Faster Beam-Search Decoding for Phrasal Statistical Machine Translation

Robert C. Moore Chris Quirk

Microsoft Research
Redmond, WA 98052, USA
{bobmoore,chrisq}@microsoft.com

Abstract

Pharaoh is a widely-used state-of-the-art decoder for phrasal statistical machine translation. In this paper, we present two modifications to the algorithm used by Pharaoh that together permit much faster decoding without losing translation quality as measured by BLEU score. The first modification improves the estimated translation model score used by Pharaoh to evaluate partial hypotheses, by incorporating an estimate of the distortion penalty to be incurred in translating the rest of the sentence. The second modification uses early pruning of possible next-phrase translations to cut down the overall size of the search space. These modifications enable decoding speed-ups of an order of magnitude or more, with no reduction in the BLEU score of the resulting translations.

1. Introduction

Statistical machine translation (SMT) is widely advocated as a promising approach to achieving translation quality at least comparable to the best rule-based machine translation (RBMT) systems, with greatly reduced effort to adapt to new language pairs and new domains, provided that sufficient parallel training data is available. Such claims are hotly debated, but there is little argument that, to date, SMT systems have been much slower than the best RBMT systems. For example, Language Weaver, currently the only commercial provider of SMT systems, claims to translate 5,000 words per minute per CPU,[1], while SYSTRAN, the market leader in commercial RBMT, claims to translate up to 450 words per second (27,000 words per minute) per CPU.[2]

In this paper, we present two modifications to the algorithm implemented in the widely-used Pharaoh phrasal SMT decoder (Koehn, 2003; Koehn 2004a; Koehn 2004b) that together permit much faster decoding without losing translation quality as measured by the BLEU metric (Papineni et al., 2002). The first modification improves the estimated cost function used by Pharaoh to rank partial hypotheses, by incorporating an estimate of the distortion penalty to be incurred in translating the rest of the sentence. The second modification uses early pruning of possible next-phrase translations to cut down the overall size of the search space. These modifications enable decoding speed-ups of an order of magnitude or more, with no reduction in the BLEU score of the resulting translations.

2. A Phrasal SMT Model

Phrasal SMT, as described by Koehn et al. (2003) translates a source sentence into a target sentence by decomposing the source sentence into a sequence of source phrases, which can be any contiguous sequences of words (or tokens treated as words) in the source sentence. For each source phrase, a target phrase translation is selected, and the target phrases are arranged in some order to produce the complete translation. A set of possible translation candidates created in this way is scored according to a weighted linear combination of feature values, and the highest scoring translation candidate is selected as the translation of the source sentence. Symbolically,

$$\hat{t} = \arg\max_{t,a} \sum_{i=1}^{n} \lambda_i f_i(s, a, t)$$

where s is the input sentence, t is a possible output sentence, and a is a phrasal alignment that specifies how t is constructed from s, and $\hat{t}$ is the selected output sentence. The weights λ_i associated with each feature f_i are tuned to maximize the quality of the translation hypothesis selected by the decoding procedure that computes the $\arg\max$.

We use a fairly standard phrasal SMT model that includes the following features:

- the sum of the log probabilities[3] of each source phrase in the hypothesis given the corresponding target phrase,
- the sum of the log probabilities of each target phrase in the hypothesis given the corresponding source phrase,
- the sum of lexical scores for each source phrase given the corresponding target phrase,
- the sum of lexical scores for each target phrase given the corresponding source phrase,
- the log of the target language model probability for the sequence of target phrases in the hypothesis,
- the total number of words in the target phrases in the hypothesis,
- the total number of source/target phrase pairs composing the hypothesis,

[1] http://www.languageweaver.com/page.asp?intNodeID=862&intPageID=851

[2] http://www.systransoft.com/index/Products/Server-Products/SYSTRAN-Enterprise-Global-Server/System-Requirements

[3] Koehn describes the translation model and the operation of Pharaoh in terms of products of probabilities rather than sums of log probabilities. Our choice is completely equivalent, since the product of a set of probabilities is monotonically related to the corresponding sum of log probabilities.

