A Performance Study of Cube Pruning for Large-Scale Hierarchical Machine Translation

Matthias Huck[1] and **David Vilar**[2] and **Markus Freitag**[1] and **Hermann Ney**[1]

[1] Human Language Technology and Pattern Recognition Group, RWTH Aachen University
D-52056 Aachen, Germany
<surname>@cs.rwth-aachen.de

[2] DFKI GmbH
Alt-Moabit 91c
D-10559 Berlin, Germany
david.vilar@dfki.de

Abstract

In this paper, we empirically investigate the impact of critical configuration parameters in the popular cube pruning algorithm for decoding in hierarchical statistical machine translation. Specifically, we study how the choice of the k-best generation size affects translation quality and resource requirements in hierarchical search. We furthermore examine the influence of two different granularities of hypothesis recombination. Our experiments are conducted on the large-scale Chinese→English and Arabic→English NIST translation tasks. Besides standard hierarchical grammars, we also explore search with restricted recursion depth of hierarchical rules based on shallow-1 grammars.

1 Introduction

Cube pruning (Chiang, 2007) is a widely used search strategy in state-of-the-art hierarchical decoders. Some alternatives and extensions to the classical algorithm as proposed by David Chiang have been presented in the literature since, e.g. cube growing (Huang and Chiang, 2007), lattice-based hierarchical translation (Iglesias et al., 2009b; de Gispert et al., 2010), and source cardinality synchronous cube pruning (Vilar and Ney, 2012). Standard cube pruning remains the commonly adopted decoding procedure in hierarchical machine translation research at the moment, though. The algorithm has meanwhile been implemented in many publicly available toolkits, as for example in Moses (Koehn et al., 2007; Hoang et al., 2009), Joshua (Li et al., 2009a), Jane (Vilar et al., 2010), cdec (Dyer et al., 2010), Kriya (Sankaran et al., 2012), and NiuTrans (Xiao et al., 2012). While the plain hierarchical approach to machine translation (MT) is only formally syntax-based, cube pruning can also be utilized for decoding with syntactically or semantically enhanced models, for instance those by Venugopal et al. (2009), Shen et al. (2010), Xie et al. (2011), Almaghout et al. (2012), Li et al. (2012), Williams and Koehn (2012), or Baker et al. (2010).

Here, we look into the following key aspects of hierarchical phrase-based translation with cube pruning:

- Deep vs. shallow grammar.
- k-best generation size.
- Hypothesis recombination scheme.

We conduct a comparative study of all combinations of these three factors in hierarchical decoding and present detailed experimental analyses with respect to translation quality and search efficiency. We focus on two tasks which are of particular interest to the research community: the Chinese→English and Arabic→English NIST OpenMT translation tasks.

The paper is structured as follows: We briefly outline some important related work in the following section. We subsequently give a summary of the grammars used in hierarchical phrase-based translation, including a presentation of the difference between a deep and a shallow-1 grammar (Section 3). Essential aspects of hierarchical search with the cube pruning algorithm are explained in Section 4. We show how the k-best generation size is defined (we apply the limit without counting recombined

Proceedings of the 7th Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 29–38,
Atlanta, Georgia, 13 June 2013.

candidates), and we present the two different hypothesis recombination schemes (*recombination T* and *recombination LM*). Our empirical investigations and findings constitute the major part of this work: In Section 5, we first accurately describe our setup, then conduct a number of comparative experiments with varied parameters on the two translation tasks, and finally analyze and discuss the results. We conclude the paper in Section 6.

2 Related Work

Hierarchical phrase-based translation (HPBT) was first proposed by Chiang (2005). Chiang also introduced the cube pruning algorithm for hierarchical search (Chiang, 2007). It is basically an adaptation of one of the k-best parsing algorithms by Huang and Chiang (2005). Good descriptions of the cube pruning implementation in the Joshua decoder have been provided by Li and Khudanpur (2008) and Li et al. (2009b). Xu and Koehn (2012) implemented hierarchical search with the cube growing algorithm in Moses and compared its performance to Moses' cube pruning implementation. Heafield et al. recently developed techniques to speed up hierarchical search by means of an improved language model integration (Heafield et al., 2011; Heafield et al., 2012; Heafield et al., 2013).

