Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars

Yuan Ding **Martha Palmer**
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA
{yding, mpalmer}@linc.cis.upenn.edu

Abstract

Syntax-based statistical machine translation (MT) aims at applying statistical models to structured data. In this paper, we present a syntax-based statistical machine translation system based on a probabilistic synchronous dependency insertion grammar. Synchronous dependency insertion grammars are a version of synchronous grammars defined on dependency trees. We first introduce our approach to inducing such a grammar from parallel corpora. Second, we describe the graphical model for the machine translation task, which can also be viewed as a stochastic tree-to-tree transducer. We introduce a polynomial time decoding algorithm for the model. We evaluate the outputs of our MT system using the NIST and Bleu automatic MT evaluation software. The result shows that our system outperforms the baseline system based on the IBM models in both translation speed and quality.

1 Introduction

Statistical approaches to machine translation, pioneered by (Brown et al., 1993), achieved impressive performance by leveraging large amounts of parallel corpora. Such approaches, which are essentially stochastic string-to-string transducers, do not explicitly model natural language syntax or semantics. In reality, pure statistical systems sometimes suffer from ungrammatical outputs, which are understandable at the phrasal level but sometimes hard to comprehend as a coherent sentence.

In recent years, syntax-based statistical machine translation, which aims at applying statistical models to structural data, has begun to emerge. With the research advances in natural language parsing, especially the broad-coverage parsers trained from treebanks, for example (Collins, 1999), the utilization of structural analysis of different languages has been made possible. Ideally, by combining the natural language syntax and machine learning methods, a broad-coverage and linguistically well-motivated statistical MT system can be constructed.

However, structural divergences between languages (Dorr, 1994)，which are due to either systematic differences between languages or loose translations in real corpora，pose a major challenge to syntax-based statistical MT. As a result, the syntax based MT systems have to transduce between non-isomorphic tree structures.

(Wu, 1997) introduced a polynomial-time solution for the alignment problem based on synchronous binary trees. (Alshawi et al., 2000) represents each production in parallel dependency trees as a finite-state transducer. Both approaches learn the tree representations directly from parallel sentences, and do not make allowances for non-isomorphic structures. (Yamada and Knight, 2001, 2002) modeled translation as a sequence of tree operations transforming a syntactic tree into a string of the target language.

When researchers try to use syntax trees in both languages, the problem of non-isomorphism must be addressed. In theory, stochastic tree transducers and some versions of synchronous grammars provide solutions for the non-isomorphic tree based transduction problem and hence possible solutions for MT. Synchronous Tree Adjoining Grammars, proposed by (Shieber and Schabes, 1990), were introduced primarily for semantics but were later also proposed for translation. Eisner (2003) proposed viewing the MT problem as a probabilistic synchronous tree substitution grammar parsing

Proceedings of the 43rd Annual Meeting of the ACL, pages 541–548,
Ann Arbor, June 2005.

problem. Melamed (2003, 2004) formalized the MT problem as synchronous parsing based on multitext grammars. Graehl and Knight (2004) defined training and decoding algorithms for both generalized tree-to-tree and tree-to-string transducers. All these approaches, though different in formalism, model the two languages using tree-based transduction rules or a synchronous grammar, possibly probabilistic, and using multi-lemma elementary structures as atomic units. The machine translation is done either as a stochastic tree-to-tree transduction or a synchronous parsing process.

However, few of the above mentioned formalisms have large scale implementations. And to the best of our knowledge, the advantages of syntax based statistical MT systems over pure statistical MT systems have yet to be empirically verified.

We believe difficulties in inducing a synchronous grammar or a set of tree transduction rules from large scale parallel corpora are caused by:

1. The abilities of synchronous grammars and tree transducers to handle non-isomorphism are limited. At some level, a synchronous derivation process must exist between the source and target language sentences.
2. The training and/or induction of a synchronous grammar or a set of transduction rules are usually computationally expensive if all the possible operations and elementary structures are allowed. The exhaustive search for all the possible sub-sentential structures in a syntax tree of a sentence is NP-complete.
3. The problem is aggravated by the non-perfect training corpora. Loose translations are less of a problem for string based approaches than for approaches that require syntactic analysis.

