Automatic Reordering Rule Generation and Application of Reordering Rules in Stochastic Reordering Model for English-Myanmar Machine Translation

Thinn Thinn Wai
Universtity of Computer Studies,Yagon.

Tin Myat Htwe
University of Computer Studies, Yangon.

Ni Lar Thein
University of Computer Studies, Yagon.

ABSTRACT
Reordering is one of the most challenging and important problems in Statistical Machine Translation. Without reordering capabilities, sentences can be translated correctly only in case when both languages implied in translation have a similar word order. When translating is between language pairs with high disparity in word order, word reordering is extremely desirable for translation accuracy improvement. Our Language, Myanmar is a verb final language and reordering is needed when our language is translated from other languages with different word orders. In this paper, automatic reordering rule generation and application of generated reordering rules in stochastic reordering model is presented. This work is intended to be incorporated into English–Myanmar Machine Translation system. In order to generate reordering rules; English-Myanmar parallel tagged aligned corpus is firstly created. Then reordering rules are generated automatically by using the linguistic information from this parallel tagged aligned corpus. In this paper, proposed function tag and part-of-speech tag reordering rule extraction algorithms are used to generate reordering rule automatically and First Order Markov theory is applied to implement stochastic reordering model.

Keywords
Reordering; English-Myanmar translation; First Order Markov theory; parallel tagged aligned corpus.

1. INTRODUCTION
The goal of statistical machine translation is to translate an input word sequence in the source language into a target language word sequence. In order to improve the translation process, it is possible to perform preprocessing steps before training and translation in both source and target language sequence. In machine translation, reordering is one of the major problems, since different languages have different word order requirements. When an English sentence is translated to Myanmar sentence, the verb in the English sentence must be moved to the end of the sentence in order to obtain the correct word order. On a sub sentential level, Myanmar word order diverges from English mostly within the noun phrase and verb phrase. In Myanmar, noun phrase exhibits multitude of word orders. In chunk level, the noun chunk made up of determiner (DT) and noun (NN) is translated as many patterns such as "DT, NN" (original English order) and "NN, DT" (Myanmar order). Moreover, some function tags are missed and some part-of-speech tags in some chunks are combined together as only one tag. For example, formal subject (F-SUBJ) and verb chunk containing verb-to-be and adjective. Without reordering, the correct word order can't be obtained. Therefore, reordering is necessary for translation from English language to Myanmar Language. In this work, corpus creation procedure and reordering rules generation procedures are described for English-Myanmar statistical machine translation.
Moreover, a stochastic word reordering model based on first order Markov theory is presented. The purpose of this reordering model is to model reordering concerning two levels; word level and chunk level. Based on function tag and pos (part-of-speech) tag information, reordering rules are extracted from parallel tagged corpus. Moreover, lexical information are also used to disambiguate some pos reordering rules consist of determiner.

2. RELATED WORK
Different approaches have been developed to deal with the word order problem. First approaches worked by constraining reordering at decoding time [23]. In [12], the alignment model already introduces restrictions in word order, which leads also to restrictions at decoding time. A comparison of these two approaches can be found in [2]. They have in common that they do not use any syntactic or lexical information; therefore they rely on a strong language model or on long phrases to get the right word order. Other approaches were introduced that use more linguistic knowledge, for example the use of bitext grammars that allow parsing the source and target language [13]. In [21], syntactic information was used to re rank the output of a translation system with the idea of accounting for different reordering at this stage. In [22], a lexicalized block-oriented reordering model is proposed that decides for a given phrase whether the next phrase should be oriented to its left or right.The most recent and very promising approaches that have been demonstrated, reorder the source sentences based on rules learned from an aligned training corpus with a POS-tagged source side [9][20]. These rules are then used to reorder the word sequence in the most likely way.

In our approach we follow the idea proposed in [20] of using a parallel training corpus with a tagged source side to extract rules which allow a reordering before the translation task. Moreover, we use the lexical information for some part of speech (pos) rules to solve ambiguity problems. By doing this we hope to differentiate between these pos rules.