3 Probabilistic SCFGs for HPBT

In hierarchical phrase-based translation, a probabilistic synchronous context-free grammar (SCFG) is induced from a bilingual text. In addition to continuous *lexical* phrases, *hierarchical* phrases with usually up to two gaps are extracted from the word-aligned parallel training data.

Deep grammar. The non-terminal set of a standard hierarchical grammar comprises two symbols which are shared by source and target: the initial symbol S and one generic non-terminal symbol X. Extracted rules of a standard hierarchical grammar are of the form $X \rightarrow \langle \alpha, \beta, \sim \rangle$ where $\langle \alpha, \beta \rangle$ is a bilingual phrase pair that may contain X, i.e. $\alpha \in (\{X\} \cup V_F)^+$ and $\beta \in (\{X\} \cup V_E)^+$, where V_F and V_E are the source and target vocabulary, respectively. The $\sim$ relation denotes a one-to-one correspondence between the non-terminals in α and in β. A non-lexicalized initial rule and a special glue rule complete the grammar. We denote standard hierarchical grammars as *deep* grammars here.

Shallow-1 grammar. Iglesias et al. (2009a) propose a limitation of the recursion depth for hierarchical rules with shallow grammars. In a *shallow-1* grammar, the generic non-terminal X of the standard hierarchical approach is replaced by two distinct non-terminals XH and XP. By changing the left-hand sides of the rules, lexical phrases are allowed to be derived from XP only, hierarchical phrases from XH only. On all right-hand sides of hierarchical rules, the X is replaced by XP. Gaps within hierarchical phrases can thus be filled with continuous lexical phrases only, not with hierarchical phrases. The initial and glue rules are adjusted accordingly.

4 Hierarchical Search with Cube Pruning

Hierarchical search is typically carried out with a parsing-based procedure. The parsing algorithm is extended to handle translation candidates and to incorporate language model scores via cube pruning.

The cube pruning algorithm. Cube pruning operates on a hypergraph which represents the whole parsing space. This hypergraph is built employing a customized version of the CYK+ parsing algorithm (Chappelier and Rajman, 1998). Given the hypergraph, cube pruning expands at most k derivations at each hypernode.[1] The pseudocode of the k-best generation step of the cube pruning algorithm is shown in Figure 1. This function is called in bottom-up topological order for all hypernodes. A heap of active derivations A is maintained. A initially contains the first-best derivations for each incoming hyperedge (line 1). Active derivations are processed in a loop (line 3) until a limit k is reached or A is empty. If a candidate derivation d is recombinable, the RECOMBINE auxiliary function recombines it and returns `true`; otherwise (for non-recombinable candidates) RECOMBINE returns `false`. Non-recombinable candidates are appended to the list D of k-best derivations (line 6). This list will be sorted before the function terminates

[1] The hypergraph on which cube pruning operates can be constructed based on other techniques, such as tree automata, but CYK+ parsing is the dominant approach.

(line 8). The PUSHSUCC auxiliary function (line 7) updates A with the next best derivations following d along the hyperedge. PUSHSUCC determines the cube order by processing adjacent derivations in a specific sequence (of predecessor hypernodes along the hyperedge and phrase translation options).[2]

k-best generation size. Candidate derivations are generated by cube pruning best-first along the incoming hyperedges. A problem results from the language model integration, though: As soon as language model context is considered, monotonicity properties of the derivation cost can no longer be guaranteed. Thus, even for single-best translation, k-best derivations are collected to a buffer in a beam search manner and finally sorted according to their cost. The k-best generation size is consequently a crucial parameter to the cube pruning algorithm.