Hajic et al. (2002) limited non-isomorphism by n-to-m matching of nodes in the two trees. However, even after extending this model by allowing cloning operations on subtrees, Gildea (2003) found that parallel trees over-constrained the alignment problem, and achieved better results with a tree-to-string model than with a tree-to-tree model using two trees. In a different approach, Hwa et al. (2002) aligned the parallel sentences using phrase based statistical MT models and then projected the alignments back to the parse trees.

This motivated us to look for a more efficient and effective way to induce a synchronous grammar from parallel corpora and to build an MT system that performs competitively with the pure statistical MT systems. We chose to build the synchronous grammar on the parallel dependency structures of the sentences. The synchronous grammar is induced by hierarchical tree partitioning operations. The rest of this paper describes the system details as follows: Sections 2 and 3 describe the motivation behind the usage of dependency structures and how a version of synchronous dependency grammar is learned. This grammar is used as the primary translation knowledge source for our system. Section 4 defines the tree-to-tree transducer and the graphical model for the stochastic tree-to-tree transduction process and introduces a polynomial time decoding algorithm for the transducer. We evaluate our system in section 5 with the NIST/Bleu automatic MT evaluation software and the results are discussed in Section 6.

2 The Synchronous Grammar

2.1 Why Dependency Structures?

According to Fox (2002), dependency representations have the best inter-lingual phrasal cohesion properties. The percentage for head crossings is 12.62% and that of modifier crossings is 9.22%. Furthermore, a grammar based on dependency structures has the advantage of being simple in formalism yet having CFG equivalent formal generative capacity (Ding and Palmer, 2004b).

Dependency structures are inherently lexicalized as each node is one word. In comparison, phrasal structures (treebank style trees) have two node types: terminals store the lexical items and non-terminals store word order and phrasal scopes.

2.2 Synchronous Dependency Insertion Grammars

Ding and Palmer (2004b) described one version of synchronous grammar: Synchronous Dependency Insertion Grammars. A Dependency Insertion Grammars (DIG) is a generative grammar formalism that captures word order phenomena within the dependency representation. In the scenario of two languages, the two sentences in the source and target languages can be modeled as being generated from a synchronous derivation process.

A synchronous derivation process for the two syntactic structures of both languages suggests the level of cross-lingual isomorphism between the two trees (e.g. Synchronous Tree Adjoining Grammars (Shieber and Schabes, 1990)).

Apart from other details, a DIG can be viewed as a tree substitution grammar defined on dependency trees (as opposed to phrasal structure trees). The basic units of the grammar are elementary trees (ET), which are sub-sentential dependency structures containing one or more lexical items. The synchronous version, SDIG, assumes that the isomorphism of the two syntactic structures is at the ET level, rather than at the word level, hence allowing non-isomorphic tree to tree mapping.

We illustrate how the SDIG works using the following pseudo-translation example:

- [*Source*] *The girl kissed her kitty cat.*
- [*Target*] *The girl gave a kiss to her cat.*

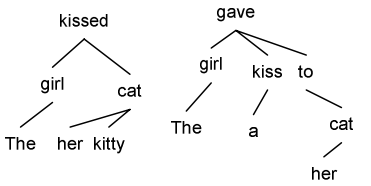


Figure 1. An example

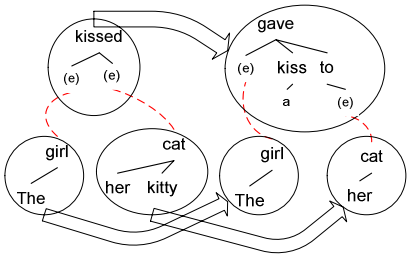


Figure 2. Tree-to-tree transduction

Almost any tree-transduction operations defined on a single node will fail to generate the target sentence from the source sentence without using insertion/deletion operations. However, if we view each dependency tree as an assembly of indivisible sub-sentential elementary trees (ETs), we can find a proper way to transduce the input tree to the output tree. An ET is a single "symbol" in a transducer's language. As shown in Figure 2, each circle stands for an ET and thick arrows denote the transduction of each ET as a single symbol.

3 Inducing a Synchronous Dependency Insertion Grammar

As the start to our syntax-based SMT system, the SDIG must be learned from the parallel corpora.