3. SYSTEM OVERVIEW

As shown in Fig 1, there are two key components in this reordering system; rule generation and reordering. In rule generation, reordering rules are automatically extracted from corpus by using rule extraction algorithms. For this corpus creation, Analyzer is used to annotate for English sentence and manual annotation is used for Myanmar sentence. In reordering component, the input sentence is firstly analyzed to extract syntactic structure by using language analyzer [15]. Language Analyzer performs part-of-speech tagging and function tagging on input sentence. Reordering Model performs reordering by taking the syntactic rules extracted from analyzer and reordering rules automatically extracted from rule extraction algorithms as input parameters.

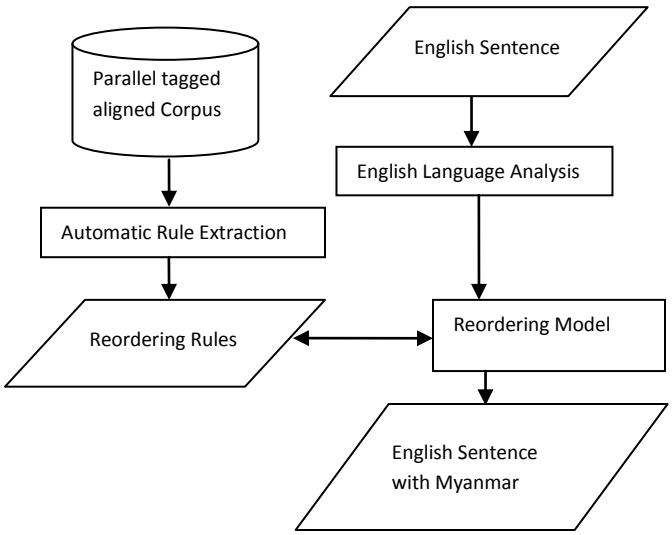


Fig 1: System Overview

4. ESSENTIALNESS OF REORDERING

When English sentence is translated to Myanmar sentence, many differences of word order can be found. In this section, we describe some word order differences; adjective movement, adverb movement, preposition movement and auxiliary verb movement.

Some adjectives (jj) in noun chunk (nc) of English sentence are necessary to move after its relative noun (nn) according to the translation. For example, when the English phrase "rich man" is translated into the Myanmar phrase "လူချမ်းသာ", the adjective "rich (jj)" must be moved after the noun "man (nn)". This can be seen in the Example (1).

Example (1),

nc[rich(jj) man(nn)]

nc[လူ (nn)ချမ်းသာ(jj)]

Myanmar is also modifier and adjunct proceeding language. Therefore, these adjuncts are necessary to move before the relative verb. When the English sentence "He runs quickly" is translated into the Myanmar sentence "သူလျှင်မြန်စွာပြေးသည်", the adverb "quickly" must be moved before its relative verb "run" in order to fit the correct Myanmar order. Such adverb movement can be seen in the Example (2).

Example (2),

He(subj) runs(active) quickly.(advl)

သူ(subj)လျှင်မြန်စွာ(advl)ပြေးသည်(active)

Moreover, auxiliary verbs (md) in verb chunk (vc) are necessary to move after the main verb in order to get an appropriate word order. Therefore, auxiliary verb movement also helps in English-Myanmar Translation. This auxiliary verb movement can be seen in the Example (3).

Example (3),

vc[can(md) go(vb)]

vc[သွား(vb)နိုင်သည်။(md)]

All of these above necessities, word reordering is essential for English-Myanmar statistical machine translation.

5. CORPUS CREATION

Corpus creation steps are described in Fig 2. For corpus creation, plain text corpus is used as a resource. For each sentence in the corpus, analysis process is carried out by using Chunk-based Syntax Analyzer [15]. This Syntax Analyzer consists of two components; Chunker and Grammatical Function Tagger. In this analysis process, there are three main steps.

(1) Morpho-lexical analysis

(2) Constituent analysis and

(3) Syntax analysis

Morpho-lexical analysis and constituent analysis are accomplished by the chunker and syntax analysis is the role of grammatical function tagger.

Morpho-lexical analysis contains tokenizing and part-of-speech tagging. Tokenizing splits input text into words by using token marker such as space, punctuation marks. Part-of-speech (POS) tagging marks up the words in the text with their corresponding part-of-speech such as noun, verb, and adjective and so on. For this POS tagging, TreeTagger is used.

