Efficient Beam Thresholding for Statistical Machine Translation

Deyi Xiong, Min Zhang, Aiti Aw and Haizhou Li
Human Language Technology
Institute for Infocomm Research
1 Fusionopolis Way, #21-01 Connexis, Singapore 138632
{dyxiong, mzhang, aaiti, hli}@i2r.a-star.edu.sg

Abstract

Beam thresholding is a widely-used pruning approach in decoding algorithms of statistical machine translation. In this paper, we propose two variations on the conventional beam thresholding, both of which speed up the decoding without degrading BLEU score. The first variation is the dynamic beam thresholding, in which the beam threshold varies with the length of source sequences covered by hypotheses. The second one incorporates a language model look-ahead probability into the beam thresholding so that the interaction between a hypothesis and the contexts outside the hypothesis can be captured. Both thresholding methods achieve significant speed improvements when used separately. By combining them together, we obtain a further speedup, which is comparable to that of the cube pruning approach (Chiang, 2007). Experiments also display that the dynamic beam thresholding can further improve the cube pruning.

1 Introduction

Most statistical machine translation (SMT) models are of high complexity, especially when intersected with an m-gram language model. To reduce the search space, practical decoding algorithms make use of pruning techniques. The most widely used methods are beam thresholding (also known as threshold pruning) and histogram pruning. The former retains only hypotheses with a probability larger than a factor α of the currently best hypothesis. The latter limits the number of retained hypotheses to a maximum number b. In this paper, we examine the beam thresholding and focus on two problems of the traditional beam thresholding in SMT decoding. We propose two efficient variations to address these limitations. Our goal is to further reduce the search space explored by the traditional beam thresholding so that we can speed up the decoding.

The first problem is that the traditional beam thresholding uses a fixed probability threshold (α) throughout the whole decoding process. This is not a good approach as the search space is not uniformly distributed during decoding. We have found that a dynamic beam thresholding, in which we adjust the beam threshold according to the length of source sequence covered by hypotheses, can remove 60% of hypotheses explored by the traditional beam thresholding without the risk of decreasing BLEU (Papineni et al., 1996) score.

The second problem with the traditional application of beam thresholding in SMT decoding is that it only uses the probability estimated from inside a hypothesis. It does not look outside the hypothesis: the interaction of the current hypothesis with outside contexts. Consider the case in which a particular hypothesis within the current beam is to be expanded further while after one or more expansion steps, the sub-hypothesis falls outside the beam. The expansion efforts are wasted if we do not generate n-best lists. Therefore can we prune such a hypothesis before we expand it further? The key to the question is to find another probability to measure the future interaction of the current hypothesis with outside contexts so that we can threshold out bad hypotheses as early as possible. We use a language model look-

ahead probability to evaluate the future interaction.

We test the two proposed beam thresholding methods on a CKY-style decoder with beam search, which is developed for BTG-based SMT (Wu, 1996). The CKY-style decoder uses a bottom-up CKY chart parsing algorithm. All hypotheses covering the same span of the source sentence are stored in a cell of the chart. The decoder compares hypotheses to other hypotheses in the same cell and thresholds out bad hypotheses. Experimental results show that two beam thresholding methods lead to significant speedups over the traditional beam thresholding, when used separately. By combining them together, we achieve a further improvement of the decoding speed, which is comparable to that of the cube pruning. When we apply our dynamic beam thresholding to the cube pruning, the decoder obtains a maximum speedup at the same performance level.

The rest of the paper is organized as follows. In Section 2, we present the dynamic beam thresholding with some statistics observations. In Section 3, we describe how to calculate the language model look-ahead probability and incorporate it into the beam thresholding. In Section 4, we compare our work to previous work. Section 5 presents the evaluation results and comparison curves using different beam thresholding methods. Finally, we conclude and discuss our future work in Section 6.

2 Dynamic Beam Thresholding

Generally speaking, if we use a loose beam threshold by retaining as many hypotheses as possible, decoding will be very slow although translation quality remains high. On the other hand, if we use a tight beam threshold, pruning as many hypotheses as possible, we can get a considerable speedup but at the cost of much degraded translation quality. So the question is how we can find an appropriate beam threshold to get the best tradeoff between translation quality and speed. Unfortunately, we are not able to find such an ideal beam threshold since we don't know exactly the search space beforehand.

