Learning to Transform and Select Elementary Trees for Improved Syntax-based Machine Translations

Bing Zhao[†], and **Young-Suk Lee**[†], and **Xiaoqiang Luo**[†], and **Liu Li**[‡]
IBM T.J. Watson Research[†] and Carnegie Mellon University[‡]
{zhaob, ysuklee, xiaoluo}@us.ibm.com and liul@andrew.cmu.edu

Abstract

We propose a novel technique of learning how to transform the source parse trees to improve the translation qualities of syntax-based translation models using synchronous context-free grammars. We transform the source tree phrasal structure into a set of simpler structures, expose such decisions to the decoding process, and find the least expensive transformation operation to better model word reordering. In particular, we integrate synchronous binarizations, verb regrouping, removal of redundant parse nodes, and incorporate a few important features such as translation boundaries. We learn the structural preferences from the data in a generative framework. The syntax-based translation system integrating the proposed techniques outperforms the best Arabic-English unconstrained system in NIST-08 evaluations by 1.3 absolute BLEU, which is statistically significant.

1 Introduction

Most syntax-based machine translation models with synchronous context free grammar (SCFG) have been relying on the off-the-shelf monolingual parse structures to learn the translation equivalences for string-to-tree, tree-to-string or tree-to-tree grammars. However, state-of-the-art monolingual parsers are not necessarily well suited for machine translation in terms of both labels and chunks/brackets. For instance, in Arabic-to-English translation, we find only 45.5% of Arabic NP-SBJ structures are mapped to the English NP-SBJ with machine alignment and parse trees, and only 60.1% of NP-SBJs are mapped with human alignment and parse trees as in § 2. The chunking is of more concern; at best only 57.4% source chunking decisions are translated contiguously on the target side. To translate the rest of the chunks one has to frequently *break* the original structures. The main issue lies in the strong assumption behind SCFG-style nonterminals – each nonterminal (or variable) assumes a source chunk should be rewritten into a *contiguous* chunk in the target. Without integrating techniques to modify the parse structures, the SCFGs are not to be effective even for translating NP-SBJ in linguistically distant language-pairs such as Arabic-English.

Such problems have been noted in previous literature. Zollmann and Venugopal (2006) and Marcu et al. (2006) used broken syntactic fragments to augment their grammars to increase the rule coverage; while we learn optimal tree fragments transformed from the original ones via a generative framework, they enumerate the fragments available from the original trees without learning process. Mi and Huang (2008) introduced parse forests to blur the chunking decisions to a certain degree, to expand search space and reduce parsing errors from 1-best trees (Mi et al., 2008); others tried to use the parse trees as soft constraints on top of unlabeled grammar such as Hiero (Marton and Resnik, 2008; Chiang, 2010; Huang et al., 2010; Shen et al., 2010) without sufficiently leveraging rich tree context. Recent works tried more complex approaches to integrate both parsing and decoding in one *single* search space as in (Liu and Liu, 2010), at the cost of huge search space. In (Zhang et al., 2009), combinations of tree forest and *tree-sequence* (Zhang et al., 2008) based approaches were carried out by adding pseudo nodes and hyper edges into the forest. Overall, the forest-based translation can reduce the risks from upstream parsing errors and expand the search space, but it cannot sufficiently address the syntactic divergences between various language-pairs. The tree sequence approach adds pseudo nodes and hyper edges to the forest, which makes the forest even denser and harder for navigation and search. As trees thrive in the search space, especially with the pseudo nodes and edges being added to the already dense forest, it is becoming harder to wade through the deep forest for the best derivation path out.

We propose to simplify suitable subtrees to a reasonable level, at which the correct reordering can be easily identified. The transformed structure should be frequent enough to have rich statistics for learning a model. Instead of creating pseudo nodes and edges and make the forest dense, we transform a tree with a few simple operators; only meaningful frontier nodes, context nodes and edges are kept to induce the correct reordering; such operations also enable the model to share the statistics among all similar subtrees.

On the basis of our study on investigating the language divergence between Arabic-English with human aligned and parsed data, we integrate several simple statistical operations, to transform parse trees adaptively to serve the translation purpose better. For each source span in the given sentence, a subgraph, corresponding to an elementary tree (in Eqn. 1), is proposed for PSCFG translation; we apply a few operators to transform the subgraph into some frequent subgraphs seen in the whole training data, and thus introduce alternative similar translational equivalences to explain the same source span with enriched statistics and features. For instance, if we regroup two adjacent nodes IV and NP-SBJ in the tree, we can obtain the correct reordering pattern for verb-subject order, which is not easily available otherwise. By finding a set of similar elementary trees derived from the original elementary trees, statistics can be shared for robust learning.

We also investigate the features using the context beyond the phrasal subtree. This is to further disambiguate the transformed subgraphs so that informative neighboring nodes and edges can influence the reordering preferences for each of the transformed trees. For instance, at the beginning and end of a sentence, we do not expect dramatic long distance reordering to happen; or under SBAR context, the clause may prefer monotonic reordering for verb and subject. Such boundary features were treated as hard constraints in previous literature in terms of re-labeling (Huang and Knight, 2006) or re-structuring (Wang et al., 2010). The boundary cases were not addressed in the previous literature for trees, and here we include them in our feature sets for learning a MaxEnt model to predict the transformations. We integrate the neighboring context of the subgraph in our transformation preference predictions, and this improve translation qualities further.

The rest of the paper is organized as follows: in section 2, we analyze the projectable structures using human aligned and parsed data, to identify the problems for SCFG in general; in section 3, our proposed approach is explained in detail, including the statistical operators using a MaxEnt model; in section 4, we illustrate the integration of the proposed approach in our decoder; in section 5, we present experimental results; in section 6, we conclude with discussions and future work.

2 The Projectable Structures

A context-free style nonterminal in PSCFG rules means the source span governed by the nonterminal should be translated into a contiguous target chunk. A "projectable" phrase-structure means that it is translated into a contiguous span on the target side, and thus can be generalized into a nonterminal in our PSCFG rule. We carried out a controlled study on the projectable structures using human annotated parse trees and word alignment for 5k Arabic-English sentence-pairs.