- a distortion penalty reflecting the degree of divergence of the order of the target phrases from the order of the source phrases.

The probabilities of source phrases given target phrases and target phrases given source phrases are estimated from a word-aligned bilingual corpus. The lexical scores are computed as the log of the unnormalized probability of the Viterbi alignment for a phrase pair under IBM word-translation Model 1 (Brown et al., 1993). For each phrase pair extracted from the word-aligned corpus, the values of these four features are stored in a "phrase table".

The target language model is a trigram model smoothed with bigram and unigram language models, estimated from the target language half of the bilingual training corpus. The distortion penalty is computed as required by the Pharaoh decoder, which we explain in Section 4. We train the feature weights for the overall translation model to maximize the BLEU metric using Och's (2003) minimum-error-rate training procedure.

3. Description of Pharaoh

The Pharaoh decoder uses a beam search to try to find the translation of an input source sentence that has the highest score according to the phrasal SMT model. It creates a set of possible translations, building each target language string from left to right. At each step, it extends a partial translation hypothesis by picking a source phrase covering words that have not yet been translated in that partial hypothesis, and a possible target language translation for that phrase, and appending the target language phrase to the incomplete target language string. The search through the partial hypotheses proceeds in order of the number of source words translated. All the partial hypotheses that cover the same number of source words are compared to each other, and this set is pruned before any members of the set are extended. This core algorithm is presented in Figure 1, taken from Koehn (2003; 2004a; 2004b).

There are at least two key features of Pharaoh that are not revealed at the level of detail presented in Figure 1. First, in addition to beam-search pruning, Pharaoh also performs lossless pruning whenever multiple partial hypotheses agree in

- the source words already translated
- the last two target words produced
- the position of the final word of last source phrase translated

In this situation, any given hypothesis completion will incur the same incremental cost starting from any of these hypotheses; so, the best scoring member of a set of such hypotheses cannot be surpassed by any other in the set. Pharaoh keeps only the highest scoring such hypothesis in the beam search, although the others are saved in case multiple translation hypotheses are desired.

The second key feature of Pharaoh not revealed in Figure 1 is how Pharaoh computes the partial hypothesis scores used for pruning. The score that Pharaoh uses to compare competing hypotheses consists of two components, an exact score for the part of the translation that the hypothesis is committed to, and an estimated score for the portion of the source sentence remaining to be translated.

To compute the estimated scores, before starting to translate a sentence Pharaoh finds the best possible estimated phrase pair score for each source phrase in the phrase table that matches some contigous subsequence of the input source sentence. An estimated score for every contiguous subsequence of the input is then computed by finding the sequence of source phrases covering the input subsequence with the highest sum of estimated scores. This is computed in $O(n^2)$ time by dynamic programming.

The estimated score for each phrase pair is computed as the sum of the feature values in the phrase table for that phrase pair, along with the target word count and phrase pair count, plus an approximate target language score for the target phrase in the pair, all weighted by the corresponding translation model weights. The target language model score can only be approximated, because we don't yet know what the language model context will be if the phrase pair in question is actually used to complete the translation of the input source sentence. The approximate target language model score therefore uses the unigram probabilty estimate for the first word of the target phrase, the bigram probability estimate for the second word of the target phrase, and the full trigram probabililty estimate only for the third and subsequent words of the target phrase.

4. Distortion Penalty Estimation

Our first improvement to the algorithm implemented by Pharaoh is to incorporate an estimate of the distortion penalty yet to be incurred into the estimated score for the portion of the source sentence remaining to be translated. Such an estimate is notably absent from the score used by Pharaoh for pruning sets of competing partial hypotheses.