Most researchers select the beam threshold empirically on the development set and use it constantly for the whole test set. We call this strategy *fixed beam thresholding* (FBT). To guarantee the translation quality, a loose beam threshold is usually used at the cost of slow decoding speed. Using such a constant loose beam threshold on the non-uniformly distributed search space will waste decoding efforts for search areas where the decoder collects more accurate information.

A better strategy is to dynamically adjust the beam threshold based on a hidden variable, which to some extent associates with the search space distribution. Here we define the variable as a ratio r (seq/sent) between the length of source sequence covered by partial hypotheses and that of the whole sentence to be translated. To investigate how we should vary the beam threshold with the length ratio r, we trace the cost[1] difference (best-corr) between the best hypothesis and the correct hypothesis[2] in chart cells on the NIST MT-02 test set (878 sentences, 19.6 words per sentence) which is decoded using a very loose beam threshold[3]. We plot the curve of average best-corr cost difference vs. seq/sent length ratio in Figure 1, which visualizes how wide we should set the beam so that correct hypotheses fall inside the beam.

From this figure, we can observe that in most cases, the longer the source fragment covered by hypotheses, the smaller the cost difference between the correct hypotheses and the best hypotheses. This means that we can safely use a tighter beam threshold for hypotheses covering longer source fragments. It is safe because correct hypotheses are still included in the beam while incorrect hypotheses are pruned as many as possible. However, for hypotheses covering shorter fragments, we should use a looser beam threshold to include all possible candidates for future exploration so that potential candidates can survive to become part of the finally best hypothesis.

According to this observation, we dynamically adjust the beam threshold parameter α as a function

[1]The cost of a hypothesis is the negative logarithm of the translation probability of it. The higher the probability, the lower the cost.

[2]The correct hypothesis is the hypothesis that is part of the best translation generated by the decoder. The best hypothesis is the hypothesis with the least cost in the current span. Note that the best hypothesis is not always the correct hypothesis.

[3]Here we loosened the beam threshold gradually until the BLEU score is not changing. Then we used the last beam threshold we tried.

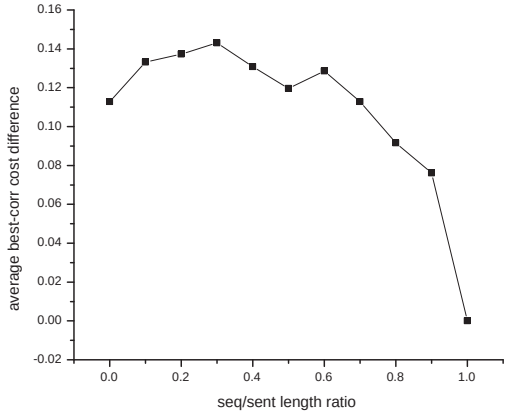

Figure 1: Average Best-corr Cost Difference vs. Seq/sent Length Ratio on the NIST MT-02.

of the length ratio:

$$\alpha = \alpha_0 + (1 - \alpha_0) \cdot r$$

where α_0 is the initial value of the beam threshold parameter which is purposely set to a small value to capture most of the candidates during the early stage of decoding. We call this pruning strategy *dynamic beam thresholding* (DBT). DBT increases the parameter α to tighten the beam when more source words covered. In theory, DBT runs faster than traditional beam thresholding FBT at the same performance level, as our experiments attest.

3 Language Model Look-ahead

In traditional beam thresholding used in SMT decoding, only the probability estimated from inside a partial hypothesis is used. This probability does not give information about the probability of the hypothesis in the context of the complete translation. In A* decoding for SMT (Och et al., 2001; Zhang and Gildea, 2006), different heuristic functions are used to estimate a "future" probability for completing a partial hypothesis. In CKY bottom-up parsing, (Goodman, 1997) introduces a prior probability into the beam thresholding. All of these probabilities are capable of capturing the outside context interaction, to some extent.