Hypothesis recombination. Partial hypotheses with states that are indistinguishable from each other are recombined during search. We define two notions of when to consider two derivations as indistinguishable, and thus when to recombine them:

- **Recombination T.** The T recombination scheme recombines derivations that produce identical translations.
- **Recombination LM.** The LM recombination scheme recombines derivations with identical language model context.

Recombination is conducted within the loop of the k-best generation step of cube pruning. Recombined derivations do not increment the generation count; the k-best generation limit is thus effectively applied after recombination.[3] In general, more phrase translation candidates per hypernode are being considered (and need to be rated with the language model) in the *recombination LM* scheme compared to the *recombination T* scheme. The more partial hypotheses can be recombined, the more iterations of the inner code block of the k-best generation loop are possible. The same internal k-best generation size results in a larger search space for *recombination LM*. We will examine how the overall number of loop iterations relates to the k-best generation limit. By measuring the number of derivations as well as the number of recombination operations on our test sets, we will be able to give an insight into how large the fraction of recombinable candidates is for different configurations.

Input: a hypernode and the size k of the k-best list
Output: D, a list with the k-best derivations

```
1 let A ← heap({(e, 1_|e|) | e ∈ incoming edges)})
2 let D ← [ ]
3 while |A| > 0 ∧ |D| < k do
4     d ← pop(A)
5     if not RECOMBINE(D, d) then
6         D ← D ++ [d]
7     PUSHSUCC(d, A)
8 sort D
```

Figure 1: k-best generation with the cube pruning algorithm.

5 Experiments

We conduct experiments which evaluate performance in terms of both translation quality and computational efficiency, i.e. translation speed and memory consumption, for combinations of deep or shallow-1 grammars with the two hypothesis recombination schemes and an exhaustive range of k-best generation size settings. Empirical results are presented on the Chinese→English and Arabic→English 2008 NIST tasks (NIST, 2008).

5.1 Experimental Setup

We work with parallel training corpora of 3.0 M Chinese–English sentence pairs (77.5 M Chinese / 81.0 M English running words after preprocessing) and 2.5 M Arabic–English sentence pairs (54.3 M Arabic / 55.3 M English running words after preprocessing), respectively. Word alignments are created by aligning the data in both directions with GIZA++ and symmetrizing the two trained alignments (Och and Ney, 2003). When extracting phrases, we apply several restrictions, in particular a maximum length of ten on source and target side for lexical phrases, a length limit of five on source and ten on target side for hierarchical phrases (including non-terminal symbols), and no more than two gaps per phrase.

[2] See Vilar (2011) for the pseudocode of the PUSHSUCC function and other details which are omitted here.

[3] Whether recombined derivations contribute to the generation count or not is a configuration decision (or implementation decision). Please note that some publicly available toolkits count recombined derivations by default.

Table 1: Data statistics for the test sets. Numbers have been replaced by a special category symbol.

	Chinese MT08	Arabic MT08
Sentences	1 357	1 360
Running words	34 463	45 095
Vocabulary	6 209	9 387

The decoder loads only the best translation options per distinct source side with respect to the weighted phrase-level model scores (100 for Chinese, 50 for Arabic). The language models are 4-grams with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998) which have been trained with the SRILM toolkit (Stolcke, 2002).

During decoding, a maximum length constraint of ten is applied to all non-terminals except the initial symbol S. Model weights are optimized with MERT (Och, 2003) on 100-best lists. The optimized weights are obtained (separately for deep and for shallow-1 grammars) with a k-best generation size of 1 000 for Chinese→English and of 500 for Arabic→English and kept for all setups. We employ MT06 as development sets. Translation quality is measured in truecase with BLEU (Papineni et al., 2002) on the MT08 test sets. Data statistics for the preprocessed source sides of both the Chinese→English MT08 test set and the Arabic→English MT08 test set are given in Table 1.