In Table 1, the unlabeled F-measures with machine alignment and parse trees show that, for only 48.71% of the time, the boundaries introduced by the source parses

Alignment	Parse	Labels	Accuracy
H	H	NP-SBJ	0.6011
		PP	0.3436
		NP	0.4832
		unlabel	0.5739
M	H	NP-SBJ	0.5356
		PP	0.2765
		NP	0.3959
		unlabel	0.5305
M	M	NP-SBJ	0.4555
		PP	0.1935
		NP	0.3556
		unlabel	0.4871

Table 1: The labeled and unlabeled F-measures for projecting the source nodes onto the target side via alignments and parse trees; unlabeled F-measures show the bracketing accuracies for translating a source span contiguously. H: human, M: machine.

are real translation boundaries that can be explained by a nonterminal in PSCFG rule. Even for human parse and alignment, the unlabeled F-measures are still as low as 57.39%. Such statistics show that we should not blindly learn tree-to-string grammar; additional transformations to manipulate the bracketing boundaries and labels accordingly have to be implemented to guarantee the reliability of source-tree based syntax translation grammars. The transformations could be as simple as merging two adjacent nonterminals into one bracket to accommodate non-contiguity on the target side, or lexicalizing those words which have fork-style, many-to-many alignment, or unaligned content words to enable the rest of the span to be generalized into nonterminals. We illustrate several cases using the tree in Figure 1.

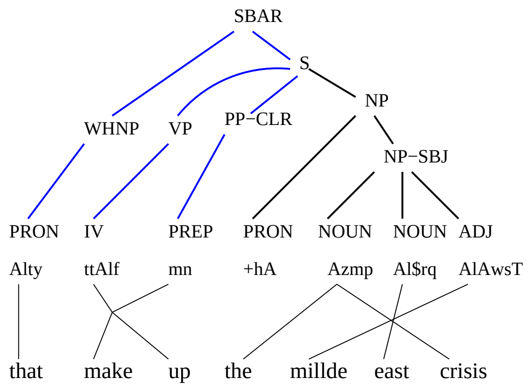

Figure 1: Non-projectable structures in an SBAR tree with human parses and alignment; there are non-projectable structures: the deleted nonterminals PRON (+hA), the many-to-many alignment for IV(ttAlf) PREP(mn), fork-style alignment for NOUN (Azmp).

In Figure 1, several non-projectable nodes were illustrated: the deleted nonterminals PRON (+hA), the many-to-many alignment for IV(ttAlf) PREP(mn), fork-style alignment for NOUN (Azmp). Intuitively, it would be good to glue the nodes NOUN(Al$rq) ADJ(AlAwsT) under the node of NP, because it is more frequent for moving ADJ before NOUN in our training data. It should be easier to model the swapping of (NOUN ADJ) using the tree (NP NOUN, ADJ) instead of the original bigger tree of (NP-SBJ Azmp, NOUN, ADJ) with one lexicalized node.

Approaches in tree-sequence based grammar (Zhang et al., 2009) tried to address the bracketing problem by using arbitrary pseudo nodes to weave a new "tree" back into the forest for further grammar extractions. Such approach may improve grammar coverage, but the pseudo node labels would be arguably a worse choice to split the already sparse data. Some of the interior nodes connecting the frontier nodes might be very informative for modeling reordering. Also, due to the introduced pseudo nodes, it would need *exponentially* many nonterminals to keep track of the matching tree-structures for translations. The created pseudo node could easily block the informative neighbor nodes associated with the subgraph which could change the reordering nature. For instance, IV and NP-SBJ tends to swap at the beginning of a sentence, but it may prefer monotone if they share a common parent of SBAR for a subclause. In this case, it is unnecessary to create a pseudo node "IV+SBJ" to block useful factors.

We propose to navigate through the forest, via simplifying trees by grouping the nodes, cutting the branches, and attaching connected neighboring informative nodes to further disambiguate the derivation path. We apply explicit translation motivated operators, on a given monolingual elementary tree, to transform it into similar but simpler trees, and expose such statistical preferences to the decoding process to select the best rewriting rule from the *enriched* grammar rule sets, for generating target strings.

3 Elementary Trees to String Grammar

We propose to use variations of an elementary tree, which is a connected subgraph fitted in the original monolingual parse tree. The subgraph is connected so that the frontiers (two or more) are connected by their *immediate common parent*. Let γ be a source elementary tree:

$$\gamma =< \ell; v^f, v^i, E >, \tag{1}$$

where v^f is a set of *frontier nodes* which contain nonterminals or words; v^i are the *interior nodes* with source labels/symbols; E is the set of *edges* connecting the nodes $v = v^f + v^i$ into a connected *subgraph* fitted in the source parse tree; ℓ is the *immediate common parent* of the frontier nodes v^f. Our proposed grammar rule is formulated as follows:

$$< \gamma; \alpha; \sim; \bar{m}; \bar{t} >, \tag{2}$$

where α is the target string, containing the terminals and/or nonterminals in a target language; $\sim$ is the one-to-one alignment of the nonterminals between γ and α; $\bar{t}$ contains possible sequence of transform operations (to be explained later in this section) associated with each rule; $\bar{m}$ is a function of enumerating the neighborhood of the source elementary tree γ, and certain tree context (nodes and edges) can be used to further disambiguate the reordering or the given lexical choices. The interior nodes of $\gamma.v^i$, however, are not necessarily informative for the reordering decisions, like the unary nodes WHNP,VP, and PP-CLR in Figure 1; while the frontier nodes $\gamma.v^f$ are the ones directly executing the reordering decisions. We can selectively cut off the interior nodes, which have no or only weak causal relations to the reordering decisions. This will enable the frequency or derived probabilities for executing the reordering to be more focused. We call such *transformation operators* $\bar{t}$. We specified a few operators for transforming an elementary tree γ, including flattening tree operators such as removing interior nodes in v^i, or grouping the children via binarizations.

