The UOT System: Improve String-to-Tree Translation Using Head-Driven Phrase Structure Grammar and Predicate-Argument Structures

Xianchao Wu[†]*, Takuya Matsuzaki*[†]*, Naoaki Okazaki*[†]*, Yusuke Miyao*[†]*, Jun'ichi Tsujii*[†‡]

[†]Department of Computer Science, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
[‡]School of Computer Science, University of Manchester
National Centre for Text Mining (NaCTeM)
Manchester Interdisciplinary Biocentre, 131 Princess Street, Manchester M1 7DN, UK

{wxc, matuzaki, okazaki, miyao, tsujii}@is.s.u-tokyo.ac.jp

Abstract

We present the UOT Machine Translation System that was used in the IWSLT-09 evaluation campaign. This year, we participated in the BTEC track for Chinese-to-English translation. Our system is based on a string-to-tree framework. To integrate *deep* syntactic information, we propose the use of parse trees and semantic dependencies on English sentences described respectively by Head-driven Phrase Structure Grammar and Predicate-Argument Structures. We report the results of our system on both the development and test sets.

1. Introduction

How can we integrate *deep* syntactic information into current statistical machine translation (SMT) systems to further improve the accuracy and fluency? How to deal with global reordering problem for a language pair that do not share isomorphic syntactic structures? These remain to be essential issues faced by current SMT research community.

In this paper, we manage to answer these questions in terms of string-to-tree translation [1, 2, 3]. English, the target language in our case study, is the most popularly researched language with plenty resources and syntactic parsers. In contrast to commit to Probabilistic Context-Free Grammar (PCFG) which only generates shallow trees of English [1, 2, 3], we propose the use of *deep* parse trees and *semantic dependencies* described respectively by Head-driven Phrase Structure Grammar (HPSG) [4, 5] and Predicate-Argument Structures (PASs).

We illustrate two major characteristics that an HPSG tree (used by us) differs from a PCFG tree. First, a node in an HPSG tree is represented by a *typed feature structure* (TFS) with richer information (Table 1) than a PCFG node that is commonly represented by only POS/phrasal tags. Second, PASs, which describe the semantic relations among a predicate (can be a verb, adjective, preposition, etc.) and its arguments, are used for guiding local/global reordering during translation. The idea proposed in this paper can be considered as a *natural* integration of syntactic information and semantic dependency information for assisting string-to-tree translation. We call the integration *natural* here, because the HPSG tree and PAS of an English sentence are generated synchronously by using a state-of-the-art HPSG parser on English, Enju[1] [6].

Feature	Description
CAT	phrasal category
XCAT	fine-grained phrasal category
SCHEMA	name of the schema applied in the node
HEAD	pointer to the head daughter
SEM_HEAD	pointer to the semantic head daughter
POS	part-of-speech
TENSE	tense of a verb (past, present, etc.)
VOICE	voice of a verb (passive/active)
PRED	type of a predicate
ARG$\langle n \rangle$	pointer to semantic arguments

Table 1: *Examples of syntactic/semantic features extracted from HPSG signs that are included in the output of Enju (top and bottom stands for features of phrasal and lexical nodes, respectively).*

Note that the information available in the output of Enju is a fairly crude approximation of the TFS used in the full HPSG grammar [4, 5] due to practicable considerations [6]. Although the information taken from Enju's output is much more than the commonly used PCFG parser, the HPSG-based translation rule is still extracted from an approximation of the full HPSG grammar.

2. System Outline

2.1. Parameter Estimation

The diagram of parameter estimation in our system is shown in Figure 1, which is similar to most syntax-based SMT sys-