Constituent analysis consists of chunking and merging some chunks that are necessary to merge. Chunking is done by generating CFG rules based on part-of-speech (POS) tags.

In syntax analysis, Grammatical function tagger searches the functional relation between chunks based on dependency grammar by using Maximum likelihood Estimation and then identifies the function of each chunk.

By aligning the analyzed text resulted from Analyzer, parallel tagged aligned corpus is created.

Our tagged align corpus format can be seen in Fig 3. As shown in Fig 3, "SUBJ","ACTIVE" , "ADVL" and "OBJ-P" are function tags of each chunk. "NC", "VC" and "PPC" refer the relevant chunk type and "PP", "VBP", "TO" and "NN" are part of speech of each word. The numbers in the parentheses are alignment position of function tags and part of speech tags. The first number before "/" indicates the position of tags in source language and the number after "/" indicates the posi-tion of tags in target language. These are separated by target position with "/". Each chunk is separated by "#".

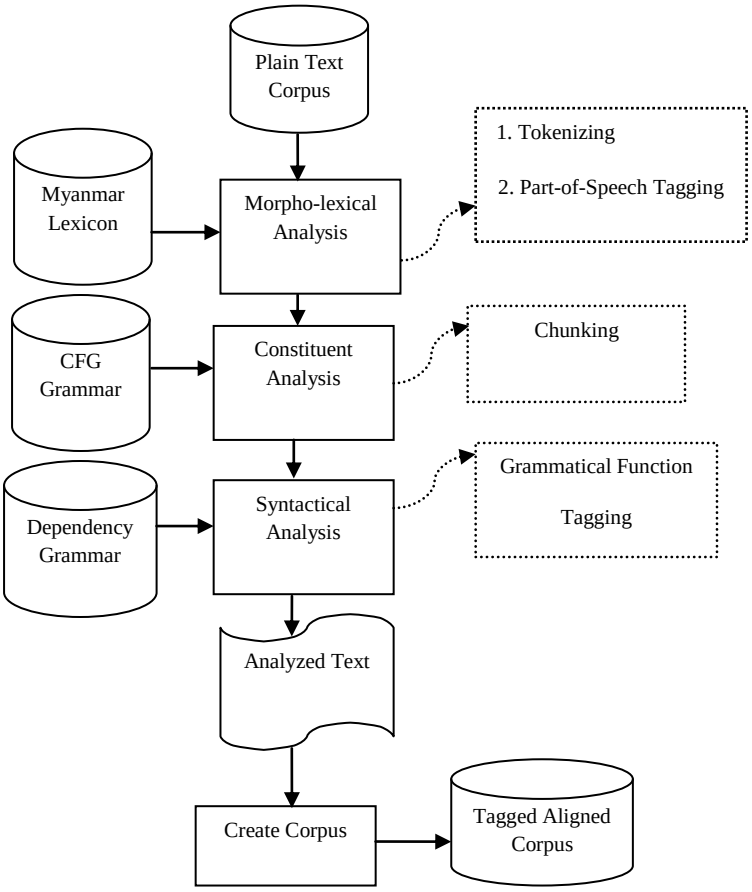


Fig 2: Corpus creation steps be extended both columns

6. RULES EXTRATION ALGORITHMS

By using the linguistic information from the corpus, two kinds of reordering rules are generated automatically. They are function tags-based reordering rules and part-of-speech tags-based reordering rules. The former is generated for using in chunk-level reordering and the latter is for using word-level reordering. They are extracted from corpus using the following rule extraction algorithms.

function rule extraction algorithm

funSeq=NULL //sequence for function tags
aliSeq=NULL //sequence for alignment position

1. Load the sentences from Tagged Aligned Corpus
2. Store all sentences in S .
3. for each sentence $s_i \in S$ do, where i=1,2,3,…k
4. for each chunk $c_i \in C$ do, where i =1,2,3,…k
5. if (k>1)
6. extract f_i for s_i
7. $funSeq \rightarrow funSeq + f_i$
8. extract alignment position a_i
9. $aliSeq \rightarrow aliSeq + a_i$
10. End if//line 5
11. End for//line 4
12. $rule \rightarrow funSeq + \#+ aliSeq$
13. write $rule$
14. End for//line 3
15. End.