The value of the distortion penalty feature used by Pharaoh is the sum of the distances between source phrases whose target phrase translations are adjacent in the target language string. Specifically, Koehn (2004a) defines the incremental distortion penalty for each pair of adjacent target phrases as:

> $d =$ abs(last word position of previously translated phrase + 1 - first word position of newly translated phrase)

We can break this down into two simple cases using the following defnitions:

- Δd is the distortion penalty increment for a partial hypothesis, relative to the immediate predecessor it was formed from by adding a translation for the source phrase S.
- S' is the last source phrase translated in the immediate predecessor,
- $L(S)$ and $L(S')$ are the length in words of S and S', respectively.
- $D(S, S')$ is the number of words between S and S'.

```
initialize hypothesisStack[0 .. nf];
create initial hypothesis hyp_init;
add to stack hypothesisStack[0];
for i=0 to nf-1:
  for each hyp in hypothesisStack[i]:
    for each new_hyp that can be derived from hyp:
      nf[new_hyp] = number of foreign words covered by new_hyp;
      add new_hyp to hypothesisStack[nf[new_hyp]];
      prune hypothesisStack[nf[new_hyp]];
find best hypothesis best_hyp in hypothesisStack[nf];
output best path that leads to best_hyp;
```

Figure 1: Pharaoh beam search algorithm

In terms of these symbols, the two cases are:

- If S is to the right of S',

$$\Delta d = D(S, S')$$

- If S is to the left of S',

$$\Delta d = D(S, S') + L(S) + L(S')$$

We estimate the distortion penalty yet to be incurred by a partial hypothesis to be the minimum possible additional distortion penalty, given the source words translated so far and the final word position of the last source phrase translated. It is easy to prove by induction on the number of untranslated words that, for any partial hypothesis, the minimum additional distortion penalty is that produced by picking as the next source phrase to translate one that begins with the left-most untranslated source word and proceeding left-to-right covering all the remaining untranslated source words in order.[4]

The computationally simplest way to take this minimum possible additional distortion penalty into account is just to fold it into the distortion penalty as we incrementally accumulate it. To describe this modification, we use the previous definitions, and we also define S'' to be the longest fully-translated initial segement of the source sentence prior to translating S, and $D(S, S'')$ to be the number of words between S and S''. Note that S'' immediately precedes the left-most untranslated word. The computation of the modified Δd can be broken down four cases:

- If S is adjacent to S'',

$$\Delta d = 0$$

- Otherwise, if S is to the left of S',

$$\Delta d = 2L(S)$$

- Otherwise, if S' is a subsequence of S''

$$\Delta d = 2(D(S, S'') + L(S))$$

- Otherwise,

$$\Delta d = 2(D(S, S') + L(S))$$

This modified distortion penalty can be shown to have the same value as that used in Pharaoh, over an entire, completed translation hypothesis,[5] but it "front-loads" the accumulation of the distortion penalty. For example, if we skip over a single word towards the beginning of a source sentence and then translate a number of phrases monotonically, the distortion penalty as calculated by Pharaoh will be 1, until we finally jump back to translate the skipped word. Using our modified distortion penalty, as we translate more and more words beyond the skipped word, we accumulate a progressively larger distortion penalty, because we know that we must eventually go back to translate the skipped word.

5. Early Pruning

Our second modification of the Pharaoh algorithm addresses the sixth line in Figure 1, which says:

```
for each new_hyp that can be
derived from hyp:
```

This means that (subject to static phrase table and distortion limits discussed later) every possible translation of every possible next phrase (not involving words already translated) will be considered as an extension to a given partial hypothesis. No pruning of any possible extension is considered until an estimated score for the extension has been computed as described in Section 3.

Recall that in order to have an estimated score for each possible subsequence of the input source ready, we have precomputed an estimated score for each possible phrase translation that includes all aspects of the translation model, except for the distortion penalty and a language model score adjustment that replaces the unigram and bigram scores for the first two words of the target phrase with their full trigram scores. We can prune the search earlier than Pharaoh does, in a way that lets us eliminate multiple possible next source phrases and multiple possible translations for source phrases not eliminated, without even examining them, provided we are willing to forgo having the pruning take into account the language model score adjustment for the last phrase translated in a given partial hypothesis.

[4] The proof requires assuming that a distortion penalty increment is incurred if the last source phrase translated does not occur at the end the source sentence. None of the written descriptions of Pharaoh state whether this is the case.

[5] This requires making the same assumption about a sentence-final distortion penalty noted eariler.