[1]http://www-tsujii.is.s.u-tokyo.ac.jp/enju/index.html

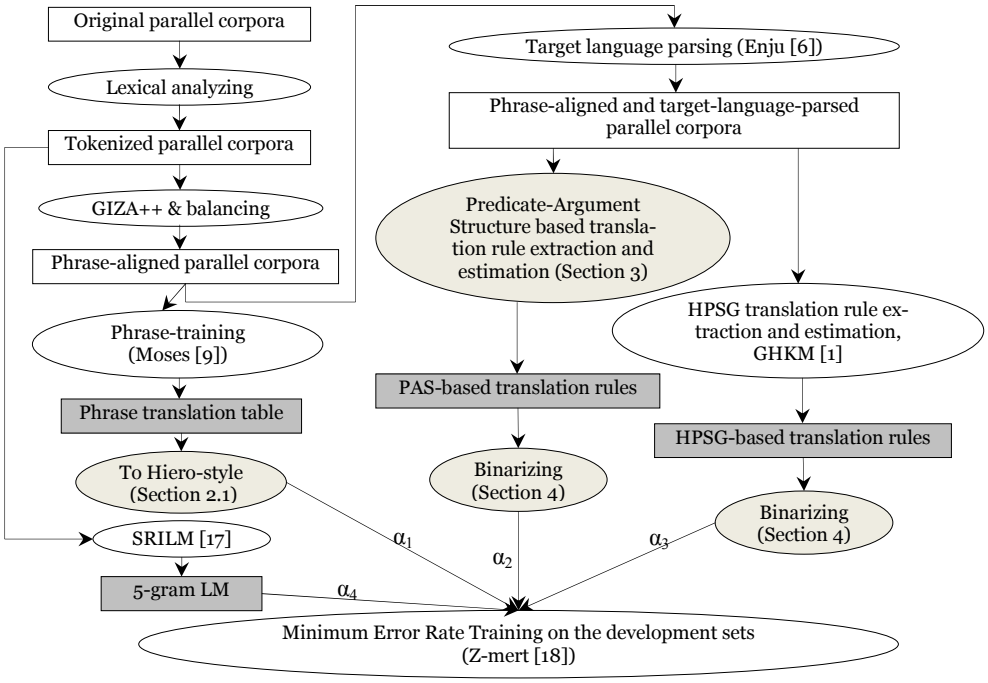

Figure 1: *The parameter estimation and rule combination diagram of our system.*

tems [1, 7]. Given bilingual parallel corpora, we first tokenize the source and target sentences (e.g., word segmentation of Chinese; punctuation segmentation and lowercase of English). Then, we use GIZA++ [8] and *grow-diag-final-and* balancing strategy (dealing with unaligned source/target words) [9] on the tokenized parallel corpora to obtain a phrase-aligned parallel corpora. A phrase translation table (PTT) is estimated from the phrase-aligned parallel corpora. We implement the step of phrase table extraction employing the Moses toolkit[2] [9].

Recall that the Moses-style phrase translation rule takes the following form:

$$f_1^m \;|||\; e_1^n \;|||\; a_1 \;|||\; a_2 \;|||\; p(f|e)\; l(f|e)\; p(e|f)\; l(e|f)\; wp. \quad (1)$$

Here, a_1 and a_2 are word alignments between a source language phrase f_1^m and a target language phrase e_1^n; $p(\cdot)$ and $l(\cdot)$ are phrasal translation probabilities and lexical weights, respectively; wp stands for the word penalty.

In order to be used in CKY-style decoding [10], a rule in the form of (1) can be easily transformed into an end-to-end Hiero-style [10] translation rule:

$$[X] \;|||\; f_1^m \;|||\; e_1^n \;|||\; p(f|e)\; l(f|e)\; p(e|f)\; l(e|f)\; wp. \quad (2)$$

And the corresponding synchronous PCFG production takes the form of:

$$X \rightarrow_{\sum_i w_i * \log(p_i)} f_1^m, e_1^n. \quad (3)$$

It has been proved that the end-to-end phrase table significantly influence the translation result of syntax-based systems [11, 12]. Note that the phrase table mentioned here is not constrained to be linguistic phrases, i.e., a phrase inside the table is not necessarily covered by a subtree. This makes the phrase table more flexible to be used than in the tree/forest-to-string systems [11, 12] where additional approaches have to be employed in order to make use of non-linguistic phrases [13, 14].

Return back to Figure 1, HPSG trees attached with PASs are generated by parsing the target language sentences using Enju. Then, we extract HPSG and PAS based translation rules from the word-aligned and target-language-parsed parallel corpora.

The HPSG-based **xRs** (tree-to-string, [15]) translation rules (binarized inversely [16]) are extracted using the GHKM minimal-rule extraction algorithm of [1]. In order to trace the lexical level translation information (similar to PAS-based rules (Section 3 and Figure 2)), we remember the end-to-end alignments in an HPSG-based translation rule. The ideas are described in [1, 11] in detail. In order to use the dependency structure, we describe a linear-time algorithm based on *minimum covering trees* to extract PAS-based translation rules (Section 3).

SRI Language Modeling (SRILM) toolkit[3] [17] is employed to train a 5-gram language model (LM) on the tokenized target language side with Kneser-Ney smoothing. Toolkit Z-mert[4] [18] is used for Minimum-Error Rate Training (MERT) [19].

[2] http://www.statmt.org/moses/

[3] http://www.speech.sri.com/projects/srilm/

[4] http://www.cs.jhu.edu/ ozaidan/zmert/

2.2. Rule Combination

Since the PTT, the HPSG-based rules, and the PAS-based rules are independently extracted and estimated, the distribution overlapping among them is inevitable. As shown in Figure 1, we use Z-mert to tune their weights on the development sets. The optimal derivation is computed by:

$$d^* = \arg\max_{d \in D} \left\{ \sum_{i=1}^{3} \sum_{j_i} \log p_{j_i}^{\alpha_i w_{j_i}}(d) + \log p_{LM}^{\alpha_4}(d) \right\}.$$

Here, p_{j_1}, p_{j_2}, and p_{j_3} represent the feature subsets from PPT, binarized PAS-based rules, and binarized HPSG-based rules, respectively. Bidirectional translation/lexical probabilities are included in the three subsets. In addition, number of phrases and words are contained in p_{j_1}, and number of rules are included in p_{j_2} and p_{j_3}.