pos rule extraction algorithm

posSeq=NULL //sequence of pos tags
aliSeq=NULL //sequence of alignment position

1. Load the sentences from Tagged Aligned Corpus
2. Store all sentences in S .
3. for each sentence $s_i \in S$ do, where i=1,2,3,…k
4. for each chunk $c_i \in C$ in s_i do, where i=1,2,3,…k
5. for each words $w_i \in W$ in c_i where i=1,2,3,…,k
6. if (k>1)
7. extract pos_i for w_i
8. $posSeq \rightarrow posSeq + pos_i$
9. extract alignment position a_i for w_i
10. $aliSeq \rightarrow aliSeq + a_i$
11. End if//line 6
12. End for//line 5
13. rule= $posSeq$ +#+ $aliSeq$
14. End for//line 4
15. write rule
16. End for//line 3

alignment extraction algorithm

Input: AP//Alignment Position Array
Output: rule // for actual alignment position
A=NULL// Array for final alignment position

1. for each ap_i from Array AP do
2. if ($ap_i = ap_{i-1}$) then
3. a_{i-1} =i-1+i+\+ ap_i
4. else
5. a_i =i+\+ ap_i
6. end if
7. end for
8. for each $a_i \in$ A do
9. if $a_i \neq NULL$ then

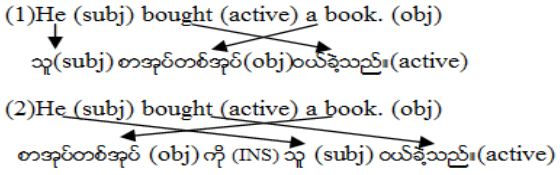
10. rule=rule+ a_i
11. end if
12. end for

7. REORDERING RULES GENERATION

In order to generate reordering rules for English-Myanmar translation, two main cases are needed to consider. In the first case, words can be reorder in several ways if they have different reordering rules. In this case, reordering in several ways does not affect the translation because Myanmar is free chunk order language. In the second case, words are needed to reorder according to the specific translation although they have same pattern with different reordering rules.

According to the first case, reordering can be seen in the following Example (4).

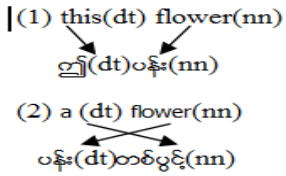
Example (4),



From the above example, the English Sentence "He bought a book." has two different reordering rules and both of these rules make correct translation. Moreover, we can see that word insertion (INS) is needed when this sentence is translated in second way (2).

The second case suggests that, the use of reordering rules mistakenly makes the translation error if they are not reorder over the specified translation. This can be seen in Example (5),

Example (5),



By studying this example, the POS rules composed of determiner and noun have several reordering patterns as shown in (1) and (2). Although they have same pattern (dt,nn), they must be reordered according to the identified translation. If these patterns are not reorder according to the specified patterns, there will be error in translation. To solve this condition, lexical information for determiner is needed to consider in generating reordering rules for this pattern. Therefore, reordering rules are generated automatically using part-of-speech tag, function tags and lexical information.

The generated reordering rule consists of two sides: the left-hand-side (lhs), which is a function tags or POS tags pattern, and the right-hand-side (rhs), which corresponds to a possible reordering of that pattern. Different rules can share the lhs: in such cases, the same pattern can be reordered in more than one way. Rules are weighted, according to statistics extracted from training data. There are two kinds of reordering patterns: function tag-based, which define reordering at the clause and phrase level, and pos tag-based, which defines reordering at the word level. Let us consider the following examples:

- **Rules using function tag**

  -SUBJ, ACTIVE, OBJ#0/0, 1/2, 2/1:7(10)

  -SUBJ, ACTIVE, OBJ#0/1, 1/2, 2/0:3(10)

  -SUBJ, ACTIVE, ADVL#0/0, 1/2, 2/1:10(10)

  -F-SUBJ, ACTIVE, PCOMPL-S, ADVL, OBJ-P# 0+1/3, 2/2, 3/1, 4/0:10(10)

- **Rules using pos tag**

  -the@ DT, NN#0+1/0:10(30)

  -this @DT, NN#0/0, 1/1:10(30)

