSTIG), that more fully satisfies the desired characteristics than context-free grammars. In Sections 4 and 5 we give the parsing algorithm and discuss how we induce a PSTIG grammar from data by synchronous parsing. We demonstrate in Section 6 that the induced model outperforms both word-based and phrase-based finite-state models on a subset of the EuroParl corpus. We suggest in Section 7 that, without increasing the expressivity of the base formalism beyond context-free, substantially higher quality translations can be produced by unsupervised induction systems that can easily be hybridized with

linguistically-motivated elementary structures generated manually or though a supervised process. In particular, we propose a method of hybridizing our system by adding elementary structures generated using the methods of Groves et al. (2004) in a manner similar to that used by Groves and Way (2005).

2 Motivation and Related Work

Recent work in statistical machine translation by parsing has identified a set of characteristics an ideal base formalism should have for the translation task (Melamed, 2003; Melamed, 2004; Melamed et al., 2004). What is desired is a formalism that has the *substitution*-based hierarchical structure of context-free grammars and the *lexical* relationship potential of n-gram models. Further, it should allow for *discontinuity* in phrases and be *synchronizable*, to allow for multilinguality. Finally, in order to support automated induction, it should allow for a *probabilistic* variant, and a reasonably *efficient* parsing algorithm. The more expressive and flexible a formalism is, the less efficient parsing of it will be. Therefore, the primary trade-off to be made is between parsing efficiency on one hand and the rest of the desired characteristics on the other. However, even among formalisms with the same parse complexity, some formalisms better satisfy the desiderata than others.

Finite-state word-based models, such as IBM Model 5 (Brown et al., 1993), use a base formalism that allows for synchronization, probabilistic variants, very efficient processing, and good ability to capture lexical and bilexical relationships. However, they are limited by the inability to use hierarchical information in the interlingual mapping. That bilingual dictionaries describe the mappings between languages in terms of constructions, not individual words, suggests that this information would be useful. For instance, the *HarperCollins Italian College Dictionary* (HCICD) translates the English "to take advantage of" as "sfruttare", although that word is a direct translation of neither "take" nor "advantage".

Retaining the same finite-state base formalism, these models can be augmented to allow multiword (in addition to single word) mapping. Marcu and Wong (2002), among others, use joint probability distributions over frequently co-occuring n-grams to find multiword translations, thereby improving on the performance of IBM Model 5. Such an approach does allow multiword relationships to be induced, but does not in any sense incorporate syntactic structure to do so. Indeed the natural way to augment the multiword approach to incorporate syntactic constraints is to restrict the multiword sequences to syntactic constituents (as determined by a statistical parser for instance) (Yamada and Knight, 2002). Yet this augmentation turns out to underperform the syntax-free variant (Koehn et al., 2003).

The reason is not hard to understand: the word sequences that map well in translation—such as the German-English example of Koehn et al. (2003) "es gibt"/"there is"—are not themselves syntactic constituents, but rather syntactic templates ("es gibt…"/"there is…") with "holes" (marked here by ellipses) that might be substituted for in some uniform manner. Bilingual dictionaries even make the mapping between such constructions explicit through the use of place fillers like "sb" ("somebody") or "qn" ("qualcuno"), as in the HCICD entry "to drive sb mad"/"far impazzire qn". Secondarily, the phrases that are mapped need not appear contiguously, either because the holes split the lexical material, as in the example "drive sb mad", or because other constituents interpose themselves, as in the phrase "take advantage yesterday of sb".[1]

This ability to substitute subparts is the hallmark of context-free grammars. A natural approach, then is to incorporate synchronization of context-free structures to allow for these kinds of mappings. However, probabilistic context-free grammars (PCFG) are well known to perform poorly as language models compared to finite-state models; they gain the ability to substitute according to abstract categories at the expense of stating lexical relationships directly. Although arbitrary CFGs can be weakly lexicalized by other CFGs, this can require changing the shape of the derived trees produced, and more critically, changes the structure of the derivation (Schabes and Waters, 1993a; Schabes and Waters, 1995). Because synchronization requires substantial isomorphism of the derivation trees, synchronization of lexicalized CFGs becomes problematic. Chiang (2005) overcomes this short-

[1] For example, "The US took advantage yesterday of the political and military momentum in its Afghan campaign…" is one of many Google hits on the phrase "took advantage yesterday of".