2.3. Decoding

We use a CKY-style algorithm with beam-pruning and cube-pruning [10] to decode Chinese sentences. For efficient decoding with integrated N-gram LMs, we binarize all translation rules into rules that contain at most two variables and can be incrementally scored by LM [16]. For each source language sentence f, the output of the chart-parsing algorithm is expressed as a *hyper-graph* representing a set of derivations. Given a hyper-graph for f, we use the Algorithm 3 described in [20] to extract its k-best derivations. Since different derivations may lead to the same target language string e, we further adopt Algorithm 3's modification (i.e., keep a hash-table to maintain the unique target sentences [21]) to efficiently generate the *unique* k-best translations.

3. PAS-based Translation Rule Extraction

In this section, we first express an example of an HPSG tree attached with PASs, and then describe the data structure and an extraction algorithm of PAS-based translation rules (short as PASR, hereafter).

3.1. Motivation

We start by observing an HPSG tree of the English sentence *She ignored the fact that I wanted to dispute.*, and the PAS for a verb *ignored*, as shown in Figure 2. Being the predicate in this sentence, *ignored* has two arguments. The first is *She*, which is the subject of the whole sentence. And the second is *fact*, which is modified by a relative clause. The PAS type of *ignored* is named *verb_arg*12. In which, '1' and '2' stand for the subject and direct-object arguments, respectively.

There is a restricted order of *She*, *ignored*, and *fact* in the English sentence. In contrast, the word order in the corresponding Chinese sentence is also restricted in the sequence of *ta*, *wushi*, and *shishi*. Thus, a translation rule can be extracted when we are provided with an alignment (as shown in the bottom of Figure 2) without overlapping among these three elements in the source and target language sides. In particular, we observe that the "head" of this rule is *ignored* whose arguments can be generalized into variables.

Note that PASs are not only attached to verbs in a sentence, but also to all other words in the sentence. The corresponding PASRs are illustrated in Figure 2 as well. Even apparently similar in data structure, we argue our PASRs are still different from the traditional **xRs** rules [1, 11, 21], since the knowledge of semantic dependencies are further explicitly employed. We give the formal definitions and a linear-time rule extraction algorithm in the following subsections.

3.2. Definitions

Using a strategy similar to most string-to-tree systems [3], we define a PASR to be an **xRs** rule and binarize it in an inverse way [16].

3.2.1. Definition of PASR

A PASR is a 4-tuple $\langle S, T, A, n \rangle$, which describes the alignment A between a substring $S = f_{j_1}^{j_2}$ and an HPSG subtree $T = T(e_{i_1}^{i_2})$. n is the count of the rule.

In order to generalize the arguments of the "head" word in a PAS yet retaining lexical information, we require T to include only one terminal node e^*. T is extracted from the best HPSG parse tree $T(e_1^I)$ of sentence e_1^I in such a way that T covers all arguments of e^*, which correspond to some non-terminal nodes in $T(e_1^I)$, in addition to the terminal node of e^*. Moreover, $T(e_{i_1}^{i_2})$ is restricted to be a *minimum* sub-tree that satisfies this constraint. String $e_{i_1}^{i_2}$, which is the postorder traversal sequence of leaf[5] nodes of $T(e_{i_1}^{i_2})$, consists of only one terminal node (e^*) and some non-terminal nodes (phrasal categories). Examples can be found in Figure 2.

The string $f_{j_1}^{j_2}$ is also composed of both terminals (contiguous or non-contiguous words which are aligned with e^*) and non-terminals (placeholders). We regard two PASRs to be identical (and n will be accumulated) if they contain the same S, T, and A.

3.2.2. Definition of Minimum Covering Tree

Suppose we are given a parse tree $\mathcal{T}_0 = \langle \mathcal{N}_0, \mathcal{E}_0 \rangle$, where $\mathcal{N}_0 = \{n_1, n_2, ..., n_M\}$ is a set of nodes and $\mathcal{E}_0 \subseteq \mathcal{N}_0 \times \mathcal{N}_0$ is a set of edges. For two nodes $n_1, n_2 \in \mathcal{N}_0$, we say $n_1 \sqsubseteq n_2$, if n_1 is a descendant node of n_2. Using this descendant-ancestor relation $\sqsubseteq$, we define a *least upper node* of $\mathcal{N}$ ($\subseteq \mathcal{N}_0$), which is the lowest common ancestor node of all the nodes in $\mathcal{N}$.