3.1 Cross-lingual Dependency Inconsistencies

One straightforward way to induce a generative grammar is using EM style estimation on the generative process. Different versions of such training algorithms can be found in (Hajic et al., 2002; Eisner 2003; Gildea 2003; Graehl and Knight 2004).

However, a synchronous derivation process cannot handle two types of cross-language mappings: crossing-dependencies (parent-descendent switch) and broken dependencies (descendent appears elsewhere), which are illustrated below:

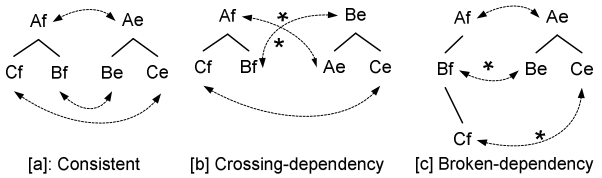


Figure 3. Cross-lingual dependency consistencies

In the above graph, the two sides are English and the foreign dependency trees. Each node in a tree stands for a lemma in a dependency tree. The arrows denote aligned nodes and those resulting inconsistent dependencies are marked with a "*".

Fox (2002) collected the statistics mainly on French and English data: in dependency representations, the percentage of head crossings per chance (case [b] in the graph) is 12.62%.

Using the statistics on cross-lingual dependency consistencies from a small word to word aligned Chinese-English parallel corpus[1], we found that the percentage of crossing-dependencies (case [b]) between Chinese and English is 4.7% while that of broken dependencies (case [c]) is 59.3%.

The large number of broken dependencies presents a major challenge for grammar induction based on a top-down style EM learning process.

Such broken and crossing dependencies can be modeled by SDIG if they appear inside a pair of elementary trees. However, if they appear between the elementary trees, they are not compatible with the isomorphism assumption on which SDIG is based. Nevertheless, the hope is that the fact that the training corpus contains a significant percentage of dependency inconsistencies does not mean that during decoding the target language sentence cannot be written in a dependency consistent way.

3.2 Grammar Induction by Synchronous Hierarchical Tree Partitioning

(Ding and Palmer, 2004a) gave a polynomial time solution for learning parallel sub-sentential de-

[1] Total 826 sentence pairs, 9957 Chinese words, 12660 English words. Data made available by the courtesy of Microsoft Research, Asia and IBM T.J. Watson Research.

pendency structures from non-isomorphic dependency trees. Our approach, while similar to (Ding and Palmer, 2004a) in that we also iteratively partition the parallel dependency trees based on a heuristic function, departs (Ding and Palmer, 2004a) in three ways: (1) we base the hierarchical tree partitioning operations on the categories of the dependency trees; (2) the statistics of the resultant tree pairs from the partitioning operation are collected at each iteration rather than at the end of the algorithm; (3) we do not re-train the word to word probabilities at each iteration. Our grammar induction algorithm is sketched below:

Step 0. View each tree as a "bag of words" and train a statistical translation model on all the tree pairs to acquire word-to-word translation probabilities. In our implementation, the IBM Model 1 (Brown et al., 1993) is used.

Step 1. Let i denote the current iteration and let $C = CategorySequence[i]$ be the current syntactic category set.

For each tree pair in the corpus, do {

a) For the tentative synchronous partitioning operation, use a heuristic function to select the BEST word pair (e_{i*}, f_{j*}), where both e_{i*}, f_{j*} are NOT "chosen", $Category(e_{i*}) \in C$ and $Category(f_{j*}) \in C$.

b) If (e_{i*}, f_{j*}) is found in (a), mark e_{i*}, f_{j*} as "chosen" and go back to (a), else go to (c).

c) Execute the synchronous tree partitioning operation on all the "chosen" word pairs on the tree pair. Hence, several new tree pairs are created. Replace the old tree pair with the new tree pairs together with the rest of the old tree pair.

d) Collect the statistics for all the new tree pairs as elementary tree pairs. }

Step 2. $i = i + 1$. Go to Step 1 for the next iteration.