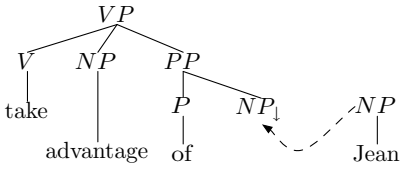

Figure 1: An example TAG/TIG substitution

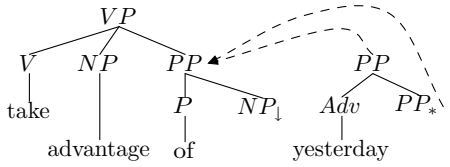

Figure 2: An example TAG/TIG adjunction

coming in his synchronous CFG-based system by making it both hierachical and phrase-based so that n-grams used in phrasal mappings could still capture some of the lexical dependencies. His system outperformed Pharaoh, a state-of-the-art phrase-based decoder (Koehn, 2004), on several translation tasks.

Although systems such as Chiang's are the current state-of-the-art, because of the limitations of CFGs as a base formalism Melamed and others continue to explore the possibility of trading off more parsing efficiency for greater expressivity (Melamed, 2004; Melamed et al., 2004). Formalisms such as Generalized Multitext Grammars (GMTG) do in principle satisfy all of the desiderata if the higher time and space complexity of the parsing algorithms for them does not make training prohibitively expensive (Melamed et al., 2004). It is an empirical question whether systems with a high degree of parse complexity can be induced in practice. Burbank et al. (2005) implemented a framework in which base formalisms such as GMTG can be tested though no substantial results have yet been reported. Ding and Palmer (2005) also employ a more expressive formalism but use heuristic approaches to limit the complexity of the processing.

All of these considerations led us to seek a more expressive formalism that could still be parsed efficiently. As we will argue, probabilistic synchronous tree-insertion grammar substantially satisfies each of the desiderata without increasing parse complexity. We present an MT system based on it in the remainder of this paper.

3 Synchronous Tree-Insertion Grammar

Tree-adjoining grammars (TAG), introduced in monolingual form by Joshi (1985), and in a synchronous variant (STAG) by Shieber and Schabes (1990), are natural choices to capture lexically-based dependencies while also allowing the substitution of sub-parts. Due to space limitations, for a detailed description of the TAG formalism we refer readers to the introduction by Joshi (1985). Importantly, Schabes et al. (1988) show that TAG can lexicalize CFG without changing the trees produced. That is, given a CFG a lexicalized TAG can be constructed that will produce the same set of derived structures produced by the CFG. Because each elementary tree contains a lexical item, the operations of *substitution* and *adjunction* implicitly manifest a lexical relationship. In addition, the two operations of TAG, substitution and adjunction, are exactly what is needed to handle noncontiguity, as shown in Figures 1 and 2.

However, the TAG formalism's additional expressivity leads to additional processing complexity. TAG parsing requires $O(n^6)$ time; synchronous TAG parsing would therefore require at least $O(n^{12})$ time. Because training an MT system based on synchronous TAG would require repeated parsing of the training corpus, this time complexity is prohibitive.

Tree-insertion grammars (TIG) are a computationally attractive alternative to TAG (Schabes and Waters, 1993a). TIGs are similar to TAGs except that restrictions are placed on the form of elementary trees and on the adjunction operation. In particular, the foot node of an auxiliary tree is required to be at the left or right edge of the frontier, so that all textual material dominated by the spine will fall to the right or left, respectively, of the foot. The auxiliary trees can thus be classified as either right or left auxiliary trees, as determined by the location of the non-foot material. To maintain the invariant that textual material falls only on a single side of the spine, adjunction is restricted so that left auxiliary trees may not adjoin into a node on the spine of a right auxiliary tree and vice versa. This prevents the formation of "wrapping" trees in which there are terminal symbols on both sides of the foot node. This restriction, coupled with the requirement

$$\textsc{WordAx} \quad \frac{}{\langle [\eta_S, (i, i+1)], [\eta_T, (l, l+1)], \emptyset, 1 \rangle} \quad \begin{array}{l} w_{i+1} = Label(\eta_S) \\ v_{l+1} = Label(\eta_T) \end{array}$$

$$\textsc{AuxAx} \quad \frac{}{\langle [\eta_S, (i, i)], [\eta_T, (l, l)], \emptyset, 1 \rangle} \quad \begin{array}{l} Foot(\eta_S) \\ Foot(\eta_T) \end{array}$$

$$\textsc{EmptyAx} \quad \frac{}{\langle [\eta_\epsilon, (i, i)], [\eta_\epsilon, (j, j)], \{(x, y)\}, 1 \rangle} \quad \begin{array}{l} x, y \in \{L, R\} \\ EmptyTree(\eta_\epsilon) \end{array}$$

Figure 3: Axioms for CKY-style PSTIG parsing

that all elementary auxiliary trees be non-wrapping, is sufficient to limit the formalism to context-free expressivity and $O(n^3)$ parsability. In addition, Schabes and Waters (Schabes and Waters, 1995) demonstrate that TIG, like TAG, can lexicalize CFGs without changing the shape of the trees produced. Hwa (2001) shows that a probabilistic variant of TIG can have language modeling performance at the level of bigram models thereby capturing lexical relationships, while also retaining the advantages of CFGs in capturing syntactic structure.[2]

Synchronous TIG (STIG) extends TIG by making elementary structures pairs of TIG trees with links between particular nodes in those trees. An STIG is a set of triples, $\langle t_L, t_R, \frown \rangle$ where t_L and t_R are elementary TIG trees and $\frown$ is the linking relation between nodes in t_L and t_R (Shieber, 1994). Derivation proceeds as in TIG except that all operations must be paired. That is, a tree can only be substituted or adjoined at a node if its pair is simultaneously substituted or adjoined at a linked node.