Let's use the trigram "Alty ttAlf mn" in Figure 1 as an example, the immediate common parent for the span is SBAR: $\gamma.\ell$ = SBAR; the interior nodes are $\gamma.v^i = \{\text{WHNP VP S PP-CLR}\}$; the frontier nodes are $\gamma.v^f$ = (x:PRON x:IV x:PREP). The edges $\gamma.E$ (as highlighted in Figure 1) connect $\gamma.v^i$ and $\gamma.v^f$ into a subgraph for the given source ngram.

For any source span, we look up one elementary tree γ covering the span, then we select an operator $\bar{t} \in T$, to explore a set of similar elementary trees $\bar{t}(\gamma, \bar{m}) = \{\gamma'\}$ as simplified alternatives for translating that source tree (span) γ into an optimal target string α^* accordingly. Our generative model is summarized in Eqn. 3:

$$\begin{aligned}\alpha^* = \operatorname*{arg\,max}_{\bar{t}\in T;\gamma'\in\bar{t}(\gamma,\bar{m})} \quad & p_a(\alpha'|\gamma')\times \\ & p_b(\gamma'|\bar{t},\gamma,\bar{m})\times \\ & p_c(\bar{t}|\gamma,\bar{m}).\end{aligned} \tag{3}$$

In our generative scheme, for a given elementary tree γ, we sample an operator (or a combination of operations) $\bar{t}$ with the probability of $p_c(\bar{t}|\gamma)$; with operation $\bar{t}$, we transform γ into a set of simplified versions $\gamma' \in \bar{t}(\gamma, \bar{m})$ with the probability of $p_b(\gamma'|\bar{t}, \gamma)$; finally we select the transformed version γ' to generate the target string α' with a probability of $p_a(\alpha'|\gamma')$. Note here, γ' and γ share the same immediate common parent ℓ, but not necessarily the frontier, or interior, or even neighbors. The frontier nodes can be merged, lexicalized, or even deleted in the tree-to-string rule associated with γ', as long as the alignment for the nonterminals are book-kept in the derivations. To simplify the model, one can choose the operator $\bar{t}$ to be only one level, and the model using a single operator $\bar{t}$ is to be deterministic. Thus, the final set of models to learn are $p_a(\alpha'|\gamma')$ for rule alignment, and the preference model $p_b(\gamma'|\bar{t},\gamma,\bar{m})$, and the operator proposal model $p_c(\bar{t}|\gamma,\bar{m})$, which in our case is a maximum entropy model— the key model in our proposed approach in this paper for transforming the original elementary tree into similar trees for evaluating the reordering probabilities.

Eqn. 3 significantly enriches reordering powers for syntax-based machine translation. This is because it uses all *similar* set of elementary trees to generate the best target strings. In the next section, we'll first define the operators conceptually, and then explain how we learn each of the models.

3.1 Model $p_a(\alpha'|\gamma')$

A log linear model is applied here to approximate $p_a(\alpha'|\gamma') \propto \exp(\bar{\lambda}\cdot\overline{ff})$ via weighted combination ($\bar{\lambda}$) of feature functions $\overline{ff}(\alpha',\gamma')$, including relative frequencies in both directions, and IBM Model-1 scores in both directions as γ' and α' have lexical items within them. We also employed a few binary features listed in the following table.

γ' is observed less than 2 times
(α',γ') deletes a src content word
(α',γ') deletes a src function word
(α',γ') over generates a tgt content word
(α',γ') over generates a tgt function word

Table 2: Additional 5 Binary Features for $p_a(\alpha'|\gamma')$

3.2 Model $p_b(\gamma'|\bar{t},\gamma,\bar{m})$

$p_b(\gamma'|\bar{t},\gamma,\bar{m})$ is our preference model. For instance using the operator $\bar{t}$ of cutting an unary interior node in $\gamma.v^i$, if $\gamma.v^i$ has more than one unary interior node, like the SBAR tree in Figure 1, having three unary interior node: WHNP, VP and PP-CLR, $p_b(\gamma'|\bar{t},\gamma,\bar{m})$ specifies which one should have more probabilities to be cut. In our case, to make model simple, we simply choose histogram/frequency for modeling the choices.

3.3 Model $p_c(\bar{t}|\gamma,\bar{m})$

$p_c(\bar{t}|\gamma,\bar{m})$ is our operator proposal model. It ranks the operators which are valid to be applied for the given source tree γ together with its neighborhood $\bar{m}$. Here, in our approach, we applied a Maximum Entropy model, which is also employed to train our Arabic parser: $p_c(\bar{t}|\gamma,\bar{m}) \propto \exp\bar{\lambda}\cdot\overline{ff}(\bar{t},\gamma,\bar{m})$. The feature sets we use here are almost the same set we used to train our Arabic parser; the only difference is the future space here is operator categories, and we check bag-of-nodes for interior nodes and frontier nodes. The key feature categories we used are listed as in the Table 3. The headtable used in our training is manually built for Arabic.

bag-of-nodes $\gamma.v^i$
bag-of-nodes and ngram of $\gamma.v^f$
chunk-level features: left-child, right-child, etc.
lexical features: unigram and bigram
pos features: unigram and bigram
contextual features: surrounding words

Table 3: Feature Features for learning $p_c(\bar{t}|\gamma,\bar{m})$

3.4 $\bar{t}$: Tree Transformation Function

Obvious systematic linguistic divergences between language-pairs could be handled by some simple operators such as using binarization to re-group contiguously aligned children. Here, we start from the human aligned and parsed data as used in section 2 to explore potential useful operators.