Our translation experiments are conducted with the open source translation toolkit Jane (Vilar et al., 2010; Vilar et al., 2012). The core implementation of the toolkit is written in C++. We compiled with GCC version 4.4.3 using its `-O2` optimization flag. We employ the SRILM libraries to perform language model scoring in the decoder. In binarized version, the language models have a size of 3.6G (Chinese→English) and 6.2G (Arabic→English). Language models and phrase tables have been copied to the local hard disks of the machines. In all experiments, the language model is completely loaded beforehand. Loading time of the language model and any other initialization steps are not included in the measured translation time. Phrase tables are in the Jane toolkit's binarized format. The decoder initializes the prefix tree structure, required nodes get loaded from secondary storage into main memory on demand, and the loaded content is being cleared each time a new input sentence is to be parsed. There is nearly no overhead due to unused data in main memory. We do not rely on memory mapping. Memory statistics are with respect to virtual memory. The hardware was equipped with RAM well beyond the requirements of the tasks, and sufficient memory has been reserved for the processes.

5.2 Experimental Results

Figures 2 and 3 depict how the Chinese→English and Arabic→English setups behave in terms of translation quality. The k-best generation size in cube pruning is varied between 10 and 10 000. The four graphs in each plot illustrate the results with combinations of deep grammar and recombination scheme T, deep grammar and recombination scheme LM, shallow grammar and recombination scheme T, as well as shallow grammar and recombination scheme LM. Figures 4 and 5 show the corresponding translation speed in words per second for these settings. The maximum memory requirements in gigabytes are given in Figures 6 and 7. In order to visualize the trade-offs between translation quality and resource consumption somewhat better, we plotted translation quality against time requirements in Figures 8 and 9 and translation quality against memory requirements in Figures 10 and 11. Translation quality and model score (averaged over all sentences; higher is better) are nicely correlated for all configurations, as can be concluded from Figures 12 through 15.

5.3 Discussion

Chinese→English. For Chinese→English translation, the system with deep grammar performs generally a bit better with respect to quality than the shallow one, which accords with the findings of other groups (de Gispert et al., 2010; Sankaran et al., 2012). The LM recombination scheme yields slightly better quality than the T scheme, and with the shallow-1 grammar it outperforms the T scheme at any given fixed amount of time or memory allocation (Figures 8 and 10).

Shallow-1 translation is up to roughly 2.5 times faster than translation with the deep grammar. However, the shallow-1 setups are considerably slowed down at higher k-best sizes as well, while the effort pays off only very moderately. Overall, the

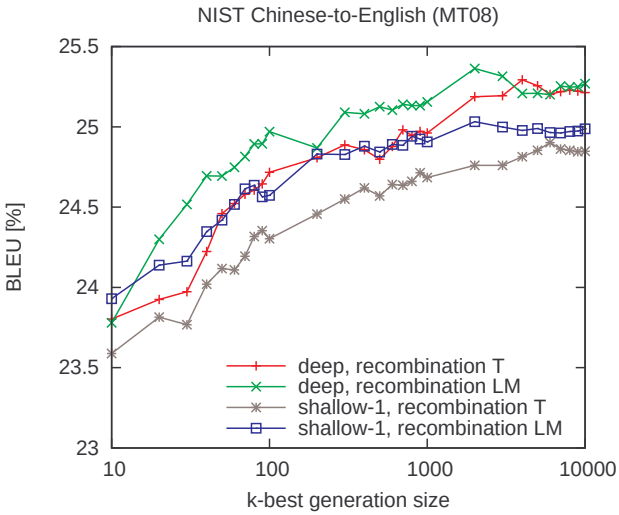

Figure 2: Chinese→English translation quality (truecase).

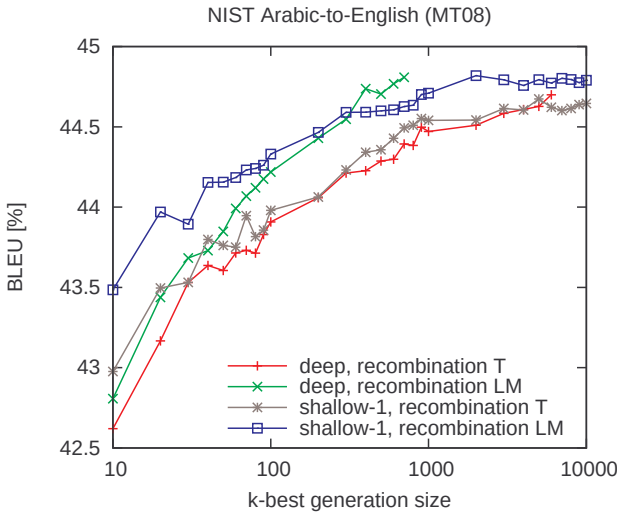

Figure 3: Arabic→English translation quality (truecase).