TIGs are equivalent to CFGs in weak-generative capacity, and in fact can be converted to weakly-equivalent CFGs, as shown by Schabes and Waters (1995). This raises the alternate possibility of processing STIGs by conversion to synchronous CFGs. However, the simple methods for parsing STIGs, essentially CKY parsing, do not carry over to the corresponding SCFG directly. (The CFG rules tend to have long right-hand sides, and are not lexicalized.) Without optimization, parse complexity would be greatly increased. It may be possible to optimize the resulting SCFG to achieve $O(n^6)$ parse complexity, but parsing the STIG is already this efficient. The STIG grammar can, in effect, be seen as an optimized version of the SCFG at the outset. For this reason (in addition the naturalness of the STIG approach and its potential for hybridization with linguistically informed STIG or STAG lexical entries) we use STIG as the basis for our translation system rather than converting the STIG grammar to an SCFG, optimizing, and using one of the existing CFG-based systems.

A synchronous TIG can easily express lexically-based dependencies, can (under some restrictions discussed in the next section) be parsed in $O(n^6)$ time, and can handle both the substitution and adjunction requirements described above. Thus, a probabilistic variant of synchronous TIG possesses an appealing balance of the desired characteristics that we would like as the basis for a syntax-aware translation formalism.

4 Parsing Synchronous Tree-Insertion Grammars

To define a STIG parsing algorithm we generalize Schabes and Waters' (1993a) $O(n^3)$ CKY-style lexicalized CFG parsing algorithm. In conjunction with a standard chart parsing algorithm the rules described below define an $O(n^6)$ parser for a restricted STIG parsing a pair of strings $w_1 \ldots w_{n_s}, v_1 \ldots v_{n_T}$.

We present this algorithm in Figures 3-5 as a set of inference rules in the deductive parsing style of Shieber et al. (1994). As with all deductive parsing algorithms, the algorithm works by generating items. Each item is of the form $\langle [\eta_S, I], [\eta_T, J], LinkSet, SV \rangle$, where η_S and η_T are nodes in some elementary tree pair of the grammar,[3] I is an interval (i, j) between string positions i and j characterizing the substring $w_{i+1} \ldots w_j$ covered by the item (and similarly for J, an interval in V), and $LinkSet$ specifies the

[2] For further background and discussion of TIGs and LTIGs, see (Schabes and Waters, 1993a; Schabes and Waters, 1993b; Schabes and Waters, 1995; Hwa, 2001).

[3] A node may be thought of as specified by the name of the tree and a unique address for that node in the tree.

$$\textsc{SibCat}\quad \frac{\langle[\eta_{1S}, I_1], [\eta_{1T}, J_1], \emptyset, P_1\rangle\ \langle[\eta_{2S}, I_2], [\eta_{2T}, J_2], \emptyset, P_2\rangle}{\langle[\eta_S, I_1 \cup_L I_2], [\eta_T, J_1 \cup_x J_2], LS(\eta_S, \eta_T), P_1 \otimes P_2\rangle} \quad \begin{array}{l}\eta_S \rightarrow \eta_{1S}\ \eta_{2S} \\ (\eta_T \rightarrow \eta_{1T}\ \eta_{2T} \text{ or} \\ \eta_T \rightarrow \eta_{2T}\ \eta_{1T})\end{array}$$

$$\textsc{SParSrc}\quad \frac{\langle[\eta_{1S}, I], [\eta_T, J], \emptyset, P\rangle}{\langle[\eta_S, I], [\eta_T, J], LS(\eta_S, \eta_T), P\rangle} \quad \begin{array}{l}\eta_S \rightarrow \eta_{1S} \\ NoLinks(\eta_{1S})\end{array}$$

$$\textsc{SParTrg}\quad \frac{\langle[\eta_S, I], [\eta_{1T}, J], \emptyset, P\rangle}{\langle[\eta_S, I], [\eta_T, J], LS(\eta_S, \eta_T), P\rangle} \quad \begin{array}{l}\eta_T \rightarrow \eta_{1T} \\ NoLinks(\eta_{1T})\end{array}$$