At each iteration, one specific set of categories of nodes is handled. The category sequence we used in the grammar induction is:

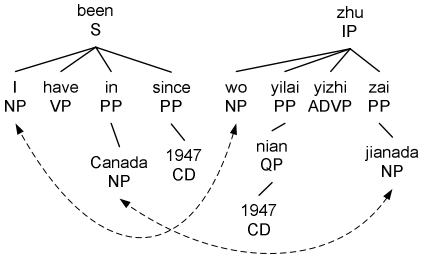
1. *Top-NP*: the noun phrases that do not have another noun phrase as parent or ancestor.
2. *NP*: all the noun phrases
3. *VP, IP, S, SBAR*: verb phrases equivalents.
4. *PP, ADJP, ADVP, JJ, RB*: all the modifiers
5. *CD*: all the numbers.

We first process top *NP* chunks because they are the most stable between languages. Interestingly, NPs are also used as anchor points to learn monolingual paraphrases (Ibrahim et al., 2003). The phrasal structure categories can be extracted from automatic parsers using methods in (Xia, 2001).

An illustration is given below (Chinese in *pinyin* form). The placement of the dependency arcs reflects the relative word order between a parent node and all its immediate children. The collected ETs are put into square boxes and the partitioning operations taken are marked with dotted arrows.

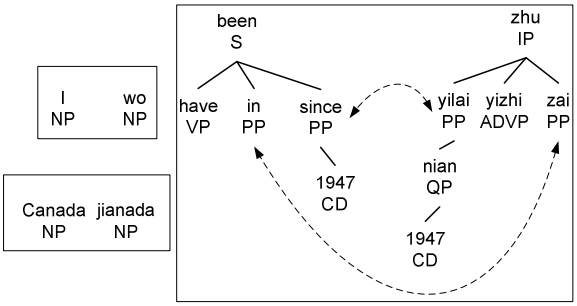
- [*English*] *I have been in Canada since 1947.*
- [*Chinese*] *Wo 1947 nian yilai yizhi zhu zai jianada.*
- [*Glossary*] *I 1947 year since always live in Canada*

[ITERATION 1 & 2] Partition at word pair ("*I*" and "*wo*") ("Canada" and "janada")

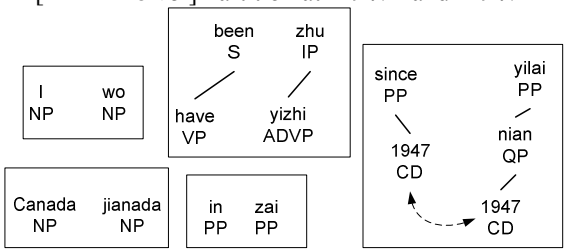


[ITERATION 3] ("*been*" and "*zhu*") are chosen but no partition operation is taken because they are roots.

[ITERATION 4] Partition at word pair ("*since*" and "*yilai*") ("*in*" and "*zai*")



[ITERATION 5] Partition at "*1947*" and "*1947*"



[FINALLY] Total of 6 resultant ET pairs (figure omitted)

Figure 4. An Example

3.3 Heuristics

Similar to (Ding and Palmer, 2004a), we also use a heuristic function in Step 1(a) of the algorithm to rank all the word pairs for the tentative tree parti-

tioning operation. The heuristic function is based on a set of heuristics, most of which are similar to those in (Ding and Palmer, 2004a).

For a word pair (e_i, f_j) for the tentative partitioning operation, we briefly describe the heuristics:

- Inside-outside probabilities: We borrow the idea from PCFG parsing. This is the probability of an English subtree (inside) generating a foreign subtree and the probability of the English residual tree (outside) generating a foreign residual tree. Here both probabilities are based on a "bag of words" model.
- Inside-outside penalties: here the probabilities of the inside English subtree generating the outside foreign residual tree and outside English residual tree generating the inside English subtree are used as penalty terms.
- Entropy: the entropy of the word to word translation probability of the English word e_i.
- Part-of-Speech mapping template: whether the POS tags of the two words are in the "highly likely to match" POS tag pairs.
- Word translation probability: $\mathrm{P}(f_j \mid e_i)$.
- Rank: the rank of the word to word probability of f_j in as a translation of e_i among all the foreign words in the current tree.