Definition 1 (**Least Upper Node**) $n_{lun} \in \mathcal{N}_0$ is called the least upper node of a set of nodes $\mathcal{N} \subseteq \mathcal{N}_0$, if the following conditions are satisfied:

- $\forall n_m \in \mathcal{N}$, $n_m \sqsubseteq n_{lun}$;

[5] We use *leaf* to denote a terminal/non-terminal node that does not have any descendant nodes in a (sub)tree.

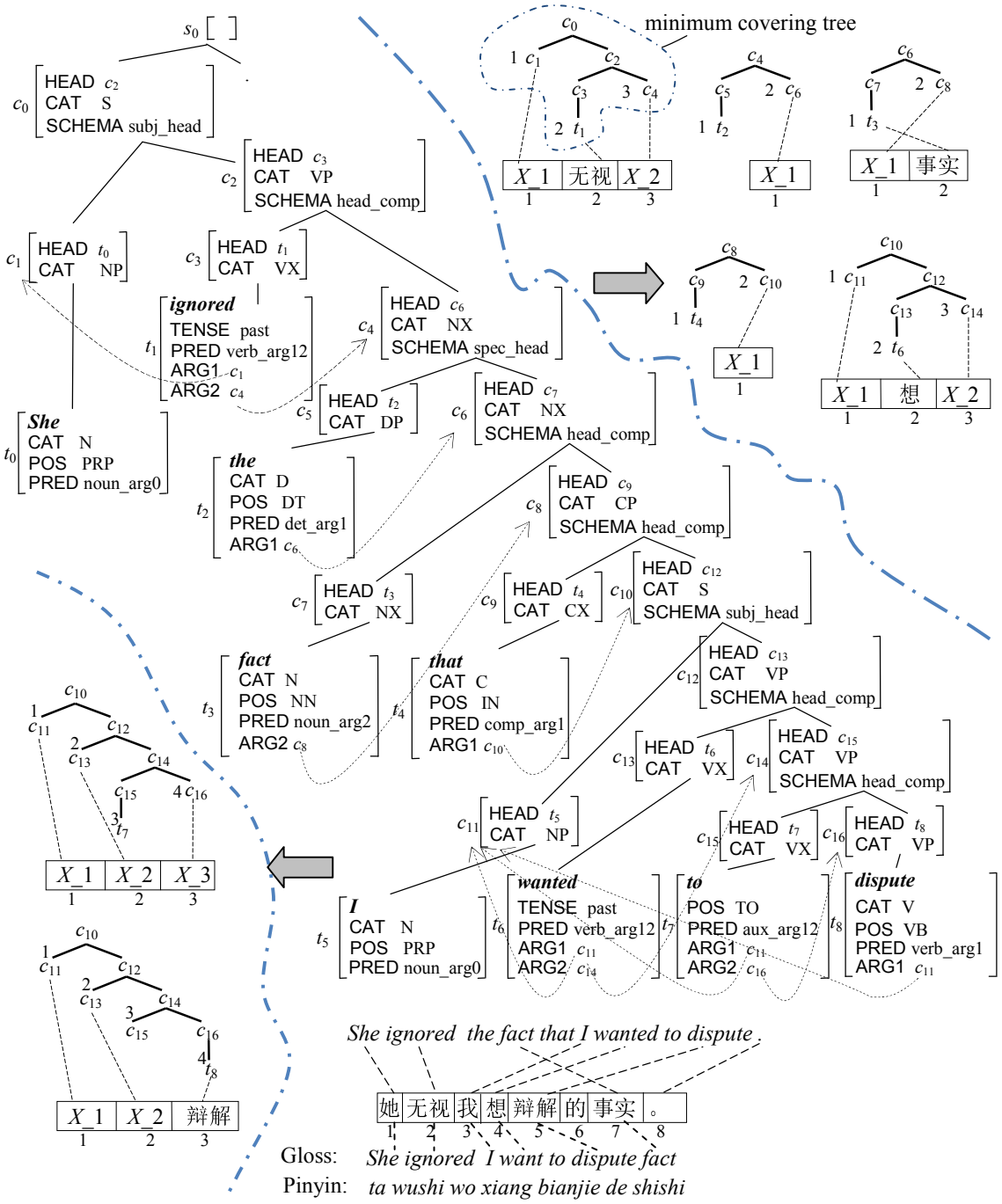

Figure 2: *PASR extraction based on the predicate-argument structure in an HPSG tree. The figure contains three parts: the English-Chinese sentence pair and the alignment (bottom), the HPSG tree for the English sentence (middle), and the PASRs extracted (top-right and bottom-left). Arrows with broken lines denote the PAS dependencies from the terminal nodes to their argument nodes.*

Algorithm 1 PASR extraction

Require: f_1^J, $T = T(e_1^I)$, $A = \{a\} = \{(j,i)\}$

1: PASRs = {}
2: **for** *node* $\in$ Leaves(T) **do**
3: **if** Open(*node*.PRED) and AlignCheck(*node*.index, A) **then**
4: T_c = MinimumCoveringSubtree(T, *node*, *node*.ARGs)
5: *Regions* = fWordRegion(Leaves(T_c), A)
6: **if** RegionCheck(*Regions*) **then**
7: S = Generalize_f_String(Leaves(T_c), e_1^I, *Regions*);
8: A' = ConstructAlignment(Leaves(T_c), S, A);
9: PASRs = PASRs $\cup$ {new PASR(S, T_c, A', 1)}
10: **end if**
11: **end if**
12: **end for**
13: **return** PASRs

- for any common ancestor node n' of $\mathcal{N}$, $n_{lun} = n'$ or $n_{lun} \sqsubseteq n'$.