3.4.1 Binarizations

One of the simplest way for transforming a tree is via binarization. Monolingual binarization chooses to re-group children into smaller subtree with a suitable label for the newly created root. We choose a function mapping to select the *top-frequent* label as the root for the grouped children; if such label is not found we simply use the label of the immediate common parent for γ. In decoding time, we need to select trees from all possible binarizations, while in the training time, we restrict the choices allowed with the alignment constraint, that every grouped children should be aligned contiguously on the target side. Our goal is to simulate the synchronous binarization as much as we can. In this paper, we applied the four basic operators for binarizing a tree: left-most, right-most and additionally head-out left and head-out right for more than three children. Two examples are given in Table 4, in which we used LDC style representation for the trees.

With the proper binarization, the structure becomes rich in sub-structures which allow certain reordering to happen more likely than others. For instance for the subtree (VP PV NP-SBJ), one would apply stronger statistics from training data to support the swap of NP-SBJ and PV for translation.

3.4.2 Regrouping verbs

Verbs are keys for reordering especially for Araic-English with VSO translated into SVO. However, if the verb and its relevant arguments for reordering are at different levels in the tree, the reordering is difficult to model as more interior nodes combinations will distract the distributions and make the model less focused. We provide the following two operations specific for verb in VP trees as in Table 5.

3.4.3 Removing interior nodes and edges

For reordering patterns, keeping the deep tree structure might not be the best choice. Sometimes it is not even

Binarization Operations	Examples
right-most	(NP X_{noun} X_{adj_1} X_{adj_2}) $\mapsto$ (NP X_{noun} (ADJP X_{adj_1} X_{adj_2}))
left-most	(VP X_{pv} $X_{\text{NP-SBJ}}$ X_{SBAR}) $\mapsto$ (VP (VP X_{pv} $X_{\text{NP-SBJ}}$) X_{SBAR})

Table 4: Operators for binarizing the trees

Operators for regroup verbs	Examples
regroup verb	$(VP_1\ X_v\ (VP_2\ Y)) \mapsto (VP_1\ (VP_2\ X_v\ Y))$
regroup verb and remove the top level VP	$(R\ (VP_1\ X_v\ (R_2\ Y))) \mapsto (R\ (R_2\ X_vY))$

Table 5: Operators for manipulating the trees

possible due to the many-to-many alignment, insertions and deletions of terminals. So, we introduce the operators to remove the interior nodes $\gamma.v^i$ selectively; this way, we can flatten the tree, remove irrelevant nodes and edges, and can use more frequent observations of simplified structures to capture the reordering patterns. We use two operators as shown in Table 6.

The second operator deletes all the interior nodes, labels and edges; thus reordering will become a Hiero-alike (Chiang, 2007) *unlabeled rule*, and additionally a special glue rule: $X_1X_2 \rightarrow X_1X_2$. This operator is necessary, we need a scheme to automatically back off to the meaningful glue or Hiero-alike rules, which may lead to a cheaper derivation path for constructing a partial hypothesis, at the decoding time.

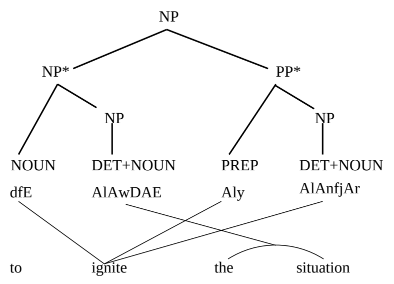

Figure 2: A NP tree with an "inside-out" alignment. The nodes "NP*" and "PP*" are not suitable for generalizing into NTs used in PSCFG rules.

As shown in Table 1, NP brackets has only 35.56% of time to be translated contiguously as an NP in machine aligned & parsed data. The NP tree in Figure 2 happens to be an "inside-out" style alignment, and context free grammar such as ITG (Wu, 1997) can not explain this structure well without necessary lexicalization. Actually, the Arabic tokens of "dfE Aly AlAnfjAr" form a combination and is turned into English word "ignite" in an idiomatic way. With lexicalization, a Hiero style rule "dfE X Aly AlAnfjAr $\mapsto$ to ignite X" is potentially a better alternative for translating the NP tree. Our operators allow us to back off to such Hiero-style rules to construct derivations, which share the immediate common parent NP, as defined for the elementary tree, for the given source span.

3.5 $\bar{m}$: Neighboring Function

For a given elementary tree, we use function $\bar{m}$ to check the context beyond the subgraph. This includes looking the nodes and edges connected to the subgraph. Similar to the features used in (Dyer et al., 2009), we check the following three cases.

3.5.1 Sentence boundaries

When the tree γ frontier sets contain the left-most token, right-most token, or both sides, we will add to the neighboring nodes the corresponding decoration tags L (left), R (right), and B (both), respectively. These decorations are important especially when the reordering patterns for the same trees are depending on the context. For instance, at the beginning or end of a sentence, we do not expect dramatic reordering – moving a token too far away in the middle of the sentences.

3.5.2 SBAR/IP/PP/FRAG boundaries

We check siblings of the root for γ for a few special labels, including SBAR, IP, PP, and FRAG. These labels indicate a partial sentence or clause, and the reordering patterns may get different distributions due to the position relative to these nodes. For instance, the PV and SBJ nodes under SBAR tends to have more monotone preference for word reordering (Carpuat et al., 2010). We mark the boundaries with position markers such as L-PP, to indicate having a left sibling PP, R-IP for having a right sibling IP, and C-SBAR to indicate the elementary tree is a child of SBAR. These labels are selected mainly based on our linguistic intuitions and errors in our translation system. A data-driven approach might be more promising for identifying useful markups w.r.t specific reordering patterns.