In this paper, we discuss the LM look-ahead (LMLA) and examine the question of whether, given the complicated reordering in SMT, the LM look-ahead can obtain a considerable speedup in SMT decoding. The basic idea of the LM look-ahead is to incorporate the language model interaction of the boundary words of a hypothesis and neighboring words outside the hypothesis on the target side into the pruning process as early as possible. Since the exact neighboring words are not available until the partial hypothesis is completed, we obtain potential neighboring words in two steps as follows.

Firstly, for each sequence of source words $s_i...s_j$, we find its most probable translation $T(s_i...s_j)$ with a monotone search through translation options, only considering the translation model and the language model cost. This can be quickly done with dynamic programming, similar to (Koehn, 2004). Then we cache the leftmost/rightmost target boundary words $T^l(s_i...s_j)/T^r(s_i...s_j)$, which both include $m' = min(m-1, |T(s_i...s_j)|)$ (m is the language model order) words[4]. Since there are only $n(n+1)/2$ continuous sequences for n words, the target boundary words for all these sequences can be quickly found and cached before decoding with a very cheap overhead.

Secondly, for a hypothesis H covering a source span $s_i...s_j$, we look up the leftmost/rightmost target boundary words of its neighboring spans: $T^l(s_1...s_{i-1})/T^r(s_1...s_{i-1})$ and $T^l(s_{j+1}...s_n)/T^r(s_{j+1}...s_n)$, which are cached in the first step. Although these boundary words are not exactly adjacent to H since there exist thousands of word reorderings, they still provide context information for language model interaction. We utilize them according to the following two reorderings.

If a straight order is preferred (Fig. 2(a)), the language model look-ahead probability $\pi_s(H)$ can be estimated as follows

$$\pi_s(H) = m\text{-}gram(T^r(s_1...s_{i-1}), H^l) \cdot m\text{-}gram(H^r, T^l(s_{j+1}...s_n))$$

where $H^{l/r}$ are the leftmost/rightmost boundary words of H, which both include $m' = min(m -$

[4]The reason for caching m' words is to keep the same with what we do for each hypothesis, where m' words are also stored on the left/right of the hypothesis for the dynamic programming to compute new m-grams in the CKY algorithm intersected with an m-gram language model (Huang et al., 2005).

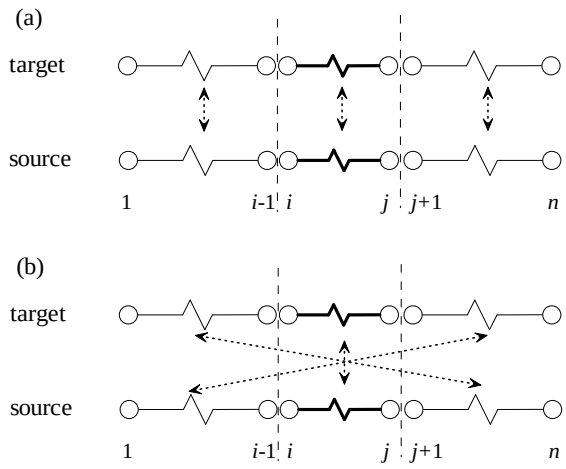

Figure 2: Two Reorderings (straight and inverted) for Language Model Look-Ahead.

$1, |H|)$ words. If an inverted order is preferred (Fig. 2(b)), the language model look-ahead probability $\pi_i(H)$ can be estimated as follows

$$\pi_i(H) = m\text{-}gram(T^r(s_{j+1}...s_n), H^l) \cdot m\text{-}gram(H^r, T^l(s_1...s_{i-1}))$$

Since we don't know which order will be preferred, we take the maximum of the straight and inverted LM look-ahead probability for the hypothesis

$$\pi(H) = max(\pi_s(H), \pi_i(H))$$

The final beam thresholding measure for H when compared to the best hypothesis within the same cell is

$$p(H) = p_{in}(H) \cdot \pi(H)^{\lambda_{LM}}$$

where $p_{in}(H)$ is the probability estimated from inside the hypothesis H, λ_{LM} is the weight of the language model. Note that this probability is only used for the beam thresholding.