$$\textsc{Subst}\quad \frac{\langle[\eta_{1S}, I], [\eta_{1T}, J], \emptyset, P_1\rangle}{\langle[\eta_{2S}, I], [\eta_{2T}, J], \emptyset, P_1 \otimes \theta_{(\overline{\eta_1}, \overline{\eta_2})}\rangle} \quad \begin{array}{l}RootPair(\overline{\eta_1}) \\ Subst(\overline{\eta_1}, \overline{\eta_2})\end{array}$$

$$\textsc{Adjoin}\quad \frac{\langle[\eta_{1S}, I_1], [\eta_{1T}, J_1], LS, P_1\rangle\ \langle[\eta_{2S}, I_2], [\eta_{2T}, J_2], \emptyset, P_2\rangle}{\langle[\eta_S, I_1 \cup_x I_2], [\eta_{1T}, J_1 \cup_y J_2], LS - (x, y), P_1 \otimes P_2 \otimes \theta_{(\overline{\eta_1}, \overline{\eta_2})}\rangle} \quad \begin{array}{l}RootPair(\overline{\eta_2}) \\ (x, y) \in LS \\ Adjoin(\overline{\eta_1}, \overline{\eta_2}, x, y)\end{array}$$

Figure 4: Inference Rules for CKY-style STIG parsing

set of (unused) links between the two nodes. SV is a semiring value associated with the item that can be thought of as the probability of that item being generated. (We discuss semirings and their use in our model in Section 5.) A goal item is of the form $\langle[\eta_S, (0, n_S)], [\eta_T, (0, n_T)], \emptyset, SV\rangle$, where the labels of η_S and η_T are the start symbols of the source and target sides of the grammar and the intervals $(0, n_S)$ and $(0, n_T)$ are complete covers of the source and target sentences, respectively. We use the notation $LS(\eta_S, \eta_T)$ to indicate the set of links between nodes η_S and η_T. We use the predicate $NoLinks(\eta)$ to signify that η has no links to any other node, including those not in the current item. We assume that each node η in an elementary tree is associated with a nonterminal or terminal label $Label(\eta)$. We notate the pair of nodes, η_S and η_T within an item by $\overline{\eta}$. Root nodes η_S and η_T of a tree pair satisfy $RootPair(\overline{\eta})$ and an item specifying a root pair is a *root item*. If η_{1S} and η_{1T} are the root nodes of a tree pair that can substitute or adjoin at nodes η_{2S} and η_{2T}, then $Subst(\overline{\eta_1}, \overline{\eta_2})$, $Adjoin(\overline{\eta_1}, \overline{\eta_2}, x, y)$ are respectively satisfied. We write $\eta_0 \rightarrow \eta_1 \ldots \eta_k$ to indicate that node η_0 dominates nodes $\eta_1, \ldots, \eta_k$. (As is standard for CKY-style algorithms, we assume that all trees are at most binary branching.) We make use of an interval union operation $\bigcup_x$, parameterized by the order in which the intervals abut, where x is either L or R, defined by

$$\begin{array}{rcl}(i, j) \cup_L (j, k) & = & (i, k) \\ (j, k) \cup_R (i, j) & = & (i, k)\end{array}$$

and is otherwise undefined.[4]

We briefly explicate the STIG inference rules shown in Figures 3 and 4 here. WORDAX adds items to the chart for each pair of nodes η_S and η_T labeled with words that appear in the source and target input sentences respectively. AUXAX adds items to the chart for the foot nodes of each auxiliary tree pair in the lexicon, making each auxiliary tree pair available for adjunction at each pair of string positions in the source and target input sentences. $EmptyAx$ adds a special $EmptyTree$ to the chart at every pair of string positions. This tree is a special single node auxiliary tree used to avoid having to maintain a separate parameter for when a particular link is not used in the course of a parse. Instead, in our system every link will be used, but they may be used by an $EmptyTree$ that does not change the shape or lexical content of the resulting tree.[5]

The sibling concatenation (SIBCAT) and single parent (SPARSRC and SPARTRG) rules simply move the derivation from child nodes to their parent nodes in the tree pair. They only apply when all adjunction operations on the child node are completed. The substitution rule (SUBST) applies at the root nodes of initial tree pairs whenever all adjunc-

[4]Note that x and y, as used with the $\cup$ operator and as members of link sets, are variables over the direction (L or R) in which the adjunction takes place on the source and target sides of the tree pair.