The above heuristics are a set of real valued numbers. We use a Maximum Entropy model to interpolate the heuristics in a log-linear fashion, which is different from the error minimization training in (Ding and Palmer, 2004a).

$$\mathrm{P}\left(y \mid h_0(e_i, f_j), h_1(e_i, f_j) \ldots h_n(e_i, f_j)\right) = \frac{1}{Z} \exp\left(\sum_k \lambda_k h_k(e_i, f_j) + \lambda_s\right) \quad (1)$$

where $y = (0,1)$ as labeled in the training data whether the two words are mapped with each other.

The MaxEnt model is trained using the same word level aligned parallel corpus as the one in Section 3.1. Although the training corpus isn't large, the fact that we only have a handful of parameters to fit eased the problem.

3.4 A Scaled-down SDIG

It is worth noting that the set of derived parallel dependency Elementary Trees is not a full-fledged SDIG yet. Many features in the SDIG formalism such as arguments, head percolation, etc. are not yet filled. We nevertheless use this derived grammar as a Mini-SDIG, assuming the unfilled features as empty by default. A full-fledged SDIG remains a goal for future research.

4 The Machine Translation System

4.1 System Architecture

As discussed before (see Figure 1 and 2), the architecture of our syntax based statistical MT system is illustrated in Figure 5. Note that this is a non-deterministic process. The input sentence is first parsed using an automatic parser and a dependency tree is derived. The rest of the pipeline can be viewed as a stochastic tree transducer. The MT decoding starts first by decomposing the input dependency tree in to elementary trees. Several different results of the decomposition are possible. Each decomposition is indeed a derivation process on the foreign side of SDIG. Then the elementary trees go through a transfer phase and target ETs are combined together into the output.

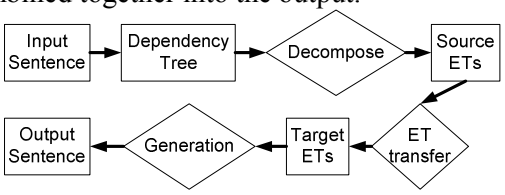


Figure 5. System architecture

4.2 The Graphical Model

The stochastic tree-to-tree transducer we propose models MT as a probabilistic optimization process.

Let f be the input sentence (foreign language), and e be the output sentence (English). We have $\mathrm{P}(e \mid f) = \frac{\mathrm{P}(f \mid e)\mathrm{P}(e)}{\mathrm{P}(f)}$, and the best translation is:

$$e^* = \arg\max_e \mathrm{P}(f \mid e)\mathrm{P}(e) \quad (2)$$

$\mathrm{P}(f \mid e)$ and $\mathrm{P}(e)$ are also known as the "translation model" (TM) and the "language model" (LM). Assuming the decomposition of the foreign tree is given, our approach, which is based on ETs, uses the graphical model shown in Figure 6.

In the model, the left side is the input dependency tree (foreign language) and the right side is the output dependency tree (English). Each circle stands for an ET. The solid lines denote the syntactical dependencies while the dashed arrows denote the statistical dependencies.

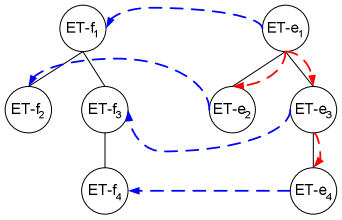


Figure 6 The graphical model

Let $\mathrm{T}(x)$ be the dependency tree constructed from sentence x. A tree-decomposition function $\mathrm{D}(t)$ is defined on a dependency tree t, and outputs a certain ET derivation tree of t, which is generated by decomposing t into ETs. Given t, there could be multiple decompositions. Conditioned on decomposition D, we can rewrite (2) as:

$$\begin{aligned} e^* &= \arg\max_e \sum_D \mathrm{P}(f, e \mid D)\mathrm{P}(D) \\ &= \arg\max_e \sum_D \mathrm{P}(f \mid e, D)\mathrm{P}(e \mid D)\mathrm{P}(D) \end{aligned} \quad (3)$$