  -a @DT, NN#0/1, 1/0:10(30)

  -CD, NNS#1/0, 0, 1:10(10)

  -DT, JJ, NN#1/0, 2/1, 0/2:10(10)

In the above rules, "SUBJ","ACTIVE" and OBJ are function tags and "DT",,"NN","CD","NNS" are POS's tags. Therefore, "SUBJ, ACTIVE, OBJ" is function rule pattern and "DT, JJ, NN" is POS rule pattern. The string of numbers after "#" is position of source and target words and source word position is divided by "/" target word position. For example, in the rhs of the third pos rule pattern"1/0, 2/1, 0/ 2", the 1/0 means that the pos tag at the position 1,"JJ" is move to the position 0. In this model, we used array structure to store the position and so the starting index is 0. Moreover, in the function tag rule, the formal subject(F-SUBJ) *"there"* in English is not in the Myanmar Function tag and then it is translated as "ရှိသည်" (ACTIVE) in Myanmar language by combining it into the Function tag(ACTIVE) containing the words *"am ,is are, was, were"*. This means that the two function tags F-SUBJ and ACTIVE are become only ACTIVE and F-SUBJ is dismissed in Myanmar. Therefore, in the third function tag rule described above, the string after #, "0+1/3" means that the words at position "0"and "1" are move together into the position "3".

The sequences "SUBJ, ACTIVE, OBJ" and "DT, JJ, NN" are function and pos rule patterns (p_1^n).The strings of numbers in between the symbols "#" and ":" represent suggested reordering (r_1^n): each integer after "/", r_i represents the new position of (the translation of) p_i. The two numbers after the colon (:) are collected from training data and are respectively the number of times the rhs (reordering suggestion) of the rule has been observed and (inside brackets) the number of occurrences of the rule pattern $count(p_1^n)$. The probability of each reordering suggestion is computed as follows:

$$P(r_1^n / p_1^n) = \frac{count(r_1^n)}{count(p_1^n)} \qquad (1)$$

Moreover, some pos rules composed of determiner (a, an, the, this… etc) have same pattern with different reordering rules

according to the kind of determiner used. To solve the ambiguity problem of these pos rules, lexical information concerned with determiner is added before the rule pattern divided by "@" to obtain the correct order.

8. MARKOV -BASED REORDERING MODEL

In this reordering model we proposed, function tag and pos tag sequences are taken as input. And then, reordering is performed based on first order Markov model by using the alignment probabilities extracted from corresponding function tag and pos tag reordering rules. The tag alignment sequence a_1^K specifies a reordering of source tag sequences into target language tag order. In this way the source language tag sequence u_1^K is reordered into $u_{a_1}, u_{a_2}, \ldots, u_{a_k}$ under the model $P(a_1^k \setminus u_1^k, K, e_1^k)$. The First Order Markov process is firstly defined over tag alignment sequences $a_k \in \{1, 2, ..., K\}$ as can be seen in the following equation and these alignment sequences are obtained from the extracted reordering rules explained above.

$$P(a_1^k \setminus u_1^k, K, e_1^k) = P(a_1^k \setminus u_1^k)$$

$$= P(a_1)\prod_{k=2}^{K} P(a_k \setminus a_{k-1}, u_1^K) \quad (2)$$

The tag alignment sequence is further constrained to be a valid reordering of u_1^K, i.e. the tag alignment sequence is constrained to be a permutation of the set {1, 2, . . . ,K}. The alignment sequence distribution is constructed to assign lower likelihood to tag re-orderings that diverge from the original tag order. Suppose $u_{a_k} = e_l^{l'}$ and $u_{a_{k-1}} = e_m^{m'}$, the Markov chain probabilities is set as shown in equation (3).