4 Comparison to Previous Work

Efficient decoding is of great importance to rapid SMT development and commercial applications. Much of previous work focuses on reducing the overwhelming overhead introduced by the intersection of the m-gram language model and the translation model (phrase-based or syntax-based). This is the fundamental motivation for cube pruning/growing(Chiang, 2007; Huang and Chiang, 2007), and multi-pass decoding approaches(Venugopal et al., 2007; Zhang and Gildea, 2008). Other efforts have been made for A* decoding using search heuristics (Och et al., 2001; Zhang and Gildea, 2006).

The Pharaoh decoder (Koehn, 2004) uses an estimated score of uncovered source sequences as an important component to compare hypotheses. In A* decoding (Och et al., 2001; Zhang and Gildea, 2006), a heuristic function is used to estimate the probability to complete a partial hypothesis. To some extent, both are similar to our LMLA probability. The biggest difference is that we emphasize the effect of the language model interaction on the beam thresholding. We neither use the LMLA probability as a component of priority functions for A* decoding (the former), nor use the estimated score of the full uncovered source sequence for both threshold pruning and histogram pruning (the latter). The Pharaoh-style "future cost" can not provide any discriminative information for our pruning since we compare competing hypotheses within the same cell (This means that they have the same future cost). We remains the same as the Pharaoh decoder to find the most probable path through translation options for source words that are not yet translated. But we go further to take into account the interaction of current hypotheses and the most probable path for not yet translated source sequence.

Moore and Quirk (2007) present two modifications for beam-search decoding, the Pharaoh decoder in particular by improving the future cost estimation and early pruning out next-phrase translations. Their success and the high efficiency of our beam thresholding methods (verified by experiments in the next section) show that there is much room for search space reduction in widely-used beam-search decoding.

5 Experiments

We carried out a series of experiments to examine the effect of our beam thresholding techniques by comparing them with the fixed beam thresholding as well as the cube pruning, and also by combining all these pruning approaches step by step. We tested them on a Chinese-to-English system with a CKY-style decoder.

The system is based on the Bracketing Transduction Grammars (BTG) (Wu, 1997), which uses the BTG lexical rules ($A \rightarrow x/y$) to translate source phrase x into target phrase y and the BTG merging rules ($A \rightarrow [A, A]|\langle A, A\rangle$) to combine two neighboring phrases with a straight or inverted order. The BTG lexical rules are weighted with several features, such as phrase translation, word penalty and language model, in a log-linear form. For the merging rules, a MaxEnt-based reordering model using boundary words of neighboring phrases as features is used to predict the merging order, similar to (Xiong et al., 2006). All the log-linear model weights are tuned on the development set to maximize the BLEU score. A CKY-style decoder is developed to generate the best BTG binary tree for each input sentence, which yields the best translation.

We used the FBIS corpus (7.06M Chinese words and 9.15M English words) as our bilingual training data, from which a MaxEnt-based reordering model was also trained. The 4-gram language model training data (181.1M words) consists of English texts mostly derived from Xinhua section of the English Gigaword corpus. We used the NIST MT-05 as our test set (27.4 words per sentence) and the NIST MT-02 as our development set.

5.1 Dynamic Beam Thresholding

We firstly carried out experiments to compare the dynamic beam thresholding to the fixed beam thresholding. By varying the beam threshold (histogram pruning parameter b set to 40, beam threshold α for FBT and α_0 for DBT varying from 0.9 to 0.05), we plot curves of BLEU score vs. decoding time, and BLEU score vs. search space measured by hypotheses explored per sentence in Figure 3(a) and 3(b) respectively. At most levels of BLEU score, the speedup is about a factor of 2-3. This improvement is highly valuable given that the dynamic beam thresholding can be implemented without effort in most SMT systems. The effect of DBT on the search space reduction is also significant, which can be observed from Figure 3(b). Usually, 60% of hypotheses explored by the fixed beam thresholding can be removed safely by DBT without losing the translation quality measured by BLEU score .