Let *lun($\mathcal{N}$)* denote the least upper node of $\mathcal{N}$. Obviously, *lun($\mathcal{N}_0$)* is the root node of $\mathcal{T}_0$.

Definition 2 (**Minimum Covering Tree**) $\mathcal{T} = \langle \mathcal{N}, \mathcal{E} \rangle$, which is a subtree of $\mathcal{T}_0$, is called the minimum covering tree of a set of nodes $\mathcal{N}'$ ($\subseteq \mathcal{N}$), if the following conditions are satisfied:

- $\mathcal{N}$ includes all the nodes that appears on the paths of from $\forall n \in \mathcal{N}'$ to *lun($\mathcal{N}'$)*;
- if n and at least one direct child node of n are included in $\mathcal{N}$, then all the direct child nodes of n should be included in $\mathcal{N}$;
- $\forall n_1, n_2 \in \mathcal{N}$, if $(n_1, n_2) \in \mathcal{E}_0$, then $(n_1, n_2) \in \mathcal{E}$.

For example, the minimum covering tree of $\{t_8, c_{11}\}$ is shown in the bottom-left corner in Figure 2. It is easy to see that *lun($\{t_8, c_{11}\}$)* equals to c_{10}. There is an edge that directly connects c_{11} and c_{10}. The path from t_8 to c_{10} also includes the nodes of c_{16}, c_{14}, and c_{12}. The definition of minimum covering trees forces us to include c_{13} (since c_{12} and its direct child c_{14} are included already in the subtree) and c_{15} (since c_{14} and its direct child c_{16} are included already in the subtree) as well. We include c_{13} and c_{15} here to retain the semantic dependencies that depends on the tree-structures. In order to release from the tree-structures in a PASR, it will be interesting to investigate the condition of not including this kind of nodes in the minimum covering tree. We leave this as a future work.

3.3. PASR Extraction Algorithm

In a complete HPSG tree, a PAS is attached to each leaf node which corresponds to a terminal word (c.f. Figure 2). Thus, we extract PASRs by traversing the leaf nodes of an HPSG tree (Lines 2∼12 in Algorithm 1). We explain the extraction algorithm sketched in Algorithm 1 through focusing on extracting a PASR headed by *ignored* (Figure 2).

First, we check whether the PAS in a leaf node (e.g., t_1:*ignored*) is *open* or not, i.e., whether or not the corresponding word has at least one argument (Line 3, first condition). When a word does not have any arguments, x in $arg x$ takes the value of 0 in PRED. In Figure 2, *ignored* has two arguments: ARG1=c_1 and ARG2=c_4, while *She* does not have any arguments and its PRED takes the value of *noun_arg*0. Obviously, PASRs can only be extracted from the open PASs. Also, we check the alignment consistency between e^*=*ignored* and the f_1^J sentence (Line 3, second condition). e^* may align with zero or more f phrase(s), which may be contiguous or non-contiguous. The consistency condition for the terminal node is that, if e^* aligns with some f words, then all these f words can only align with e^* in A. In Figure 2, *ignored* aligns with a word *wushi* that also only aligns back to *ignored*.

Second, we pick a minimum covering subtree T_c from $T(e_1^I)$ that covers e^* and all e^*'s arguments (Line 4). In Figure 2, T_c for *ignored* is rooted at c_0 with leaf nodes to be, from left to right, c_1, t_1, and c_4.

Third, through retrieving A, we look for the f word regions for each leaf node in T_c (Line 5). In Figure 2, the region of c_1 is [1, 1], c_4's is [3, 7], and t_1's is [2, 2].

Fourth, we check the regions to ensure that there are no overlapping among them (Line 6). Obviously, there are no overlapping among [1, 1], [2, 2], and [3, 7]. Generally, we require that there are no overlapping target words that the tree's leaves are aligned to. Under this constraint, each argument in the tree can be generalized into a unique variable in the target string.

Finally, S is generalized to contain terminals and non-terminals (Line 7). A non-terminal (placeholder) in S, which aligns with a leaf of T_c other than e^*, takes the form of *X_*1, *X_*2, etc. S takes the value of *X_1 wushi X_2* in *ignored*'s PASR. The alignment A' is reconstructed between the leaves in T_c and the symbols in S (Line 8). A' takes the value of {(1, 1), (2, 2), (3, 3)} in *ignored*'s PASR in Figure 2. A PASR for *ignored* is created based on T_c, S, and A', with an initial number n to be 1 (Line 9). The PASRs for other words with open PASs (as listed in Figure 2) can be easily extracted through this process.