[5]The addition of $EmptyTree$s strictly speaking breaks lexicalization, but removes neither the linguistic advantages of lexicalization nor the parsing advantage that comes from disallowing adjunctions that do not increase the span of the item. The reason the latter is not a problem is that links can only be used once. Thus no spurious adjunctions are introduced.

$$\textsc{WordEpsAx} \quad \frac{}{\langle [\eta_S, (i, i+1)], [\eta_T, (l, l)], \emptyset, 1 \rangle} \quad \begin{array}{l} w_{i+1} = Label(\eta_S), \epsilon = Label(\eta_T) \\ Anchor(\eta_S, src, TP) \\ Anchor(\eta_T, trg, TP), TP \in G \end{array}$$

$$\textsc{EpsWordAx} \quad \frac{}{\langle [\eta_S, (i, i)], [\eta_T, (l, l+1)], \emptyset, 1 \rangle} \quad \begin{array}{l} w_{l+1} = Label(\eta_T), \epsilon = Label(\eta_S) \\ Anchor(\eta_S, src, TP) \\ Anchor(\eta_T, trg, TP), TP \in G \end{array}$$

Figure 5: Axioms for introducing tree pairs with one ϵ anchored tree.

tion operations are completed at those nodes. It creates new items for each pair of nodes (within a single elementary tree pair) into which the completely processed initial tree could substitute, covering the span of the input string that the completed initial tree covers. The adjunction rule (ADJOIN) applies whenever a pair of nodes have an adjunction available in their $LinkSet$ and there is an auxiliary tree pair that can adjoin at those nodes without violating the TIG wrapping tree restrictions.

Adding the two axioms in Figure 5 lets the rules generate translations of differing length. These axioms relax lexicalization so that only one of the trees in a pair is required to have a non-empty anchor.

Another method for allowing sentences of differing length is to allow parsing to proceed asynchronously on just one side of a tree pair where possible. This method allows a restricted amount of non-isomorphism as well. Nesson and Shieber (2005) present a set of inference rules that implement this solution.

Parsing STIGs requires only a straightforward generalization of well-understood parsing algorithms such as CKY or Earley's algorithm. However, the procedural order for visiting the nodes in a tree dictated by the parsing algorithm may conflict with the order in which linked nodes need to be processed. This means that the obvious generalizations of these parsing algorithms are not complete for the STIG languages. The difficulty arises because the links between trees may "cross" in ways that make it impossible for the parsing algorithm to perform operations using both links. This problem is similar to the one discussed by Melamed (2004) in the context of generalizing Eisner and Satta's (1999) splitting technique to the synchronous case for lexicalized multitext grammars (LMTG). Melamed opts to generalize a less efficient algorithm in order to allow the crossing correspondences to be parsed. Rather than take the penalty in parsing efficiency, we opt to restrict the elementary structures to those that do not exhibit the crossing links (Nesson et al., 2005).

5 Model Induction and Optimization

The advantage of PSTIG as a formalism for MT is its naturalness in describing relations between constructions in different languages. Nonetheless, to make use of that ability it must be embeddable in a system that can show at least the robustness of performance of the finite-state methods that have become standard. In this section, we describe how to perform unsupervised induction of a PSTIG based on aligned corpora that by itself shows good translation performance.

The induced PSTIG is structured as a normal form grammar in which adjunction parameters are estimated by EM. The normal forms specify both the shape of the trees in the tree pair as well as the links between them. We elected to generalize the normal form Hwa (2001) found to be most effective for monolingual TIG induction. Hwa allows two auxiliary tree shapes that differ only in the orientation of the foot node relative to the root. Each normal form tree has both high and low adjunction sites into which other trees can adjoin. We generalize to tree pairs by combining the two monolingual normal forms in all four possible orientations and then choosing a canonical set of links between nodes in the tree pairs. In addition, we include up to four initial tree pairs rooted in the start symbol of the grammar and with a single adjunction sites into which the auxiliary trees can adjoin. The resulting set of normal form trees is shown in Figure 6.

For every observed word cooccurence in the training set, we introduce one of each of the normal form auxiliary tree pairs anchored by the cooccurring words. In order to handle sentence pairs in which the sentences have differing length, we also

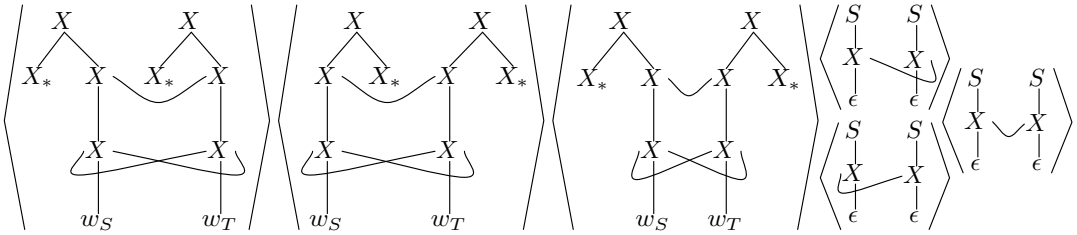

Figure 6: Canonical tree pair forms. Note that because of the set of links chosen, one auxiliary tree pair orientation and one initial tree pair are not needed because they cannot be used.