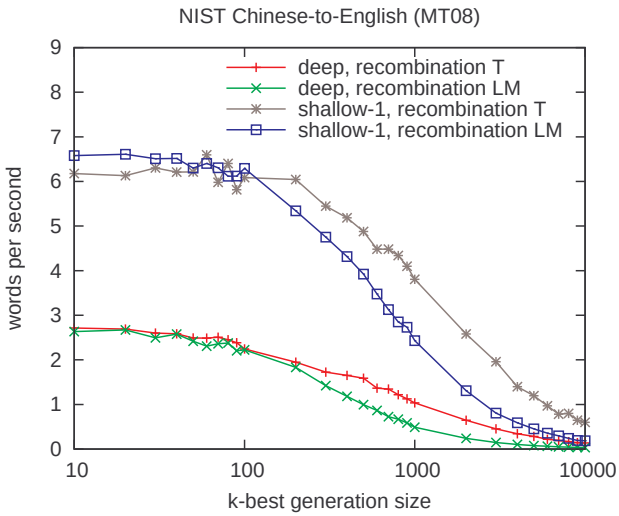

Figure 4: Chinese→English translation speed.

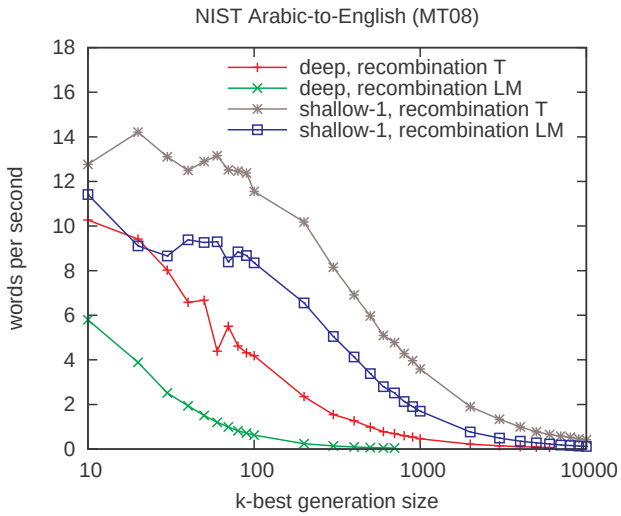

Figure 5: Arabic→English translation speed.

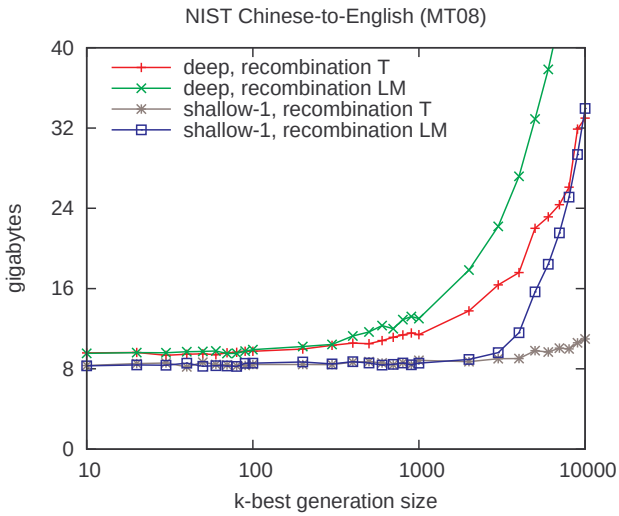

Figure 6: Chinese→English memory requirements.