3.5.3 Translation boundaries

In the Figure 2, there are two special nodes under NP: NP* and PP*. These two nodes are aligned in a "inside-out" fashion, and none of them can be generalized into a nonterminal to be rewritten in a PSCFG rule. In other words, the phrasal brackets induced from NP* and PP*

operators for removing nodes/edges	Examples
remove unary nodes	$(R\ \ X_{t_1}(R_1\ \ (R2\ \ X_{t_2}))) \mapsto (R\ \ X_{t_1}(R2\ \ X_{t_2})))$
remove all labels	$(R\ \ (R_1\ \ X_{t_1}(R_2\ \ X_{t_2}))) \mapsto\ \ (R\ \ X_{t_2}X_{t_1})$

Table 6: Operators for simplifying the trees

are not *translation boundaries*, and to avoid translation errors we should identify them by applying a PSCFG rule on top of them. During training, we label nodes with translation boundaries, as one additional *function tag*; during decoding, we employ the MaxEnt model to predict the translation boundary label probability for each span associated with a subgraph γ, and discourage derivations accordingly for using nonterminals over the non-translation boundary span. The translation boundaries over elementary trees have much richer representation power. The previous works as in Xiong et al. (2010), defined translation boundaries on phrase-decoder style derivation trees due to the nature of their shift-reduce algorithm, which is a special case in our model.

4 Decoding

Decoding using the proposed elementary tree to string grammar naturally resembles bottom up chart parsing algorithms. The key difference is at the grammar querying step. Given a grammar G, and the input source parse tree π from a monolingual parser, we first construct the elementary tree for a source span, and then retrieve all the relevant subgraphs seen in the given grammar through the proposed operators. This step is called populating, using the proposed operators to find all relevant elementary trees γ which may have contributed to explain the source span, and put them in the corresponding cells in the chart. There would have been exponential number of relevant elementary trees to search if we do not have any restrictions in the populating step; we restrict the maximum number of interior nodes $|\gamma.v^i|$ to be 3, and the size of frontier nodes $|\gamma.v^f|$ to be less than 6; additional pruning for less frequent elementary trees is carried out.

After populating the elementary trees, we construct the partial hypotheses bottom up, by rewriting the frontier nodes of each elementary tree with the probabilities(costs) for $\gamma \rightarrow \alpha*$ as in Eqn. 3. Our decoder (Zhao and Al-Onaizan, 2008) is a template-based chart decoder in C++. It generalizes over the dotted-product operator in Earley style parser, to allow us to leverage many operators $\bar{t} \in T$ as above-mentioned, such as binarizations, at different levels for constructing partial hypothesis.

5 Experiments

In our experiments, we built our system using most of the parallel training data available to us: 250M Arabic running tokens, corresponding to the "unconstrained" condition in NIST-MT08. We chose the testsets of *newswire* and *weblog* genres from MT08 and DEV10[1]. In particular, we choose MT08 to enable the comparison of our results to the reported results in NIST evaluations. Our training and test data is summarized in Table 5. For testings, we have 129,908 tokens in our testsets. For language models (LM), we used 6-gram LM trained with 10.3 billion English tokens, and also a shrinkage-based LM (Chen, 2009) – "ModelM" (Chen and Chu, 2010; Emami et al., 2010) with 150 word-clusters learnt from 2.1 million tokens.

From the parallel data, we extract phrase pairs(blocks) and elementary trees to string grammar in various configurations: basic tree-to-string rules (Tr2str), elementary tree-to-string rules with boundaries $\bar{t}$(elm2str+$\bar{m}$), and with both $\bar{t}$ and $\bar{m}$ (elm2str+$\bar{t}$ + $\bar{m}$). This is to evaluate the operators' effects at different levels for decoding. To learn our MaxEnt models defined in § 3.3, we collect the events during extracting elm2str grammar in training time, and learn the model using improved iterative scaling. We use the same training data as that used in training our Arabic parser. There are 16 thousand human parse trees with human alignment; additional 1 thousand human parse and aligned sent-pairs are used as unseen test set to verify our MaxEnt models and parsers. For our Arabic parser, we have a labeled F-measure of 78.4%, and POS tag accuracy 94.9%. In particular, we'll evaluate model $p_c(\bar{t}|\gamma, \bar{m})$ in Eqn. 3 for predicting the translation boundaries in § 3.5.3 for projectable spans as detailed in § 5.1.

Our decoder (Zhao and Al-Onaizan, 2008) supports grammars including monotone, ITG, Hiero, tree-to-string, string-to-tree, and several mixtures of them (Lee et al., 2010). We used 19 feature functions, mainly from those used in phrase-based decoder like Moses (Koehn et al., 2007), including two language models (one for a 6-gram LM, one for ModelM, one brevity penalty, IBM Model-1 (Brown et al., 1993) style alignment probabilities in both directions, relative frequency in both directions, word/rule counts, content/function word mismatch, together with features on tr2str rule probabilities. We use BLEU (Papineni et al., 2002) and TER (Snover et al., 2006) to evaluate translation qualities. Our baseline used basic elementary tree to string grammar without any manipulations and boundary markers in the model,

[1]DEV10 are unseen testsets used in our GALE project. It was selected from recently released LDC data LDC2010E43.v3.

Data	Train	MT08-NW	MT08-WB	Dev10-NW	Dev10-WB
# Sents	8,032,837	813	547	1089	1059
# Tokens	349M(ar)/230M(en)	25,926	19,654	41,240	43,088

Table 7: Training and test data; using all training parallel training data for 4 test sets

and we achieved a BLEUr4n4 55.01 for MT08-NW, or a cased BLEU of 53.31, which is close to the best officially reported result 53.85 for unconstrained systems.[2] We expose the statistical decisions in Eqn. 3 as the rule probability as one of the 19 dimensions, and use Simplex Downhill algorithm with Armijo line search (Zhao and Chen, 2009) to optimize the weight vector for decoding. The algorithm moves all dimensions at the same time, and empirically achieved more stable results than MER(Och, 2003) in many of our experiments.

5.1 Predicting Projectable Structures

The projectable structure is important for our proposed elementary tree to string grammar (elm2str). When a span is predicted not to be a translation boundary, we want the decoder to prefer alternative derivations outside of the immediate elementary tree, or more aggressive manipulation of the trees, such as deleting interior nodes, to explore unlabeled grammar such as Hiero style rules, with proper costs. We test separately on predicting the projectable structures, like predicting function tags in § 3.5.3, for each node in syntactic parse tree. We use one thousand test sentences with two conditions: human parses and machine parses. There are totally 40,674 nodes excluding the sentence-level node. The results are shown in Table 8. It showed our MaxEnt model is very accurate using human trees: 94.5% of accuracy, and about 84.7% of accuracy for using the machine parsed trees. Our accuracies are higher compared with the 71+% accuracies reported in (Xiong et al., 2010) for their phrasal decoder.