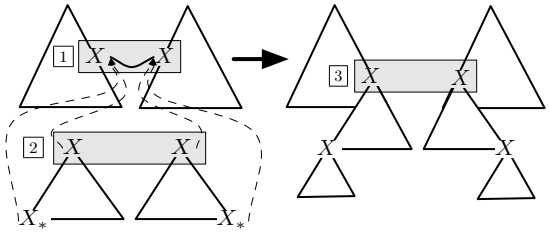

Figure 7: An example adjunction with items defined as follows: $\boxed{1}$: $\langle [\eta_{1S}, I_1], [\eta_{1T}, J_1], LS_1, P_1 \rangle$, $\boxed{2}$: $\langle [\eta_{2S}, I_2], [\eta_{2T}, J_2], \emptyset, P_2 \rangle$, $\boxed{3}$: $\langle [\eta_{1S}, I_1 \cup_L I_2], [\eta_{1T}, J_1 \cup_R J_2], LS_1 - (L, R), P_1 \otimes P_2 \otimes \theta_{(\overline{\eta_1}, \overline{\eta_2})} \rangle$

include tree pairs in which one tree in the pair is anchored by the empty string. The full grammar has $O(|S| \cdot |D|)$ tree pairs, where $|S|$ and $|D|$ are the size of the source and destination lexicons.

We use the semiring parsing methodology described by Goodman (1999) to make the grammar probabilistic. A semiring defines a set containing two binary operations $\oplus$ and $\otimes$ in addition to an additive identity 0 and a multiplicative identity 1. By varying the defintions used for these operators, the semiring can be used to calculate probabilities, derivation forests, and many other useful quantities. As shown in the inference rules in Figures 3-5, we associate a semiring value, denoted P_i for item i, with each item in the chart. Taking the probability semiring as an example, the semiring value for an item will be the probability that the pair of nodes in the item cover the intervals of the source and target sentences given in the intervals of the item. Each inference rule uses semiring operations to calculate the semiring value of the consequent item from the semiring values of the antecedent items and the parameters of the model. The most interesting operation is performed by the ADJOIN rule. As illustrated by Figure 7, this rule could be used to adjoin item 2 with semiring value P_2 and nodes $\overline{\eta_2}$ into tree pair 1 with semiring value P_1 and nodes $\overline{\eta_1}$. The new consequent item will be assigned the semiring value $P_1 \otimes P_2 \otimes \theta_{(\overline{\eta_1}, \overline{\eta_2})}$ where $\theta_{(\overline{\eta_1}, \overline{\eta_2})}$ is a parameter of the model indicating the probability of tree pair $\overline{\eta_2}$ adjoining into the tree pair containing nodes $\overline{\eta_1}$ at nodes $\overline{\eta_1}$. The values for the θ parameters are maintained for each link in the grammar as a distribution over all of the tree pairs in the grammar that can adjoin at that link. Our grammar has $O(n^2)$ tree pairs, where n is the size of the vocabulary. Since each tree can adjoin into every other tree, we maintain a total of $O(n^4)$ parameters. We also maintain a distribution over the initial trees used to root parses.

Our parser learns these parameters on an unsupervised bilingual corpus using an adaptation of the PCFG inside-outside algorithm developed by Lari and Young (1991). During training, we parse using the inside (probability) semiring which defines $\oplus$ as $+$, $\otimes$ as $\times$, and semiring values as real numbers.

To perform translation we use inference rules that are modified slightly to account for not having a target sentence at the time of parsing. For instance, we modify the WORDAX rule to remove the side condition that requires the anchor of the target tree to appear in the target sentence. We then parse using the Viterbi *n*-best derivation semiring, which produces a list of the *n* most probable derivation trees. Using the derivation trees it is trivial to generate the corresponding target language sentences. We currently use a simple trigram-based reranker learned from the corpus to choose the best translation. In

future work we plan to use a full discriminative reranker as described by Collins (2000).

5.1 Parameter Pruning

The full model presented above learns a probability for every combination of tree pairs in the corpus. In a corpus with high word cooccurence this results in $O(n^4)$ free parameters where n is the size of the largest monolingual vocabulary. The large number of parameters slows training and allows the model to "memorize" the training data by learning high probabilities for parameters that appear infrequently in the corpus. In order to handle both overfitting and training time constraints, we implemented two methods to prune unneeded trees and parameters.