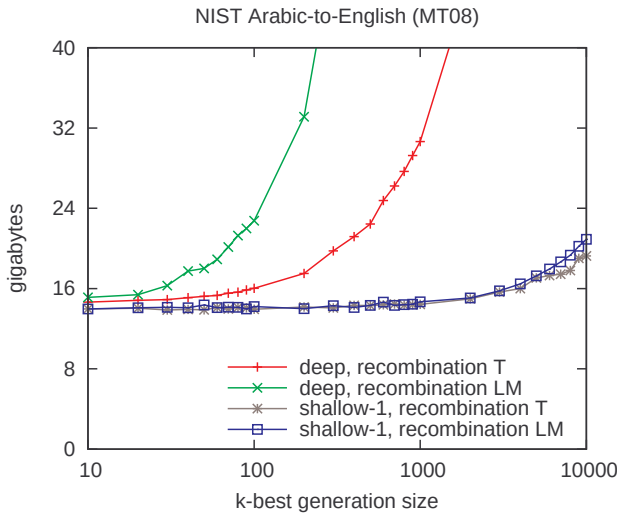

Figure 7: Arabic→English memory requirements.

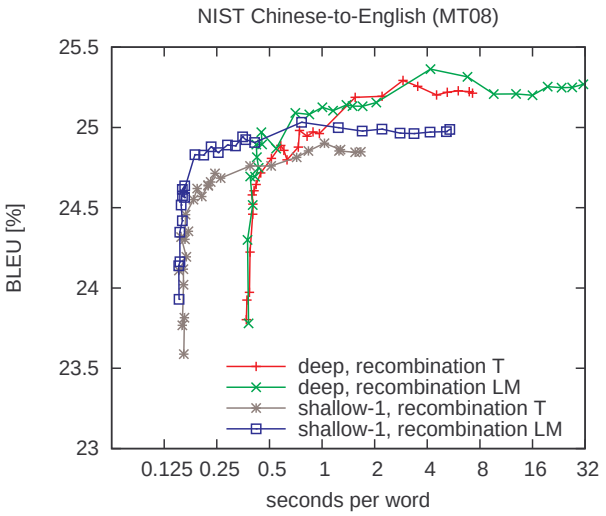

Figure 8: Trade-off between translation quality and speed for Chinese→English.

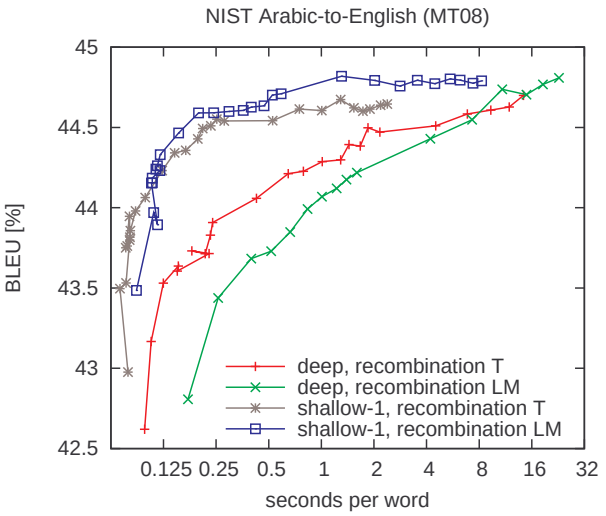

Figure 9: Trade-off between translation quality and speed for Arabic→English.