By definition, the ET derivation trees of the input and output trees should be isomorphic: $\mathrm{D}(\mathrm{T}(f)) \cong \mathrm{D}(\mathrm{T}(e))$. Let $\mathrm{Tran}(u)$ be a set of possible translations for the ET u. We have:

$$\begin{aligned} \mathrm{P}(f \mid e, D) &= \mathrm{P}(\mathrm{T}(f) \mid \mathrm{P}(\mathrm{T}(e), D) \\ &= \prod_{u \in \mathrm{D}(\mathrm{T}(f)),\ v \in \mathrm{D}(\mathrm{T}(e)),\ v \in \mathrm{Tran}(u)} \mathrm{P}(u \mid v) \end{aligned} \quad (4)$$

For any ET v in a given ET derivation tree d, let $\mathrm{Root}(d)$ be the root ET of d, and let $\mathrm{Parent}(v)$ denote the parent ET of v. We have:

$$\begin{aligned} \mathrm{P}(e \mid D) &= \mathrm{P}(\mathrm{T}(e) \mid D) \\ &= \mathrm{P}\left(\mathrm{Root}\left(\mathrm{D}(\mathrm{T}(e))\right)\right) \cdot \\ &\cdot \left(\prod_{v \in \mathrm{D}(\mathrm{T}(e)), v \neq \mathrm{Root}(\mathrm{D}(\mathrm{T}(e)))} \mathrm{P}(v \mid \mathrm{Parent}(v)) \right) \end{aligned} \quad (5)$$

where, letting $\mathrm{root}(v)$ denote the root word of v,

$$\mathrm{P}\left(v \mid \mathrm{Parent}(v)\right) = \mathrm{P}\left(\mathrm{root}(v) \mid \mathrm{root}\left(\mathrm{Parent}(v)\right)\right) \quad (6)$$

The prior probability of tree decomposition is defined as: $\mathrm{P}\left(\mathrm{D}(\mathrm{T}(f))\right) = \prod_{u \in \mathrm{D}(\mathrm{T}(f))} \mathrm{P}(u)$ (7)

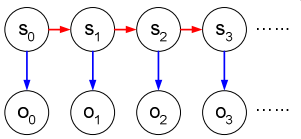


Figure 7 Comparing to the HMM

An analogy between our model and a Hidden Markov Model (Figure 7) may be helpful. In Eq. (4), $\mathrm{P}(u \mid v)$ is analogous to the emission probably $\mathrm{P}(o_i \mid s_i)$ in an HMM. In Eq. (5), $\mathrm{P}(v \mid \mathrm{Parent}(v))$ is analogous to the transition probability $\mathrm{P}(s_i \mid s_{i-1})$ in an HMM. While HMM is defined on a sequence our model is defined on the derivation tree of ETs.

4.3 Other Factors

- Augmenting parallel ET pairs

In reality, the learned parallel ETs are unlikely to cover all the structures that we may encounter in decoding. As a unified approach, we augment the SDIG by adding all the possible word pairs (f_j, e_i) as a parallel ET pair and using the IBM Model 1 (Brown et al., 1993) word to word translation probability as the ET translation probability.

- Smoothing the ET translation probabilities.

The LM probabilities $\mathrm{P}(v \mid \mathrm{Parent}(v))$ are simply estimated using the relative frequencies. In order to handle possible noise from the ET pair learning process, the ET translation probabilities $\mathrm{P}_{emp}(u \mid v)$ estimated by relative frequencies are smoothed using a word level model. For each ET pair (u, v), we interpolate the empirical probability with the "bag of words" probability and then re-normalize:

$$\bar{\mathrm{P}}(u \mid v) = \frac{1}{Z}\mathrm{P}_{emp}(u, v) \cdot \frac{1}{\mathrm{size}(u)^{\mathrm{size}(v)}} \prod_{f_j \in u} \sum_{e_i \in v} \mathrm{P}(f_j \mid e_i) \quad (8)$$

4.4 Polynomial Time Decoding

For efficiency reasons, we use maximum approximation for (3). Instead of summing over all the possible decompositions, we only search for the best decomposition as follows:

$$e^*, D^* = \arg\max_{e,D} \mathrm{P}(f \mid e, D)\mathrm{P}(e \mid D)\mathrm{P}(D) \quad (9)$$

So bringing equations (4) to (9) together, the best translation would maximize:

$$\prod \bar{\mathrm{P}}(u \mid v) \cdot \mathrm{P}\left(\mathrm{Root}(e)\right) \cdot \left(\prod \mathrm{P}(v \mid \mathrm{Parent}(v)) \right) \cdot \prod \mathrm{P}(u) \quad (10)$$

Observing the similarity between our model and a HMM, our dynamic programming decoding algorithm is in spirit similar to the *Viterbi* algorithm except that instead of being sequential the decoding is done on trees in a top down fashion.