Setups	Accuracy
Human Parses	94.5%
Machine Parses	84.7%

Table 8: Accuracies of predicting projectable structures

We zoom in the translation boundaries for MT08-NW, in which we studied a few important frequent labels including VP and NP-SBJ as in Table 9. According to our MaxEnt model, 20% of times we should discourage a VP tree to be translated contiguously; such VP trees have an average span length of 16.9 tokens in MT08-NW. Similar statistics are 15.9% for S-tree with an average span of 13.8 tokens.

[2]See link: http://www.itl.nist.gov/iad/mig/tests/mt/2008/doc/mt08_official_results_v0.html

Labels	total	NonProj	Percent	Avg.len
VP*	4479	920	20.5%	16.9
NP*	14164	825	5.8%	8.12
S*	3123	495	15.9%	13.8
NP-SBJ*	1284	53	4.12%	11.9

Table 9: The predicted projectable structures in MT08-NW

Using the predicted projectable structures for elm2str grammar, together with the probability defined in Eqn. 3 as additional cost, the translation results in Table 11 show it helps BLEU by 0.29 BLEU points (56.13 v.s. 55.84). The boundary decisions penalize the derivation paths using nonterminals for non-projectable spans for partial hypothesis construction.

Setups	TER	BLEUr4n4
Baseline	39.87	55.01
right-binz (rbz)	39.10	55.19
left-binz (lbz)	39.67	55.31
Head-out-left (hlbz)	39.56	55.50
Head-out-right (hrbz)	39.52	55.53
+all binzation (abz)	39.42	55.60
+regroup-verb	39.29	55.72
+deleting interior nodes $\gamma.v^i$	38.98	55.84

Table 10: TER and BLEU for MT08-NW, using only $\bar{t}(\gamma)$

5.2 Integrating $\bar{t}$ and $\bar{m}$

We carried out a series of experiments to explore the impacts using $\bar{t}$ and $\bar{m}$ for elm2str grammar. We start from transforming the trees via simple operator $\bar{t}(\gamma)$, and then expand the function with more tree context to include the neighboring functions: $\bar{t}(\gamma, \bar{m})$.

Setups	TER	BLEUr4n4
Baseline w/ $\bar{t}$	38.98	55.84
+ TM Boundaries	38.89	56.13
+ SENT Bound	38.63	56.46
all $\bar{t}(\gamma, \bar{m})$	38.61	56.87

Table 11: TER and BLEU for MT08-NW, using $\bar{t}(\gamma, \bar{m})$.

Experiments in Table 10 focus on testing operators especially binarizations for transforming the trees. In Table 10, the four possible binarization methods all improve

Data	MT08-NW	MT08-WB	Dev10-NW	Dev10-WB
Tr2Str	55.01	39.19	37.33	41.77
elm2str+$\bar{t}$	55.84	39.43	38.02	42.70
elm2str+$\bar{m}$	55.57	39.60	37.67	42.54
elm2str+$\bar{t}(\gamma, \bar{m})$	56.87	39.82	38.62	42.75

Table 12: BLEU scores on various test sets; comparing elementary tree-to-string grammar (tr2str), transformation of the trees (elm2str+$\bar{t}$), using the neighboring function for boundaries (elm2str+$\bar{m}$), and combination of all together (elm2str+$\bar{t}(\gamma, \bar{m})$). MT08-NW and MT08-WB have four references; Dev10-WB has three references, and Dev10-NW has one reference. BLEUn4 were reported.

over the baseline from +0.18 (via right-most binarization) to +0.52 (via head-out-right) BLEU points. When we combine all binarizations (abz), we did not see additive gains over the best individual case – hrbz. Because during our decoding time, we do not frequently see large number of children (maximum at 6), and for smaller trees (with three or four children), these operators will largely generate same transformed trees, and that explains the differences from these individual binarization are small. For other languages, these binarization choices might give larger differences. Additionally, regrouping the verbs is marginally helpful for BLEU and TER. Upon close examinations, we found it is usually beneficial to group verb (PV or IV) with its neighboring nodes for expressing phrases like "have to do" and "will not only". Deleting the interior nodes helps on shrinking the trees, so that we can translate it with more statistics and confidences. It helps more on TER than BLEU for MT08-NW.

Table 11 extends Table 10 with neighboring function to further disambiguate the reordering rule using the tree context. Besides the translation boundary, the reordering decisions should be different with regard to the positions of the elementary tree relative to the sentence. At the sentence-beginning one might expect more for monotone decoding, while in the middle of the sentence, one might expect more reorderings. Table 11 shows when we add such boundary markups in our rules, an improvement of 0.33 BLEU points were obtained (56.46 v.s. 56.13) on top of the already improved setups. A close check up showed that the sentence-begin/end markups significantly reduced the leading "and" (from Arabic word w#) in the decoding output. Also, the verb subject order under SBAR seems to be more like monotone with a leading pronoun, rather than the general strong reordering of moving verb after subject. Overall, our results showed that such boundary conditions are helpful for executing the correct reorderings. We conclude the investigation with full function $\bar{t}(\gamma, \bar{m})$, which leads to a BLEUr4n4 of 56.87 (cased BLEUr4n4c 55.16), a significant improvement of 1.77 BLEU point over a already strong baseline.

We apply the setups for several other NW and WEB datasets to further verify the improvement. Shown in Table 12, we apply separately the operators for $\bar{t}$ and $\bar{m}$ first, then combine them as the final results. Varied improvements were observed for different genres. On DEV10-NW, we observed 1.29 BLEU points improvement, and about 0.63 and 0.98 improved BLEU points for MT08-WB and DEV10-WB, respectively. The improvements for newwire are statistically significant. The improvements for weblog are, however, only marginally better. One possible reason is the parser quality for web genre is reliable, as our training data is all in newswire. Regarding to the individual operators proposed in this paper, we observed consistent improvements of applying them across all the datasets. The generative model in Eqn. 3 leverages the operators further by selecting the best transformed tree form for executing the reorderings.