We introduce additional points at which the search is pruned by comparing each partial hypothesis to its possible extensions, and stopping the search for extensions when the estimated score of the extensions (before making the language model score adjustment) is worse than the estimated score of the partial hypothesis we are extending by more than a fixed early pruning threshold. We do this in addition to performing Pharaoh's pruning step, which compares all partial hypotheses that cover the same number of source words using an estimated score that does include the language model score adjustment.

Several observations help us organize the search through possible extensions to a given partial hypothesis. First, for any given starting point for the next phrase to be translated, a phrase of length 1 will produce the minimum additional distortion penalty. Second, the minimum additional distortion penalty given a starting point never decreases as we move the starting point from left to right. Third, for any given starting point, the additional distortion penalty never decreases as we increase the length of the source phrase to be translated.

With these facts in mind, we search from left to right through the possible starting positions for the next source phrase to translate. For each position, we compute the minimum additional distortion penalty for a source phrase starting at that position. If we find a possible starting position such that the minimum additional distortion penalty (weighted by the corresponding translation model weight) is greater than our early pruning threshold, we stop looking for possible next source phrases to translate, because all the ones that we have not considered will also have additional distortion penalties greater than the threshold.

For each possible starting position that passes this test, we search through possible ending positions from left to right. If we find a possible ending position such that the weighted additional distortion penalty for the phrase spanning the starting and ending positions is greater than the threshold, we stop looking for possible ending positions for that starting position, because all the ones that we have not considered will also have weighted additional distortion penalties greater than the threshold.

Each starting and ending position pair that passes this test defines a possible next source phrase to translate. For each such source phrase that has entries in the phrase table, we search through its possible translations, from best scoring to worst scoring, having sorted the phrase table in this way offline. For each translation, we compute the estimated score of the resulting partial hypothesis, taking into account everything except the language model score adjustment. If the difference between this estimated score and that of the hypothesis we are extending is greater than the early pruning threshold, we stop looking at possible translations for this source phrase, because all the translations that we have not considered will also yield estimated score differences greater than the threshold.

6. Evaluation

We have carried out experiments evaluating three different algorithms: the original Pharaoh algorithm, the Pharaoh algorithm plus distortion penalty estimation, and the Pharaoh algorithm plus distortion penalty estimation and early pruning.[6] In order to measure the effects of our modifications to the Pharaoh algorithm as accurately as possible, we have reimplemented the algorithm described by Koehn in such a way that the three systems are identical except for the algorithmic differences under evaluation.

We have implemented all three algorithms in Perl, which is a byte-code interpreted language, so the absolute time measurements are slower that what would be expected from implementations that compile to native machine code. The relative timings should still be indicative of the relative efficiency of the algorithms, however. Moreover, we also report a measure of the search space explored that should be independent of other implementation details: the number of partial hypotheses evaluated per source word.

Since decoding effort depends on several pruning parameters, a fair evaluation of the Pharaoh algorithm and its variants requires testing many combinations of settings for these parameters. There are four main pruning parameters:

- T-table threshold: the maximum difference in estimated score between the best translation and the worst translation in the phrase table for a given source phrase
- Beam threshold: the maximum difference in estimated score between the best partial hypothesis and the worst partial hypothesis retained for a given number of source words covered
- T-table limit: the maximum number of translations in the phrase table for a given source phrase
- Beam limit: the maximum number of partial hypotheses retained for a given number of source words covered

Below, when we discuss particular vectors of pruning parameter settings, we will give them in the order above. In Koehn's implemetation of Pharaoh, the two threshold parameters are expresssed as ratios of probabilities. Our threshold parameters have exactly the same effect, but at different specific settings.

A fifth parameter that can be viewed as a pruning parameter is the distortion limit, which restricts the maximum distortion increment permitted between source phrases whose translations are adjacent in the output target sentence. We prefer to view this as a model parameter, however, because setting it to an optimum value usually improves translation quality over leaving it unrestricted. For all the experiments reported here, we set the distortion limit to 5. This seems to be within the range of typical settings for using Pharaoh, and it also appeared in informal experimentation that for settings greater than 5, translation quality started to decline markedly given our data and models.