As to the relative orders of the ETs, we currently choose not to reorder the children ETs given the parent ET because: (1) the permutation of the ETs is computationally expensive (2) it is possible that we can resort to simple linguistic treatments on the output dependency tree to order the ETs.

Currently, all the ETs are attached to each other

at their root nodes.

In our implementation, the different decompositions of the input dependency tree are stored in a shared forest structure, utilizing the dynamic programming property of the tree structures explicitly.

Suppose the input sentence has n words and the shared forest representation has m nodes. Suppose for each word, there are maximally k different ETs containing it, we have $m \le kn$. Let b be the max breadth factor in the packed forest, it can be shown that the decoder visits at most mb nodes during execution. Hence, we have:

$$T(decoding) \le O(kbn) \quad (11)$$

which is linear to the input size. Combined with a polynomial time parsing algorithm, the whole decoding process is polynomial time.

5 Evaluation

We implemented the above approach for a Chinese-English machine translation system. We used an automatic syntactic parser (Bikel, 2002) to produce the parallel parse trees. The parser was trained using the Penn English/Chinese Treebanks. We then used the algorithm in (Xia 2001) to convert the phrasal structure trees to dependency trees to acquire the parallel dependency trees. The statistics of the datasets we used are shown as follows:

Dataset	Xinhua	FBIS	NIST
Sentence#	56263	45212	206
Chinese word#	1456495	1185297	27.4 average
English word#	1490498	1611932	37.7 average
Usage	training	training	testing

Figure 8. Evaluation data details

The training set consists of Xinhua newswire data from LDC and the FBIS data (mostly news), both filtered to ensure parallel sentence pair quality. We used the development test data from the 2001 NIST MT evaluation workshop as our test data for the MT system performance. In the testing data, each input Chinese sentence has 4 English translations as references. Our MT system was evaluated using the *n*-gram based Bleu (Papineni et al., 2002) and NIST machine translation evaluation software. We used the NIST software package "mteval" version 11a, configured as case-insensitive.

In comparison, we deployed the GIZA++ MT modeling tool kit, which is an implementation of the IBM Models 1 to 4 (Brown et al., 1993; Al-Onaizan et al., 1999; Och and Ney, 2003). The IBM models were trained on the same training data as our system. We used the ISI Rewrite decoder (Germann et al. 2001) to decode the IBM models.

The results are shown in Figure 9. The score types "I" and "C" stand for individual and cumulative *n*-gram scores. The final NIST and Bleu scores are marked with bold fonts.

Systems	Score Type		1-gram	2-gram	3-gram	4-gram
IBM Model 4	I	NIST	2.562	0.412	0.051	0.008
		Bleu	0.714	0.267	0.099	0.040
	C	NIST	2.562	2.974	3.025	**3.034**
		Bleu	0.470	0.287	0.175	**0.109**
SDIG	I	NIST	5.130	0.763	0.082	0.013
		Bleu	0.688	0.224	0.075	0.029
	C	NIST	5.130	5.892	5.978	**5.987**
		Bleu	0.674	0.384	0.221	**0.132**

Figure 9. Evaluation Results.

The evaluation results show that the NIST score achieved a 97.3% increase, while the Bleu score increased by 21.1%.

In terms of decoding speed, the Rewrite decoder took 8102 seconds to decode the test sentences on a Xeon 1.2GHz 2GB memory machine. On the same machine, the SDIG decoder took 3 seconds to decode, excluding the parsing time. The recent advances in parsing have achieved parsers with $O(n^3)$ time complexity without the grammar constant (McDonald et al., 2005). It can be expected that the total decoding time for SDIG can be as short as 0.1 second per sentence.

Neither of the two systems has any specific translation components, which are usually present in real world systems (E.g. components that translate numbers, dates, names, etc.) It is reasonable to expect that the performance of SDIG can be further improved with such specific optimizations.