5.2 Language Model Look-ahead

We examined the language model look-ahead pruning method on both FBT and DBT. The curves of BLEU score vs. decoding time and BlEU score vs. search space are plotted in Figure 4(a) and 4(b), under various beam settings in the same fashion of Figure 3. The "+LMLA" item denotes that the probability $p_{in} \cdot \pi^{\lambda_{LM}}$ is used for beam thresholding. In both cases, the language model look-ahead achieves significant speedups in terms of decoding time (Figure 4(a)), a factor of 2-3 for FBT and 1.5-2.5 for DBT. Figure 4(b) displays larger relative speedups in terms of search space size, for instance, a factor of 4-5 for FBT. This is because the language model look-ahead introduces an additional overhead for calculating language model probabilities. Although there exists such an overhead for the language model look-ahead, it still runs faster than that without it. The combination of DBT and LMLA achieves a speedup of a factor 3-5 in terms of decoding time when compared to the traditional fixed beam thresholding.

5.3 Comparison to Cube Pruning

Cube pruning (CP) (Chiang, 2007; Huang and Chiang, 2007) is a state-of-the-art pruning approach which prunes out a large fraction of the possible hypotheses without computing them. We incorporated this technique in our system and compared it with our beam thresholding methods.

Figure 5 plots curves of BLEU score vs. decoding time and search space for cube pruning, fixed beam thresholding, dynamic beam thresholding and its combination with language model look-ahead. The curves of cube pruning and our thresholding method (DBT+LMLA) overlap in most points in both figures, which displays that the combination of the dynamic beam thresholding and the language model look-ahead leads to a speedup improvement comparable to that of cube pruning.

5.4 Combining with Cube Pruning

The cube pruning algorithm uses beam thresholding and histogram pruning to examine those survived. One interesting question is whether we can combine our beam thresholding methods and cube pruning together to obtain a maximum speedup. We carried out

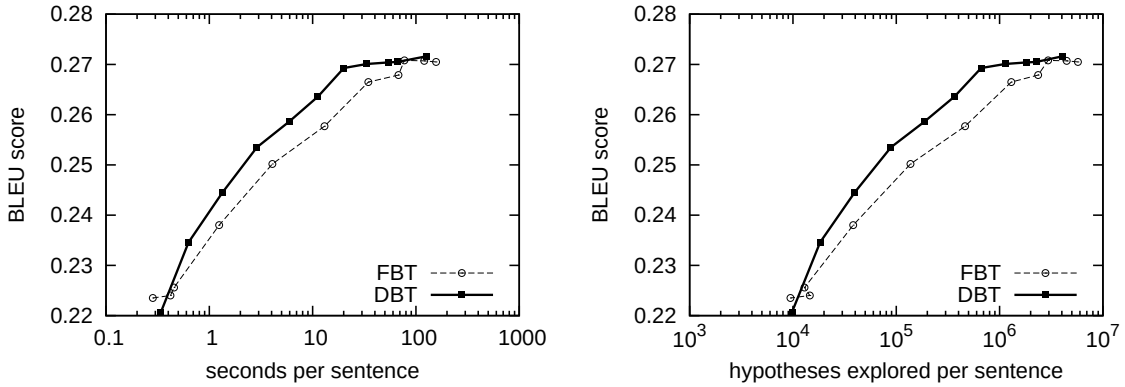

Figure 3: Dynamic Beam Thresholding vs. Fixed Beam Thresholding.