$$P(a_k \setminus a_{k-1}, u_1^k) \alpha p_0^{l-m'-1}$$

$$P(a_1 = k) = \frac{1}{K}; k \in \{1, 2, ..., K\} \quad (3)$$

In the above equations, p_0 is a tuning factor and the probabilities $P(a_k \setminus a_{k-1})$ is normalized so that $\sum_{j=1, j \neq a_{k-1}}^{K} P(a_k = j \setminus a_{k-1}) = 1$. This reordering model involves two acceptors un-weighted permutation acceptor $\prod_U$ and weighted permutation acceptor H. Un-weighted permutation acceptor $\prod_U$ contains all reordering of the source language tag sequence u_1^K. For the source sentence "He/ kicked /the ball.", the function tag for this sentence is "subj,active,obj". and there are two reordering examples in the this sentence "He the ball kicked." and "the ball he kicked". So, the first reordering of this tag sequence is "subj,obj,active" and the second reordering of this sequence is "obj,subj,active". Therefore, the function tag alignment sequence for the former reordering example is a1 = 1, a2 =3, a3 = 2and that for the latter reordering example is a1 = 3, a2 =1, a3 = 2.

The second acceptor H assigns alignment probabilities (equation 4) to a given reordering a_1^K of the source tag sequence u_1^K. Therefore, the alignment probability for the first reordering is P(a1 = 1)P(a2 = 3| a1 = 1)P(a3 = 2| a2 = 3) = $0.7 \times 0.7 \times 1 = 0.49$ and the alignment probability for the second reordering is P(a1 = 3)P(a2 = 1| a1 = 3)P(a3 = 2| a2 = 1) = $0.3 \times 0.3 \times 1 = 0.09$. By choosing the maximum alignment probability from the two alignment probabilities, the first reordering example "subj, obj, active" is defined as the optimal reordering rule. After reordering by the optimal reordering rule, phrases relevant to the function tag are extracted. Moreover, part-of-speech tag sequences and their reordering rules are applied to reorder words in each phrase by doing this manner.

9. EXPERIMENTAL RESULTS

9.1 Accuracy of Reordering Rules

The purpose of this experiment is to see how many reordering rules are accurate if they were applied to the test set. The test set is obtained randomly from High School English books. In the test set, lengths of the sentences are between 3 and 20 words. The test set was split into three subsets:

- 1000 simple sentences
- 1000 compound sentences
- 1000 complex sentences

After reordering the test set by the reordering rules, the accuracy values of the reordering rules are collected for each subset on the test set. The accuracy values are given in percentage form. Human evaluation is used for evaluating how accurately the reordering rules are applied to the test set.

Table 1 shows the accuracies of the reordering rules for each subset of English sentences on the test set. The experiment showed that the most common causes of errors of the reordering rules are incorrect tagging from Analyzer and tree-tagger.

Table 1. Accuracy of Reordering Rules

English test subsets	Accuracy
Simple sentences	98.9%
Complex sentences	95.4%
Compound sentences	93.6%

9.2 Evaluation of Proposed reordering travel

The reordering model used in [17] cannot allow all possible reordering for one input sentence and it can allow only maximum phrase jump up to 2. Therefore, it cannot solve long distance reordering. By comparison, proposed reordering model can allow possible reordering rules for one input sentence. By using this reordering model, long distance reordering can be solved easily because the input sentence word length in this system is not limited. Moreover, proposed reordering model can also solve the rule disambiguation problem in part-of-speech rules by using lexical information. In this reordering model, the problem of loss of words caused by tag missing is also solved by generating reordering rules which involve combining two or more tags those have same target position into only one group described in function tag reordering rules pattern 3.

10. CONCLUSION AND FUTURE WORK

To solve the word-order problem between English language and Myanmar language, an approach for automatic reordering rule generation and applying the extracted rule in Markov-based Reordering Model are introduced. In this work, Language Analyzer [15] is applied for function tagging for English language. Moreover, tree-tagger is used for part-of-speech tagging. The proposed approach can work correctly for the rules in the training corpus. Therefore, additional training is also needed for other rules that are not in the training data. Moreover, this work mostly depends on the correct tagging of Analyzer and so better Analyzer will be used for future work. In addition to this, proposed reordering model can be extended for reordering of other language pairs rather than English and Myanmar languages because this model works with the reordering rules based on part-of-speech tags and function tags generated by Chunk-based Analyzer. Therefore, this system can be used for other language pairs that need reordering if the part-of-speech tags and function tags of these languages will be known.