After using Algorithm 1 to extract PASRs from each sentence pair in the tree-string training corpus, we accumulate the count n of identical PASRs.

4. Binarization of HPSG and PAS based Rules

We use a linear-time algorithm [16] to binarize the HPSG and PAS based translation rules. The main modifications here are as follows. First, we inversely binarize English-tree to Chinese-string rules into Chinese-string to English-string rules. Second, instead of attaching the terminal words greedily during binarizing, we first take the end-to-end alignments into consideration. The process of binarizing a PASR is described in Figure 3. Note the process is also applicable to the HPSG-based rules.

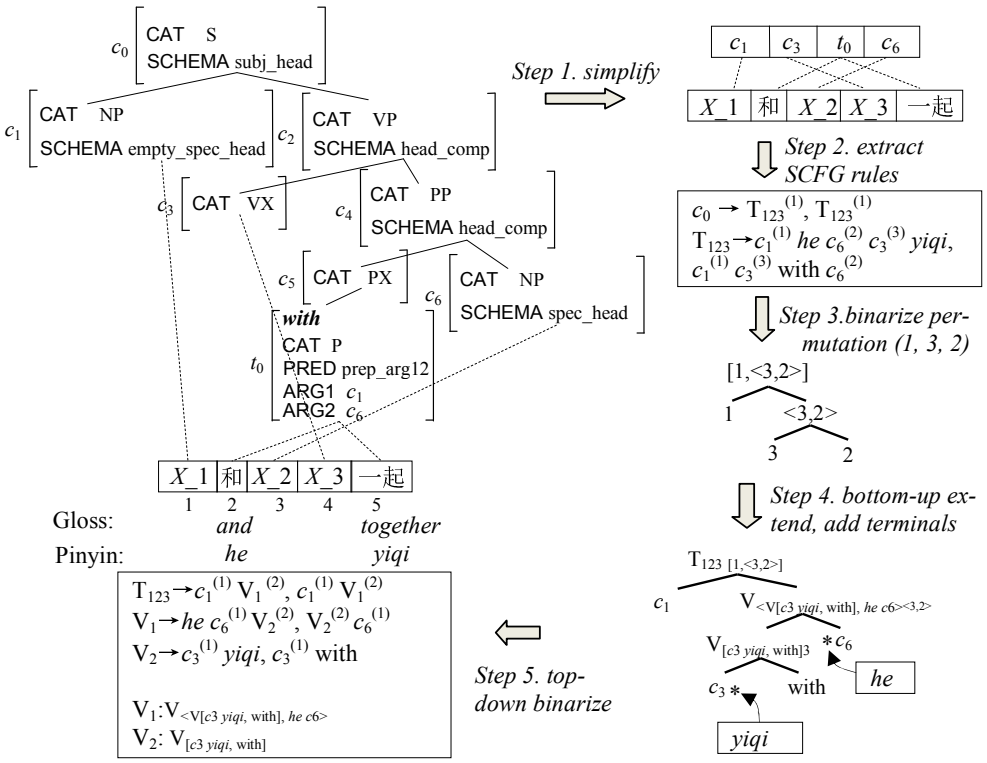

Figure 3: *The binarization process of a predicate-argument structure based rule. * indicates the places where the Chinese words (or contiguous phrases) are appended.*

The PASR we extracted is headed by *with*. We first simplify (Step 1) the tree structure, i.e., postorder traverse the tree, and store the tags (e.g., c_1, t_0, etc.) of the leaf nodes into a list. We create a hash-table to store the tag-to-TFS mapping. Then we extract two flat SCFG rules (Step 2). In a similar manner to [16], we create a specific nonterminal, say, T_{123}, which is a unique identifier for the left-hand side subtree. Note that the English-to-Chinese direction has been changed to Chinese-to-English in the second rule.

The binarization of a permutation into a binary-tree (Step 3) is similar to that described in [16]. Then, we use the end-to-end alignments to guide the attachment of terminals into the binary-tree with only nonterminals (Step 4). In Figure 3, *with* is aligned with two Chinese words *he* and *yiqi* (which is a non-contiguous phrase). Since there are two nonterminals between *he* and *yiqi*, we fail to keep *he* and *yiqi* in one binarized rule following the framework described in [16]. The back-off strategy we use here is to keep *yiqi* and *with* in one binarized rule. Finally, we gain the binarized rules through pre-order traversal of the binary-tree (Step 5).