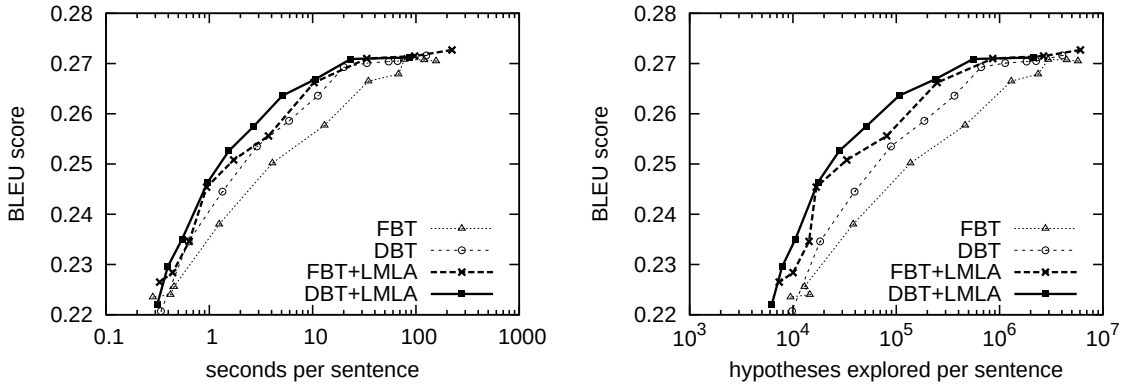

Figure 4: Effect of the Language Model Look-Ahead.

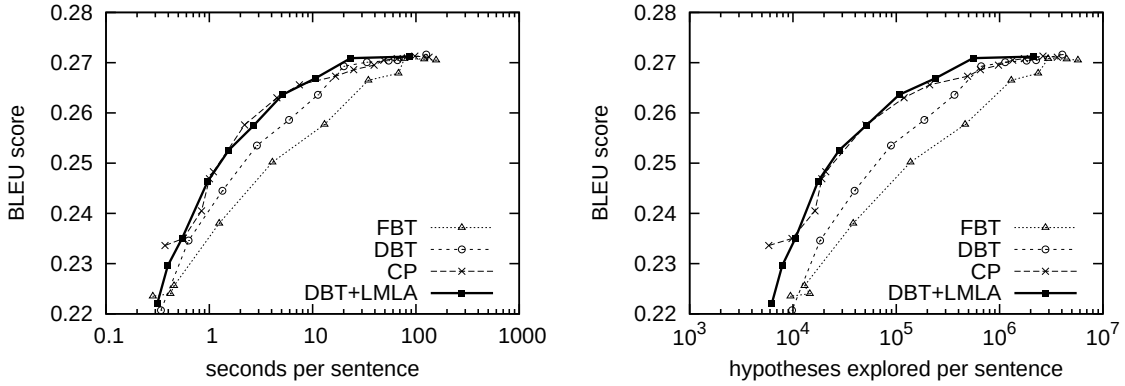

Figure 5: Our Beam Thresholding vs. Cube Pruning.

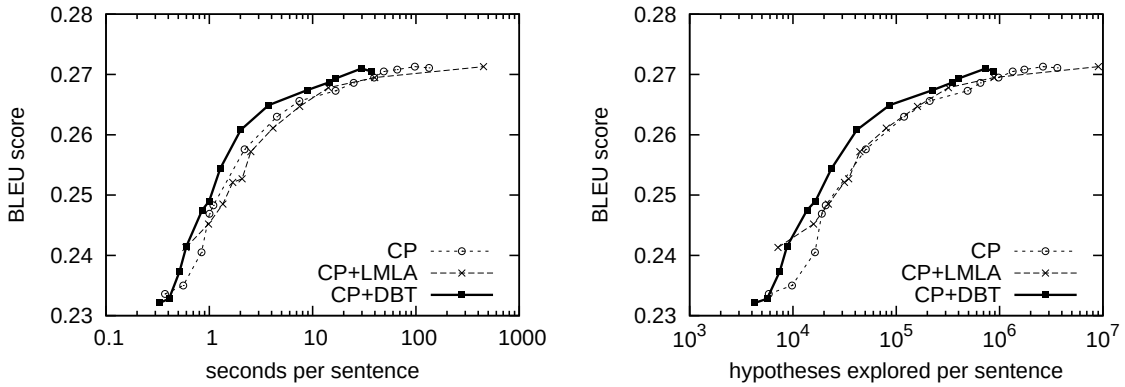

Figure 6: Combining with Cube Pruning.

experiments to investigate this combination. Figure 6(a) and 6(b) plot the comparison curves.

We observe that the dynamic beam thresholding can improve cube pruning, although the speedup is not as significant as that when compared with the fixed beam thresholding. However, the language model look-ahead seems not to be helpful for cube pruning. Figure 6(b) shows that LMLA reduces the search space of cube pruning only at lower performance levels (tight beam settings used) and almost remains the same size of the search space as that of cube pruning without LMLA at higher levels. This indicates that our language model look-ahead is not sophisticated to provide additional information for cube pruning since the hypotheses which escape being pruned by cube pruning also successfully pass the examination of LMLA.