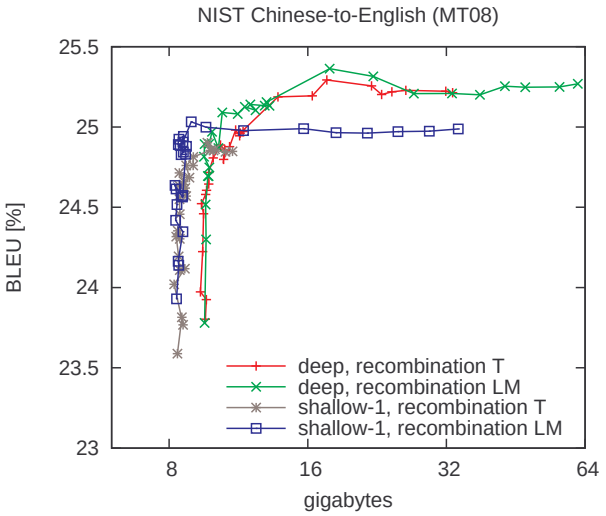

Figure 10: Trade-off between translation quality and memory requirements for Chinese→English.

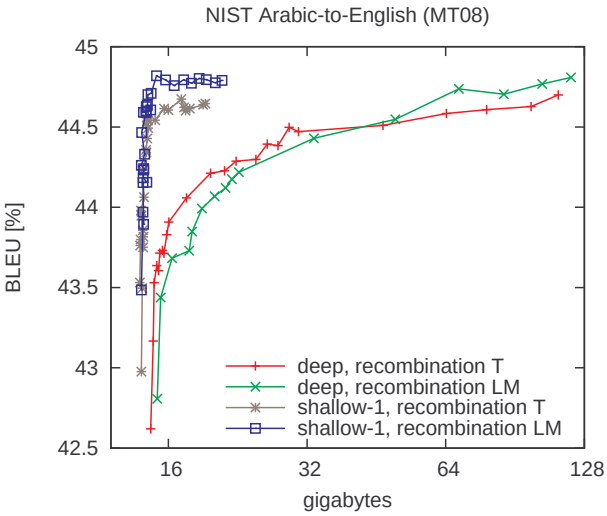

Figure 11: Trade-off between translation quality and memory requirements for Arabic→English.

shallow-1 grammar at a k-best size between 100 and 1 000 seems to offer a good compromise of quality and efficiency. Deep translation with $k = 2\,000$ and the LM recombination scheme promises high quality translation, but note the rapid memory consumption increase beyond $k = 1\,000$ with the deep grammar. At $k \leq 1\,000$, memory consumption is not an issue in both deep and shallow systems, but translation speed starts to drop at $k > 100$ already.

Arabic→English. Shallow-1 translation produces competitive quality for Arabic→English translation (de Gispert et al., 2010; Huck et al., 2011). The LM recombination scheme boosts the BLEU scores slightly. The systems with deep grammar are slowed down strongly with every increase of the k-best size. Their memory consumption likewise inflates early. We actually stopped running experiments with deep grammars for Arabic→English at $k = 7\,000$ for the T recombination scheme, and at $k = 700$ for the LM recombination scheme because 124G of memory did not suffice any more for higher k-best sizes. The memory consumption of the shallow systems stays nearly constant across a large range of the surveyed k-best sizes, but Figure 11 reveals a plateau where more resources do not improve translation quality. Increasing k from 100 to 2 000 in the shallow setup with LM recombination provides half a BLEU point, but reduces speed by a factor of more than 10.

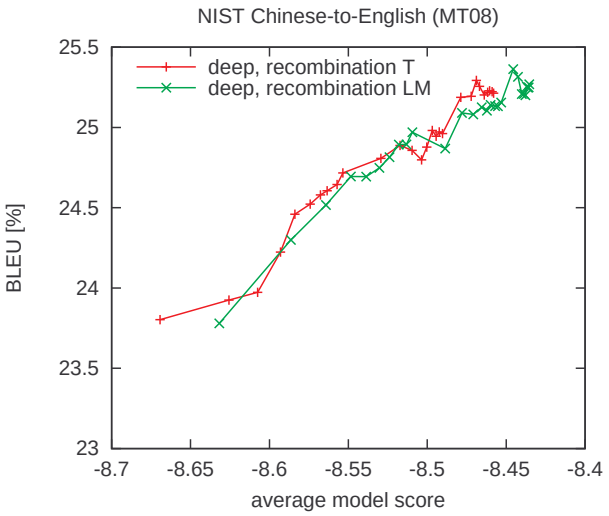

Figure 12: Relation of translation quality and average model score for Chinese→English (deep grammar).