5.3 A Translation Example

To illustrate the advantages of the proposed grammar, we use a testing case with long distance word reordering and the source side parse trees. We compare the translation from a strong phrasal decoder (DTM2) (Ittycheriah and Roukos, 2007), which is one of the top systems in NIST-08 evaluation for Arabic-English. The translations from both decoders with the same training data (LM+TM) are in Table 13. The highlighted parts in Figure 3 show that, the rules on partial trees are effectively selected and applied for capturing long-distance word reordering, which is otherwise rather difficult to get correct in a phrasal system even with a MaxEnt reordering model.

6 Discussions and Conclusions

We proposed a framework to learn models to predict how to transform an elementary tree into its simplified forms for better executing the word reorderings. Two types of operators were explored, including (a) transforming the trees via binarizations, grouping or deleting interior nodes to change the structures; and (b) neighboring boundary context to further disambiguate the reordering decisions. Significant improvements were observed on top of a strong baseline system, and consistent improvements were observed across genres; we achieved a cased BLEU of 55.16 for MT08-NW, which is significantly better than the officially reported results in NIST MT08 Arabic-English evaluations.

Src Sent	qAl AlAmyr EbdAlrHmn bn EbdAlEzyz nA}b wzyr AldfAE AlsEwdy AlsAbq fy tSryH SHAfy An +h mtfA}l b# qdrp Almmlkp Ely AyjAd Hl l# Alm$klp .
Phrasal Decoder	prince abdul rahman bin abdul aziz , deputy minister of defense former saudi said in a press statement that he was optimistic about the kingdom 's ability to find a solution to the problem .
Elm2Str+$\bar{t}(\gamma, \bar{m})$	former saudi deputy defense minister prince abdul rahman bin abdul aziz said in a press statement that he was optimistic of the kingdom 's ability to find a solution to the problem .

Table 13: A translation example, comparing with phrasal decoder.

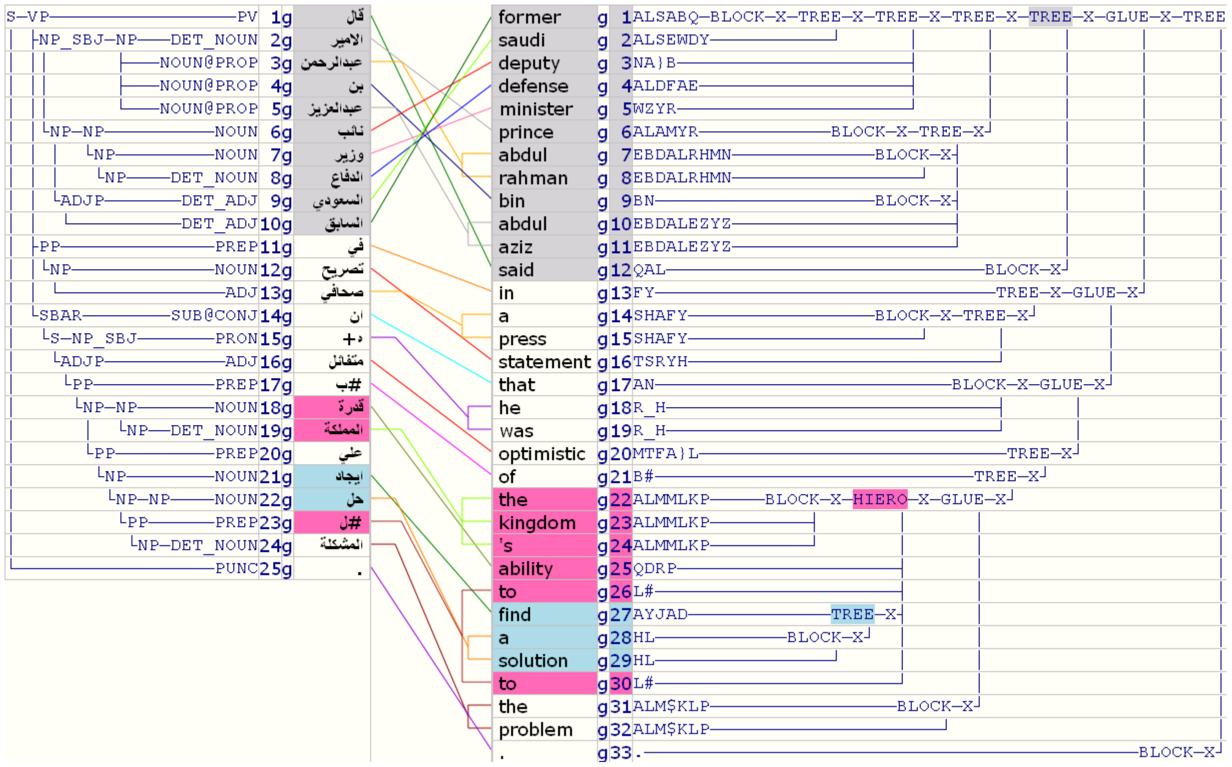

Figure 3: A testing case: illustrating the derivations from chart decoder. The left panel is source parse tree for the Arabic sentence — the input to our decoder; the right panel is the English translation together with the simplified derivation tree and alignment from our decoder output. Each "X" is a nonterminal in the grammar rule; a "Block" means a phrase pair is applied to rewrite a nonterminal; "Glue" and "Hiero" means the unlabeled rules were chosen to explain the span as explained in § 3.4.3 ; "Tree" means a labeled rule is applied for the span. For instance, for the source span [1,10], a rule is applied on a partial tree with PV and NP-SBJ; for the span [18,23], a rule is backed off to an unlabeled rule (Hiero-alike); for the span [21,22], it is another partial tree of NPs.