Before training, we pre-process the word cooccurrence data by eliminating word pairs that are unlikely to encode true relationships. However, the preprocessing algorithm must be careful not to eliminate so many word pairs that some training sentences cannot be parsed. To ensure that all training sentences remain parsable, we use a greedy algorithm to learn one-to-one mappings for each sentence pair. The preprocessor first runs a simple word-to-word alignment tool (IBM Model 1) on the corpus to produce an initial alignment. It uses this alignment as the basis for a more sophisticated one-to-one alignment that additionally scores particular mappings.[6] Using the resulting alignments we repeatedly greedily match the highest scoring word pairs. We run the preprocessor in both directions and use the union of the two resulting alignments as the basis for our elementary structures.

Before learning all of the parameters of the model independently, we produce an intermediate model in which we tie certain parameters together. In particular, rather than maintaining a distribution over tree pairs at each *link* in the grammar, we maintain a distribution over tree pairs at each *link location*. We maintain parameters that correspond to *each* tree pair in the grammar, adjoining into *any* other tree pair at a particular link location. This reduces the number of parameters to $O(n^2)$. This both provides a good initial parameter setting for training with independent parameters and prevents the model from favoring rarely occurring interactions between particular tree pairs. We also use this model for linear interpolation smoothing during translation.

[6] This differs from IBM Model 2 only in that it does not allow for one-to-many mappings.

6 Results

We evaluated our system on a set of 15,277 sentences extracted from the over 600,000 sentence pairs in the EuroParl German-English corpus (Koehn, 2005). We selected sentence pairs in which the German sentence contained only words within the 1000 most frequent German words in the corpus, in order to limit the size of the vocabulary. We then further limited the set by using only sentence pairs with combined length less than 25.[7] We held out 100 sentence pairs for evaluation of the resulting system.

As baselines we trained GIZA++ (Och and Ney, 2003) using the CMU-Cambridge Statistical Language Modeling Toolkit (CMU Toolkit) and the ISI ReWrite Decoder for testing, and Pharaoh using alignments generated according to the algorithm given in Koehn (2003) based on GIZA++ word alignments and using the CMU Toolkit for language modeling.[8] We then evaluated all three systems on the test set automatically using BLEU score. We also ran a human evaluation in which three subjects evaluated all of the 100 translations produced by each system, in random order and with no indication of which system generated the translation, against the "gold standard" reference translations using a 5 point fluency and adequacy scale. As shown in Figure 8, our system outperformed both Pharaoh and GIZA++ in automatic evaluation. In human evaluation, also shown in Figure 8, our system outperformed both Pharaoh and GIZA++ in the fluency of the translations produced. It also outperformed Pharaoh in the adequacy of the translations produced. We were surprised to find that GIZA++ received the best scores from human evaluators for adequacy. We surmise this may be due to the close

[7] The primary reason for limiting the size of the test set and the length of sentences is due to the time required for parsing. Vocabulary size is not a significant factor. However, in order to overcome sparseness in a smaller training set we opted to select a set that used only a limited vocabulary.

[8] We were unfortunately unable to compare our system's performance either to Chiang's (2005) system, for which the code is not publicly available, or the GenPar system (Burbank et al., 2005) instantiated with an MTG or LMTG, because adaptation to a new language pair proved to be quite difficult. Needless to say, it would be helpful to perform these comparisons in order to empirically evaluate the claim that STIG is a better base formalism for translation than CFG.

	STIG	Pharaoh	GIZA++
BLEU	.2441	.2273	.2184
Fluency	3.38	3.25	3.32
Adequacy	3.13	2.96	3.22

Figure 8: Results of evaluating our system and the baseline systems on a 100 sentence test.

to word-for-word nature of the GIZA++ translations. Many of the PSTIG systems adequacy errors arose because negation words were dropped. This is just one example of the type of shortcoming that could be fixed by the addition of linguistically-motivated elementary structures for the relevant constructions.

7 Potential for Hybridization

Using a base formalism that can capture hierarchical relationships makes it possible for our system to express linguistically motivated relationships between words and constructions in the languages being translated. However, unsupervised grammar induction is likely to miss many of the true relationships and generalizations. We propose to hybridize our system by augmenting the set of elementary tree pairs available to the parser with linguistically motivated trees obtained either manually or automatically from a parsed, tagged corpus of aligned sentences. Groves and Way (2005) demonstrated the efficacy of this approach by adding (flat) phrases obtained using the method of Example-Based MT to the translation table used by the Pharaoh decoder and showing that the resulting system outperformed both the Example-Based MT system and Pharaoh without the additional phrases. In our system we expect even more substantial advantages because we will be able to use the hierarchical syntactic information encoded in elementary tree pairs. We propose to adapt the methods described by Groves et al. (2004) to extract elementary tree pairs with links from a corpus of parsed, aligned sentences. We will then experiment with at least two methods of hybridization. One option is to simply add the extracted trees to our initial set of elementary trees and allow the EM algorithm to estimate the probability of their use. A second option is to use the unsupervised model as a backoff model to smooth translation using the extracted tree pairs.