6 Discussions

We noticed that the SDIG system outputs tend to be longer than those of the IBM Model 4 system, and are closer to human translations in length.

Translation Type	Human	SDIG	IBM-4
Avg. Sent. Len.	37.7	33.6	24.2

Figure 10. Average Sentence Word Count

This partly explains why the IBM Model 4 system has slightly higher individual *n*-gram precision scores (while the SDIG system outputs are still better in terms of absolute matches).

The relative orders between the parent and child ETs in the output tree is currently kept the same as the orders in the input tree. Admittedly, we benefited from the fact that both Chinese and English are SVO languages, and that many of orderings between the arguments and adjuncts can be kept the same. However, we did notice that this simple "ostrich" treatment caused outputs such as "*foreign financial institutions the president of*".

While statistical modeling of children reordering is one possible remedy for this problem, we believe simple linguistic treatment is another, as the output of the SDIG system is an English dependency tree rather than a string of words.

7 Conclusions and Future Work

In this paper we presented a syntax-based statistical MT system based on a Synchronous Dependency Insertion Grammar and a non-isomorphic stochastic tree-to-tree transducer. A graphical model for the transducer is defined and a polynomial time decoding algorithm is introduced. The results of our current implementation were evaluated using the NIST and Bleu automatic MT evaluation software. The evaluation shows that the SDIG system outperforms an IBM Model 4 based system in both speed and quality.

Future work includes a full-fledged version of SDIG and a more sophisticated MT pipeline with possibly a *tri*-gram language model for decoding.

References

Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, I. D. Melamed, F. Och, D. Purdy, N. A. Smith, and D. Yarowsky. 1999. Statistical machine translation. Technical report, CLSP, Johns Hopkins University.

H. Alshawi, S. Bangalore, S. Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Comp. Linguistics*, 26(1):45-60.

Daniel M. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *HLT 2002*.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2): 263-311.

Michael John Collins. 1999. Head-driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Ding and Palmer. 2004a. Automatic Learning of Parallel Dependency Treelet Pairs. In First International Joint Conference on NLP (IJCNLP-04).

Ding and Palmer. 2004b. Synchronous Dependency Insertion Grammars: A Grammar Formalism for Syntax Based Statistical MT. Workshop on Recent Advances in Dependency Grammars, COLING-04.

Bonnie J. Dorr. 1994. Machine translation divergences: A formal description and proposed solution. *Computational Linguistics*, 20(4): 597-633.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In ACL-03. (companion volume), Sapporo, July.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of EMNLP-02*.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast Decoding and Optimal Decoding for Machine Translation. ACL-01.

Daniel Gildea. 2003. Loosely tree based alignment for machine translation. ACL-03, Japan.

Jonathan Graehl and Kevin Knight. 2004. Training Tree Transducers. In NAACL/HLT-2004

Jan Hajic, et al. 2002. Natural language generation in the context of machine translation. Summer workshop final report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore.

Rebecca Hwa, Philip S. Resnik, Amy Weinberg, and Okan Kolak. 2002. Evaluating translational correspondence using annotation projection. ACL-02

Ali Ibrahim, Boris Katz, and Jimmy Lin. 2003. Extracting Structural Paraphrases from Aligned Monolingual Corpora. In *Proceedings of the Second International Workshop on Paraphrasing* (IWP 2003)

Dan Melamed. 2004. Statistical Machine Translation by Parsing. In ACL-04, Barcelona, Spain.

Dan Melamed. 2003. Multitext Grammars and Synchronous Parsers, In NAACL/HLT-2003.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. ACL-02, Philadelphia, USA.

Ryan McDonald, Koby Crammer and Fernando Pereira. 2005. Online Large-Margin Training of Dependency Parsers. ACL-05.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

S. M. Shieber and Y. Schabes. 1990. *Synchronous Tree-Adjoining Grammars*, Proceedings of the 13th COLING, pp. 253-258, August 1990.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):3-403.

Fei Xia. 2001. Automatic grammar generation from two different perspectives. PhD thesis, U. of Pennsylvania.

Kenji Yamada and Kevin Knight. 2001. A syntax based statistical translation model. ACL-01, France.

Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical MT. ACL-02, Philadelphia.