6 Discussion and Future Work

We have presented two efficient beam thresholding methods to speed up beam-search based SMT decoding. We introduce the dynamic beam thresholding based on the observation that most hypotheses covering a longer source sequence can be pruned safely with a tighter beam threshold. We use the language model look-ahead probability to incorporate the language model interaction of outside context with current hypotheses into beam thresholding so that inferior hypotheses can be pruned early. Both methods lead to significant speed improvements when compared to the traditional beam thresholding. We also combine these two methods together, which achieves a speed improvement comparable to that of cube pruning. The dynamic beam thresholding is even helpful for the cube pruning.

Although we discuss and use these thresholding methods on BTG-based SMT, they can be applied to other frameworks, such as standard phrase-based SMT (Koehn et al., 2003) and linguistically syntax-based SMT using CKY algorithms. The dynamic beam thresholding can be integrated into most beam-search based SMT decoders without much efforts. However, the language model look-ahead is much more difficult to be implemented in decoding algorithms using the IBM constraint (Zens et al., 2004) or dealing with gaps. This remains an open problem for future research since reorderings in these decoding algorithms are more complicated.

Although setting the beam threshold as a function is not a new idea, to our best knowledge, we are the first to use it in SMT decoding. The function which we use for the dynamic beam thresholding is actually very straightforward. In the future, we will investigate more sophisticated function for this method. For the look-ahead technique, our future work includes: 1) calculating more accurate language model look-ahead probability; 2) investigating other look-ahead techniques beyond LMLA, such as translation model look-ahead and reordering model look-ahead and their combination; 3) and finally applying the look-ahead technique to other decoding algorithms mentioned above.

References

David Chiang. 2007. Hierarchical Phrase-based Translation. *Computational Linguistics*, 33(2).

Joshua Goodman. 1997. Global Thresholding and Multiple-pass Parsing. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 11-25.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics 2007*.

Liang Huang, Hao Zhang and Daniel Gildea. 2005. Machine Translation as Lexicalized Parsing with Hooks. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT-05)*, Vancouver, BC, Canada.

Robert C. Moore and Chris Quirk. 2007. Faster Beam-search Decoding for Phrasal Statistical Machine Translation. In *Proceedings of MT-SUMMIT* 2007.

Franz Josef Och, Nicola Ueffing, and Herman Ney. 2001. An Efficient A* Search Algorithm for Statistical Machine Translation. In *Proceedings of ACL Workshop on Data-Driven Machine Translation*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics 2002*. Philadelphia, Pennsylvania, USA.

Koehn Philipp, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 127 – 133). Edmonton, Alberta, Canada.

Koehn Philipp. 2004. PHARAOH: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models, User Manual and Description for Version 1.2. USC Information Sciences Institute. http://www.isi.edu/publications/licensedsw/pharaoh/manual-v1.2.ps

Ashish Venugopal, Andreas Zollmann and Stephan Vogel. 2007. An Efficient Two-Pass Approach to Synchronous-CFG Driven Statistical MT. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL)*, Rochester, NY. April 22-27.

Dekai Wu. 1996. A Polynomial-Time Algorithm for Statistical Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics 1996*.

Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377-403.

Deyi Xiong, Qun Liu and Shouxun Lin. 2006. Maximum Entropy Based Phrase Reordering Model for Statistical Machine Translation. In *Proceedings of ACL-COLING 2006*.

R. Zens, H. Ney, T. Watanabe, and E. Sumita. 2004. Reordering Constraints for Phrase-Based Statistical Machine Translation. In *Proceedings of COLING 2004*, Geneva, Switzerland, pp. 205-211.

Hao Zhang and Daniel Gildea. 2006. Efficient Search for Inversion Transduction Grammar. In *Proceedings of ACL-COLING 2006*.

Hao Zhang and Daniel Gildea. 2008. Efficient Multi-pass Decoding for Synchronous Context Free Grammar. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics 2008*.