8 Conclusion

We have presented a formalism, probabilistic synchronous tree-insertion grammars, with several desirable properties for statistical machine-translation applications. First, we argued from first principles that PSTIG arises naturally as a means for expressing the kind of bilexical information found in bilingual dictionaries and acquired by phrase-based MT systems, indicating that it may well serve as a representation for the kind of information required for syntax-aware MT, whether acquired manually or through supervised learning techniques. Second, we showed that even used as a basis for unsupervised acquisition of syntax-aware statistical MT models, the formalism outperforms state-of-the-art finite-state word and phrase-based systems in initial tests. On the basis of these two properties, and the simplicity and uniformity of the formalism, we believe it may be ideal for the next generation of hybrid statistical MT systems. Certainly, further investigation is warranted.

Acknowledgements

This work was supported in part by grant IIS-0329089 from the National Science Foundation. We wish to thank Michael Collins, Rebecca Hwa, and Rani Nelken and the anonymous reviewers for valuable comments on earlier drafts. We also thank Rebecca Hwa for sharing her code and Paul Govereau and Daniel Mauer for work on the early stages of implementation.

References

P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, and R.L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

A. Burbank, M. Carpuat, S. Clark, M. Dreyer, P. Fox, D. Groves, K. Hall, M. Hearne, D. Melamed, Y. Shen, A. Way, B. Wellington, and D. Wu. 2005. Final report of the 2005 language engineering workshop on statistical machine translation by parsing.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *ACL 2005*, pages 263–270.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proc. 17th International*

Conf. on Machine Learning, pages 175–182. Morgan Kaufmann, San Francisco, CA.

Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL–05)*.

Jason Eisner and Giorgio Satta. 1999. Efficient parsing for bilexical context-free grammars and head-automaton grammars. In *Proceedings of the ACL*.

Joshua Goodman. 1999. Semiring parsing. *Computational Linguistics*, 25(4):573–605.

Declan Groves and Andy Way. 2005. Hybrid example-based SMT: the best of both worlds? In *Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, Ann Arbor, MI, June. ACL '05.

Declan Groves, Mary Hearne, and Andy Way. 2004. Robust sub-sentential alignment of phrase-structure trees. In *COLING '04, Geneva Switzerland*.

Rebecca Hwa. 2001. *Learning Probabilistic Lexicalized Grammars for Natural Language Processing*. Ph.D. thesis, Harvard University.

Aravind K. Joshi. 1985. *How Much Context-Sensitivity Is Necessary for Characterizing Structural Descriptions—Tree-Adjoining Grammar*. Cambridge University Press.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT/NAACL*.

Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Conference of the Association for Machine Translation in the Americas*, pages 115–124.

Philipp Koehn. 2005. Europarl: A parallel corpus for machine translation. In *MT Summit X*, Phuket, Thailand. International Association for Machine Translation.

K. Lari and S.J. Young. 1991. Applications of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 5:237–257.

D. Marcu and W. Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

I. Dan Melamed, Giorgio Satta, and Ben Wellington. 2004. Generalized multitext grammars. In *Proceedings of the 42nd Annual Conference of the Association for Computational Linguistics (ACL-04)*.

I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 79–86.

I. Dan Melamed. 2004. Statistical machine translation by parsing. In *Proceedings of the 42nd Annual Conference of the Association for Computational Linguistics (ACL-04)*.

Rebecca Nesson, Alexander Rush, and Stuart M. Shieber. 2005. Induction of probabilistic synchronous tree-insertion grammars. Technical Report TR-20-05, Division of Engineering and Applied Sciences, Harvard University, Cambridge, MA.

Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51.

Yves Schabes and Richard C. Waters. 1993a. Lexicalized context-free grammars. In *Proceedings of the 31st Conference on Association for Computational Linguistics*, pages 121–129. Association for Computational Linguistics.

Yves Schabes and Richard C. Waters. 1993b. Stochastic lexicalized context-free grammar. Technical Report 93–12, Mitsubishi Electric Research Laboratories.

Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar: A cubic time, parsable formalism that lexicalizes context-free grammars without changing the trees produced. *Computational Linguistics*, 21(3):479–512.

Yves Schabes, Anne Abeille, and Aravind K. Joshi. 1988. Parsing strategies with 'lexicalized' grammars: Application to tree-adjoining grammars. In *Proceedings of the 12th Conference on Computational Linguistics*, volume 2, pages 578–583.

Stuart M. Shieber and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*.

Stuart M. Shieber, Yves Schabes, and Fernando Pereira. 1994. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24:503–512.

Stuart M. Shieber. 1994. Restricting the weak-generative capacity of synchronous tree-adjoining grammars. *Computational Intelligence*, 10(4):371–385.

Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical MT. In *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics*.