1-SUBJ:NC[I/ကျွန်ုပ်PP(0/0)#(1/3)ACTIVE:VC[go/သွားသည်VBP(0/0)]#(2/2)1-ADVL:PPC[to/သို့TO(0/0)]#(3/1)2-OBJ-P:NC[school/ကျောင်းNN(0/0)]

Fig 3: Parallel tagged aligned corpus

11. REFERENCES

[1] C. Tillmann and H. Ney. 2002. Word reordering and DP beam search for statistical machine translation to appear in Computational Linguistics.

[2] R. Zens and H. Ney. 2003. A comparative study on reordering constraints in statistical machine trans lation. In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, vol ume 1, pages 144–151, Sapporo, Japan.

[3] S. Vogel, F.J. Och, C. Tillmann, S. Nießen, H. Sawaf, and H. Ney. 2000. Statistical methods for machine translation. InW.Wahlster, editor, Verbmobil: Foundations of Speech-to-Speech Translation, pages 377–393. Springer Verlag: Berlin, Heidelberg, New York.

[4] Y.Y. Wang and A. Waibel. 1997. Decoding algorithm in statistical translation. In Proc. 35th Annual Meeting of the Assoc. for Computational Linguistics, pages 366–372, Madrid, Spain, July.

[5] Ei Ei Han and Ni Lar Thein, "Morphological Synthesis For Myanmar Language", Proceeding of International Conference on Internet Information Retrieval, Korea, 2007.

[6] Yaser Al-Onaizan and Kishore Papineno. 2006. Distortion models for statistical machine translation. In Proceedings of the 21st International Conference on Computational Linguistics and the 4th annual meeting of the ACL, pages 529–536, Sydney, Australia.

[7] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra,1996. A maximum entropy approach to natural language processing. Computational Linguistics, 22(1):39.

[8] B. Chen, M. Cettolo, and M. Federico. 2006. Reordering rules for phrase-based statistical machine translation. In Int. Workshop on Spoken Language Translation Evaluation Campaign on Spoken Language Translation, pages 1–15.

[9] M. Popovic and H. Ney. 2006. POS-based word reorderings for statistical machine translation. In Proc. of the 5th Int. Conf. on Language Resources and Evaluation (LREC), page 1278, Genoa, Italy.

[10] L. Shen, A. Sarkar, and F. J. Och. 2004. Discriminative reranking for machine translation. In HLTNAACL 2004: Main Proc., page 177.

[11] C. Tillmann and T. Zhang. 2005. A localized prediction model for statistical machine translation. In Proceedings of the 43rd Annual Meeting of the As-soc. for Computational Linguistics (ACL), pages 557–564, Ann Arbor, MI.

[12] D. Wu. 1996. A polynomial-time algorithm for statistical machine translation. Proc. 34th Annual Meeting of the Assoc. for Computational Linguistics, page 152.

[13] D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Computational Linguistics, 23(3):377.

[14] Y. Zhang, R. Zens, and H. Ney. 2007. Chunk-Level Reordering of Source Language Sentences with Automatically Learned Rules for Statistical Machine

Translation. In Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL): Proceedings of the Workshop on Syntax and Structure in Statistical Translation (SSST), pages 1–8, Rochester, NY.

[15] Myat Thuzar Tun and Ni Lar Thein, " English Syntax Analyzer for English-to-Myanmar Machine Translation", In proceedings of the Fifth International Conference on Computer Application, Myanmar, February, 8-9,2007.

[16] Myat Thuzar Tun, Tin Myat Htwe and Ni Lar Thein, "EMTM: An Effective Language Translation Model", In proceedings of International Conference on Internet Information Retrieval, Korea, November 30, 2005.

[17] Shankar Kumar "Local Phrase Reordering Models for Statistical Machine Translation", Center for Language and Speech Processing, Johns Hopkins University, 3400 North Charles Street, Baltimore, MD 21218, U.S.A.

[18] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer, "The Mathematics of Statistical Machine Translation: Parameter Estimation," Computational Linguistics, vol. 19(2), pp. 263–312, 1993.

[19] Kenji Yamada and Kevin Knight. 2000. A Syntax based Statistical Translation Model. ACL 2000.

[20] Josep M. Crego and Jose B. Marino. 2006. Reordering Experiments for N-Gram-based SMT. In Spoken Language Technology Workshop, pages 242-245, Palm Beach, Aruba.

[21] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation", Association for Computational Linguistics, 2002, pp. 311-318.