In order to make use of the binarized rules in the CKY decoding integrated with N-gram LM, we add two kinds of glues rules:

$$S \rightarrow X\_m^{(1)}, X\_m^{(1)}; \quad (4)$$

$$S \rightarrow S^{(1)} X\_m^{(2)}, S^{(1)} X\_m^{(2)}. \quad (5)$$

Here X_m ranges over the nonterminals that appear in the binarized rule set. For example, X_m takes the value of the nonterminals of T_{123}, $c_{\{1,3,6\}}$, and $\mathrm{V}_{\{1,2\}}$ that appear in the binarized rules shown in the bottom-left corner of Figure 3. These glue rules can be seen as an extension of the two glue rules described in [10].

5. Experimental Results

5.1. Setups

Our experiments are based on the IWSLT 2009 Chinese-to-English translation data. IWSLT 2003, 2004, 2005, 2007 development sets are used as our development set. We report the final result on the IWSLT 2008 development set.

It took us 1,715.6 seconds using Enju2.3.1 to parse the 19,972 English sentences in the training data (0.086 seconds per sentence), in which 19,129 sentences were successfully parsed (successful rate of 95.8%).

Table 2 illustrates the statistical information of three PTTs and three kinds of **xRs** translation rules, such as the number of identical trees, the number of rules after binarizing, and the number of glue rules appended for decoding. Since there is only one nonterminal X in the Hiero-style PTT (refer to form (2)), only two glue rules are appended for CKY-decoding. *shpsg* stands for the simplified HPSG-based rules, whereas only the POS/phrasal tags are kept in place of the original TFSs for each tree node. We use *shpsg* to approximate traditional PCFG-based rules.

Rules	# of trees	# of rules	# of binarized rules	# of glue rules
ptt_3	-	89,902	-	2
ptt_4	-	138,036	-	2
ptt_5	-	183,274	-	2
shpsg	21,940	31,143	34,505	11,146
hpsg	26,756	36,213	37,045	13,706
pasr	8,152	12,294	17,294	4,920

Table 2: *Statistical information of three phrase translation tables with maximum-phrase-length from 3 to 5, and three kinds of **xRs** rules.*

For the beam-search decoding in our system, we set the beam size b to be 200 for pruning PTT rules, PASRs, and HPSG-based rules. We extract top-k(=2,000) unique translations for MERT. Since LM integration with cube-pruning does not guarantee strictly monotonic [10], the margin ϵ of b and k was empirically set to be 0.1.

5.2. Results

As mentioned at the beginning of this paper, we focus on answering two questions through our experiments:

- Does TFSs perform better than POS/phrasal tags for SMT?
- Does PASRs help to improve the ordering of SMT?

Driven by these two questions, we design the following experiments:

- *ptt_*{*3,4,5*}: only makes use of a Hiero-style PTT for CKY-decoding. During PPT extraction, the maximum phrase length is restricted to be 3, 4, and 5;
- *ptt_5+shpsg*: makes use of the PTT and inversely binarized simplified HPSG-based translation rules;
- *ppt_5+hpsg*: makes use of the PTT and inversely binarized HPSG-based translation rules;
- *ppt_5+pasr*: makes use of the PTT and inversely binarized PASRs;
- *ppt_5+pasr+hpsg*: makes use of the PTT, inversely binarized PASRs and HPSG rules.

Table 3 lists the BLEU [22] score of our system using different configurations of translation rules. The BLEU scores are computed under the configuration of case-sensitive with punctuation. From this table, we can see that as the maximum phrase length changes from 3 to 5, the BLEU score tends to be slightly better. *ppt_5* performs the best among *ppt_*{*3,4,5*}. No reordering are performed here.

When the *shpsg* rules are appended to *ppt_5*, the BLEU score improves 2.78 points. The syntactic information attached **xRs** rules and linguistically constrained bilingual phrases contribute to this significant improvement. In comparison with *shpsg*, *hpsg* rules perform better with a further BLEU score improvement of 0.44 points. This provides an evidence that TFSs outperform simple POS/phrasal tags.

Rule Configuration	BLEU (%)	Dev.	Test
ppt_3	30.66	2003-2007	2008
ppt_4	30.75	2003-2007	2008
ppt_5	31.31	2003-2007	2008
ppt_5+shpsg	34.09	2003-2007	2008
ppt_5+hpsg	34.53	2003-2007	2008
ppt_5+pasr	32.51	2003-2007	2008
ppt_5+pasr+hpsg	34.65	2003-2007	2008
ppt_5+pasr+hpsg	36.14	2003-2008	2008
ppt_5+pasr+hpsg	35.38	2003-2008	2009

Table 3: *BLEU scores (%) of our system on the test sets under several translation rule configurations and using different development sets for MERT.*

ppt_5+pasr improves 1.2 points compared with *ppt_5*. This suggests the effectiveness of our PASRs on reordering, since there is totally no reordering in *ppt_5*. However, the improvement is weaker than *ppt_5+shpsg* and *ppt_5+hpsg*. Recall from Table 2 that the number of PASRs extracted is approximately 1/3 of the number of HPSG-based rules. Using large scale training data to extract tree-structure independent PASRs will be necessary to avoid the low coverage problem[6].