[6] Without distortion penalty estimation, early pruning can lead to failure to find a translation, because it is possible for all extensions to fail the early pruning test for all partial hypotheses within the beam. This cannot happen if our distortion penalty estimate is used, because in that case, at least one extension of each partial hypothesis will have an estimated score (without the language model score adjustment) identical to the estimated score of the hypothesis it extends.

The exact version of distortion limit we implemented allows one more word in the backward direction than the forward direction (otherwise a distortion limit of 1 would allow no distortion at all, since the minimum cost of a backwards jump is 2), and we also disallowed configurations where jumping back to the left-most untranslated word would violate the distortion limit.[7] For all three algorithms tested, we used Koehn's definition of distortion for applying the distortion limit, even when the modified version was used in the beam search.

For the decoder with early pruning, the early pruning threshold might also be treated as an independent parameter. However, there is a close connection between the early pruning threshold and the T-table threshold. If the T-table threshold is increased beyond the early pruning threshold, none of the additional phrase table entries will ever survive early pruning. We therefore always used the same setting for the T-table and early pruning thresholds.

We performed a hill-climbing search for combinations of settings of the four pruning parameters that produce good trade-offs of decoding time vs. BLEU score. We tried five different settings for each of the pruning paramenters; 0.5, 1.0, 1.5, 2.0, and 2.5 for the threshold parameters, and 5, 10, 15, 20, and 25 for the limit paramters. For the modified algorithms, this appeared to be a sufficient range to find the operating points that produced the highest BLEU score, but the baseline Pharaoh algorithm seemed to require a greater beam limit to avoid losing translation quality. So, we also tested beam limits of 30, 35, 40, 45, 50, 60, 75, and 100 with the baseline system, with the other parameters set to selected combinations of settings that produce good time-quality tradeoffs at lower beam limits.

Our training and test data came from an English-French bilingual corpus of Canadian Hansards parliamentary proceedings supplied for the bilingual word alignment workshop held at HLT-NAACL 2003 (Mihalcea & Pedersen, 2003). Automatic sentence alignment of this data was provided by Ulrich Germann. We used 500,000 sentences pairs from this corpus for training both the phrase translation models and IBM Model 1 lexical scores. This training data was word-aligned using a state-of-the-art word-alignment method (Moore et al., 2006), and all pairs of phrases up to 7 words in length were extracted and their translation probabilities estimated using the method described by Koehn et al. (2003). A separate set of 500 sentence pairs was used to train the translation model weights, and an additional 2000 sentence pairs were used for test data.

For each combination of pruning parameter settings tested, we measured the time required for decoding in milliseconds per word, the size of the search space in partial hypotheses evaluated per word, and the BLEU score of the resulting translations on a scale of 0–100 (BLEU[%]). The scatter plots in Figures 2 and 3 display BLEU score vs. decoding time, and BLEU score vs. search space.

Since the scatter plots for the three algorithms overlap somewhat, we have highlighted the limits of each algorithm as defined by the upper convex hull of points for that algorithm. This picks out what are arguably the best points in terms of the trade off between decoding effort and translation quality as measured by our metrics. Note that the horizontal axes are presented logarithmically to make the differences in decoding effort clear at all scales.

From Figures 2 and 3, we see that all three algorithms eventually produce the same highest value for the BLEU score (30.22 BLEU[%]), but the algorithm that employs distortion penalty estimation does so with much less decoding effort than the baseline algorithm, and the algorithm that uses both distortion penalty estimation and early pruning requires even less decoding effort. This is true whether decoding effort is measured in terms of time or search space. Indeed, for each of the three algorithms, the correlation between the decoding time and the number of partial hypotheses evaluated is greater than 0.99. The pruning parameter vectors that produced the highest BLEU score were $(1.5, 1.0, 20, 10)$ for both of the modified algorithms, and $(1.5, 1.0, 20, 75)$ for the Pharaoh baseline algorithm.