Within the proposed framework, we also presented several special cases including the translation boundaries for nonterminals in SCFG for translation. We achieved a high accuracy of 84.7% for predicting such boundaries using MaxEnt model on machine parse trees. Future works aim at transforming such non-projectable trees into projectable form (Eisner, 2003), driven by translation rules from aligned data(Burkett et al., 2010), and informative features form both the source [3] and the target sides (Shen et al., 2008) to enable the system to leverage more isomorphic trees, and avoid potential detour errors. We are exploring the incremental decoding framework, like (Huang and Mi, 2010), to improve pruning and speed.

[3]The BLEU score on MT08-NW has been improved to 57.55 since the acceptance of this paper, using the proposed technique but with our GALE P5 data pipeline and setups.

Acknowledgments

This work was partially supported by the Defense Advanced Research Projects Agency under contract No. HR0011-08-C-0110. The views and findings contained in this material are those of the authors and do not necessarily reflect the position or policy of the U.S. government and no official endorsement should be inferred.

We are also very grateful to the three anonymous reviewers for their suggestions and comments.

References

Peter F. Brown, Stephen A. Della Pietra, Vincent. J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. In *Computational Linguistics*, volume 19(2), pages 263–331.

David Burkett, John Blitzer, and Dan Klein. 2010. Joint parsing and alignment with weakly synchronized grammars. In *Proceedings of HLT-NAACL*, pages 127–135, Los Angeles, California, June. Association for Computational Linguistics.

Marine Carpuat, Yuval Marton, and Nizar Habash. 2010. Reordering matrix post-verbal subjects for arabic-to-english smt. In *17th Confrence sur le Traitement Automatique des Langues Naturelles*, Montral, Canada, July.

Stanley F. Chen and Stephen M. Chu. 2010. Enhanced word classing for model m. In *Proceedings of Interspeech*.

Stanley F. Chen. 2009. Shrinking exponential language models. In *Proceedings of NAACL HLT,*, pages 468–476.

David Chiang. 2007. Hierarchical phrase-based translation. In *Computational Linguistics*, volume 33(2), pages 201–228.

David Chiang. 2010. Learning to translate with source and target syntax. In *Proc. ACL*, pages 1443–1452.

Chris Dyer, Hendra Setiawan, Yuval Marton, and Philip Resnik. 2009. The University of Maryland statistical machine translation system for the Fourth Workshop on Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 145–149, Athens, Greece, March.

Jason Eisner. 2003. Learning Non-Isomorphic tree mappings for Machine Translation. In *Proc. ACL-2003*, pages 205–208.

Ahmad Emami, Stanley F. Chen, Abe Ittycheriah, Hagen Soltau, and Bing Zhao. 2010. Decoding with shrinkage-based language models. In *Proceedings of Interspeech*.

Bryant Huang and Kevin Knight. 2006. Relabeling syntax trees to improve syntax-based machine translation quality. In *Proc. NAACL-HLT*, pages 240–247.

Liang Huang and Haitao Mi. 2010. Efficient incremental decoding for tree-to-string translation. In *Proceedings of EMNLP*, pages 273–283, Cambridge, MA, October. Association for Computational Linguistics.

Zhongqiang Huang, Martin Cmejrek, and Bowen Zhou. 2010. Soft syntactic constraints for hierarchical phrase-based translation using latent syntactic distributions. In *Proceedings of the 2010 EMNLP*, pages 138–147.

Abraham Ittycheriah and Salim Roukos. 2007. Direct translation model 2. In *Proc of HLT-07*, pages 57–64.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*, pages 177–180.

Young-Suk Lee, Bing Zhao, and Xiaoqian Luo. 2010. Constituent reordering and syntax models for english-to-japanese statistical machine translation. In *Proceedings of Coling-2010*, pages 626–634, Beijing, China, August.

Yang Liu and Qun Liu. 2010. Joint parsing and translation. In *Proceedings of COLING 2010,*, pages 707–715, Beijing, China, August.

Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. Spmt: Statistical machine translation with syntactified target language phraases. In *Proceedings of EMNLP-2006*, pages 44–52.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of ACL-08: HLT*, pages 1003–1011.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In *Proceedings of EMNLP 2008*, pages 206–214.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *In Proceedings of ACL-HLT*, pages 192–199.

Franz Josef Och. 2003. Minimum error rate training in Statistical Machine Translation. In *ACL-2003*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of the ACL-02)*, pages 311–318, Philadelphia, PA, July.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June. Association for Computational Linguistics.

Libin Shen, Bing Zhang, Spyros Matsoukas, Jinxi Xu, and Ralph Weischedel. 2010. Statistical machine translation with a factorized grammar. In *Proceedings of the 2010 EMNLP*, pages 616–625, Cambridge, MA, October. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *AMTA*.

W. Wang, J. May, K. Knight, and D. Marcu. 2010. Restructuring, re-labeling, and re-aligning for syntax-based statistical machine translation. In *Computational Linguistics*, volume 36(2), pages 247–277.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. In *Computational Linguistics*, volume 23(3), pages 377–403.

Deyi Xiong, Min Zhang, and Haizhou Li. 2010. Learning translation boundaries for phrase-based decoding. In *NAACL-HLT 2010*, pages 136–144.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *ACL-HLT*, pages 559–567.

Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw, and Chew Lim Tan. 2009. Forest-based tree sequence to string translation model. In *Proc. of ACL 2009*, pages 172–180.

Bing Zhao and Yaser Al-Onaizan. 2008. Generalizing local and non-local word-reordering patterns for syntax-based machine translation. In *Proceedings of EMNLP*, pages 572–581, Honolulu, Hawaii, October.

Bing Zhao and Shengyuan Chen. 2009. A simplex armijo downhill algorithm for optimizing statistical machine translation decoding parameters. In *Proceedings of HLT-NAACL*, pages 21–24, Boulder, Colorado, June. Association for Computational Linguistics.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proc. of NAACL 2006 - Workshop on SMT*, pages 138–141.