Finally, when both PASRs and HPSG-based rules are employed in *ppt_5*, the BLEU scores turn to be the best. Under this configuration, our system achieves the BLEU score of 36.14 (%) after employing MERT on the IWSLT 2008 development set. The BLEU score on the IWSLT 2009 test set is 35.38 (%).

6. Conclusion

In this paper, we introduced our string-to-tree system making use of translation rules from Head-driven Phrase Structure Grammar and Predicate-Argument Structure. We first described the training framework. Then we gave the definition and extraction algorithm of PAS-based translation rules through a detailed example. Later we showed the process to binarize the HPSG and PAS based rules assisted by a particular example. Finally, our experiments verified that the *deep* syntactic information, instantiated as typed feature structures and predicate-argument structures in this paper, does help improving the BLEU score of string-to-tree translation. It will be interesting to investigate the effectiveness of our proposal on other source languages, and by making use of larger scale parallel data and richer feature types.

[6] As one reviewer pointed out, a reasonable cause is due to syntax and PAS may not work well on spoken languages, like BTEC data which is less grammatical.

7. Acknowledgements

This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan), Japanese/Chinese Machine Translation Project in Special Coordination Funds for Promoting Science and Technology (MEXT, Japan), and Microsoft Research Asia Machine Translation Theme. We would like to thank the anonymous reviewers for their constructive and heuristic comments.

8. References

[1] M. Galley, M. Hopkins, K. Knight, and D. Marcu, "What's in a translation rule?" in *Proceedings of HLT-NAACL*, 2004.

[2] M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, and I. Thayer, "Scalable inference and training of context-rich syntactic translation models," in *Proceedings of COLING-ACL*, Sydney, 2006, pp. 961–968.

[3] D. Chiang, K. Knight, and W. Wang, "11,001 new features for statistical machine translation," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Boulder, Colorado: Association for Computational Linguistics, June 2009, pp. 218–226.

[4] C. Pollard and I. A. Sag, *Head-Driven Phrase Structure Grammar*. University of Chicago Press, 1994.

[5] I. A. Sag, T. Wasow, and E. M. Bender, *Syntactic Theory: A Formal Introduction*. Number 152 in CSLI Lecture Notes. CSLI Publications, 2003.

[6] Y. Miyao and J. Tsujii, "Feature forest models for probabilistic hpsg parsing," *Computational Lingustics*, vol. 34, no. 1, pp. 35–80, 2008.

[7] Y. Liu, *Research on Tree-to-String Statistical Translation Models*. Ph.D. Dissertation (In Chinese). Institute of Computing Technology, Chinese Academy of Sciences, 2007.

[8] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Computational Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[9] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of the ACL 2007 Demo and Poster Sessions*, 2007, pp. 177–180.

[10] D. Chiang, "Hierarchical phrase-based translation," *Computational Lingustics*, vol. 33, no. 2, pp. 201–228, 2007.

[11] Y. Liu, Q. Liu, and S. Lin, "Tree-to-string alignment templates for statistical machine transaltion," in *Proceedings of COLING-ACL*, Sydney, Australia, 2006, pp. 609–616.

[12] H. Mi, L. Huang, and Q. Liu, "Forest-based translation," in *Proceedings of ACL-08:HLT*, Columbus, Ohio, 2008, pp. 192–199.

[13] Y. Liu, Y. Huang, Q. Liu, and S. Lin, "Forest-to-string statistical translation rules," in *Proceedings of ACL*, Prague, Czech, 2007, pp. 704–711.

[14] H. Zhang, M. Zhang, H. Li, A. Aw, and C. L. Tan, "Forest-based tree sequence to string translation model," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Suntec, Singapore: Association for Computational Linguistics, August 2009, pp. 172–180.

[15] J. Graehl and K. Knight, "Training tree transducers," in *Proceedings of HLT-NAACL*, 2004, pp. 105–112.

[16] H. Zhang, L. Huang, D. Gildea, and K. Knight, "Synchronous binarization for machine translation," in *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*. New York City, USA: Association for Computational Linguistics, June 2006, pp. 256–263.

[17] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Proceedings of International Conference on Spoken Language Processing*, 2002, pp. 901–904.

[18] O. F. Zaidan, "Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems," *The Prague Bulletin of Mathematical Linguistics*, vol. 91, pp. 79–88, 2009.

[19] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proceedings of ACL*, 2003, pp. 160–167.

[20] L. Huang and D. Chiang, "Better k-best parsing," in *Proceedings of IWPT*, 2005.

[21] L. Huang, K. Knight, and A. Joshi, "Statistical syntax-directed translation with extended domain of locality," in *Proceedings of 7th AMTA*, Boston, MA, 2006.

[22] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation." in *Proceedings of ACL*, 2002, pp. 311–318.