Comparing the decoding times needed to obtain the highest BLEU score, the Pharaoh algorithm takes 106.0 milliseconds per word, adding distortion penalty estimation brings this down to 34.2 milliseconds per word, and adding early pruning to that reduces the time to 9.02 milliseconds per word. If we are willing to accept a score 0.02 BLEU[%] lower (30.20), the Pharaoh algorithm takes 38.6 milliseconds per word, adding distortion penalty estimation yields a time of 14.7 milliseconds per word, and adding early pruning yields 3.59 milliseconds per word.

The ratio of decoding times for the Pharaoh algorithm compared to that for the best system is 11.8 to 1 to reach the highest BLEU score, and 10.8 to 1 to reach a score of 30.20 BLEU[%]. The ratios of search space for the Pharaoh algorithm compared to that for the best system are even more dramatic. The ratio to reach the highest BLEU score is 18.4, and the ratio to reach a score of 30.20 BLEU[%] is 20.5.

We can cast some light on where the speed-ups are coming from by comparing the algorithms at the same pruning settings, looking at differences in BLEU score and decoding time ratios. Comparing the original Pharaoh algorithm to the Pharaoh algorithm plus distortion penalty estimation, up to a beam limit of 25, the decoding time ratio for the same pruning settings ranged from 0.88 to 1.19 — very little difference. However, the difference in BLEU score when distortion penalty estimation was used ranged from $+0.17$ BLEU[%] to $+0.68$ BLEU[%]. Thus the speed-up from distortion penalty estimation came from being able to obtain a given BLEU score at much tighter pruning settings than were necessary with the baseline alogorithm, rather than speeding up decoding at a given combination of pruning settings.

Comparing distortion penalty estimation to distortion penalty estimation plus early pruning, the difference in BLEU score at the same pruning settings ranged only from -0.024 BLEU[%] to $+0.016$ BLEU[%]. The decoding time ratio, however, ranged from 1.37 to 6.36 times faster for the decoder with early pruning. Thus early pruning makes almost no difference in BLEU score at a given combination of pruning settings, but it makes decoding up to six times faster.

[7] We believe this is more-or-less how the distortion limit works in Koehn's implementation of Phraraoh, but the published descriptions do not go into these details.

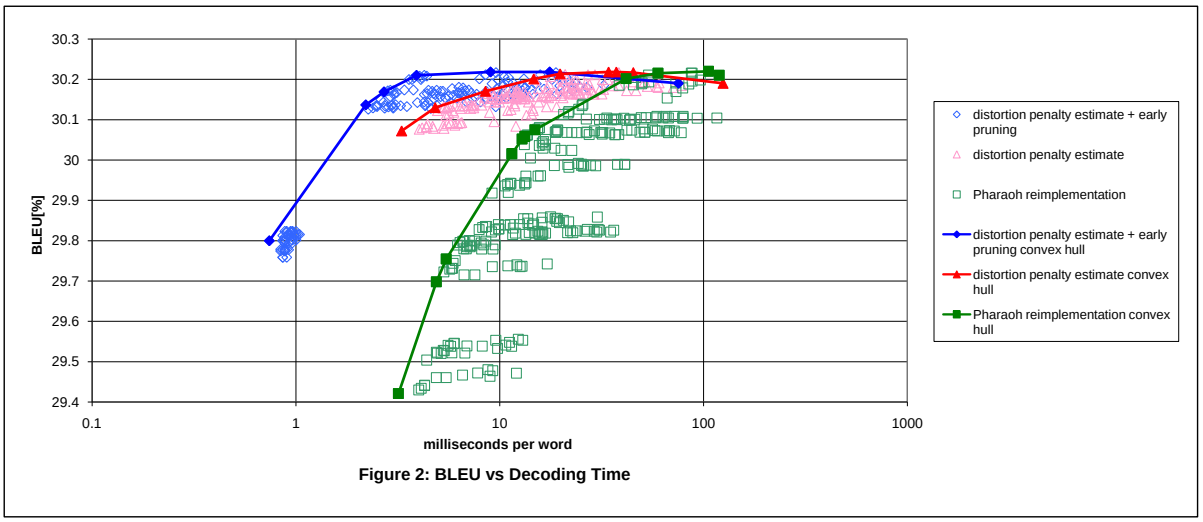

Figure 2: BLEU vs Decoding Time

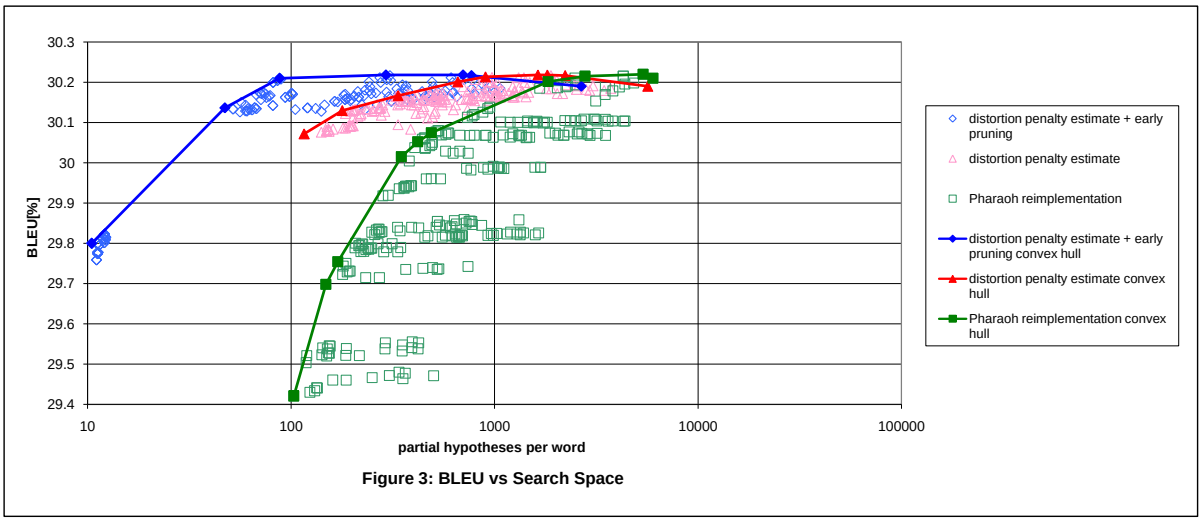

Figure 3: BLEU vs Search Space

7. Conclusions

We have demonstrated two fairly simple modifications to the Pharaoh decoding algorithm that result in decoding speed-ups of more than an order of magnitude. The decoding speed of 3.59 milliseconds per word produced by accepting a score lower than the best obtainable by just 0.02 BLEU[%] is equivalent to translating more than 16,700 words per minute. Since this was obtained using an implementation in Perl, there seems little doubt that we could easily obtain translation speeds comparable to those of commercial RBMT systems, simply by coding the algorithm in a language that compiles to native machine code.

References

Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., & Mercer, R.L. (1993). The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics, 19(2):263–311.

Koehn, P. (2003). Noun Phrase Translation. PhD Dissertation, Computer Science, University of Southern California, Los Angeles, California, USA.

Koehn, P. (2004a). 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/licensed-sw/pharaoh/manual-v1.2.ps

Koehn, P. (2004b). Pharaoh: A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In Proceedings of AMTA-2004, The 6th Conference of the Association for Machine Translation in the Americas (pp. 115–124). Washington, DC, USA.

Koehn P., Och, F.J., & Marcu, D. (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.

Moore, R.C, Yih, W., & Bode, A. (2006). Improved Discriminative Bilingual Word Alignment. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (pp. 513-520). Sydney, Australia.

Mihalcea, R. & Pedersen, T. (2003). An Evaluation Exercise for Word Alignment. In Proceedings of the HLT-

NAACL 2003 Workshop, Building and Using Parallel Texts: Data Driven Machine Translation and Beyond (pp. 1–6). Edmonton, Alberta, Canada.

Och, F.J. (2003). Minimum Error Rate Training in Statistical Machine Translation. In Proceedings of the 41st Annual Meeting of the ACL (pp. 160–167). Sapporo, Japan.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (pp. 311–318). Philadelphia, Pennsylvania, USA.