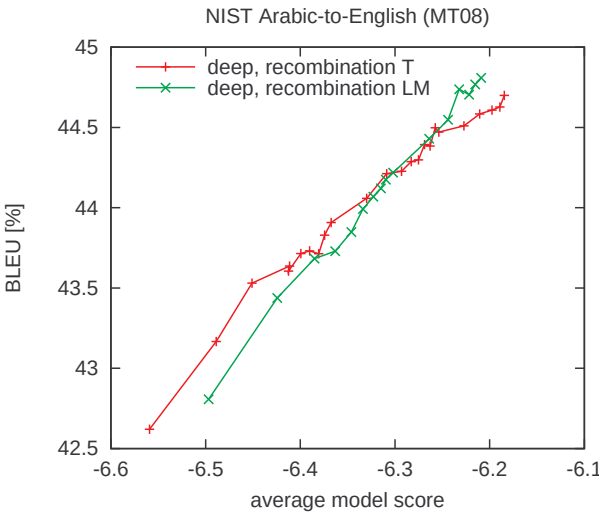

Figure 13: Relation of translation quality and average model score for Arabic→English (deep grammar).

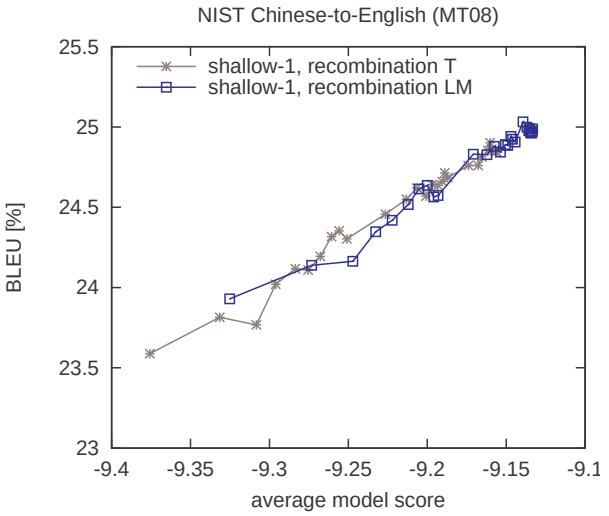

Figure 14: Relation of translation quality and average model score for Chinese→English (shallow-1 grammar).

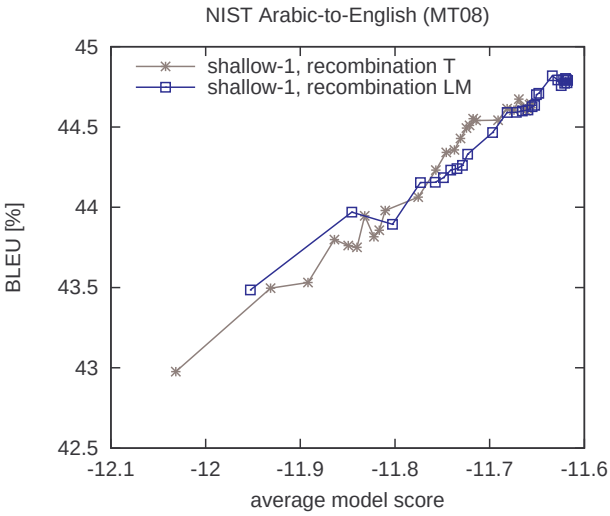

Figure 15: Relation of translation quality and average model score for Arabic→English (shallow-1 grammar).

Actual amount of derivations. We measured the amount of hypernodes (Table 2), the amount of actually generated derivations after recombination, and the amount of generated candidate derivations including recombined ones—or, equivalently, loop iterations in the algorithm from Figure 1—for selected limits k (Tables 3 and 4). The ratio of the average amount of derivations per hypernode after and before recombination remains consistently at low values for all recombination T setups. For the setups with LM recombination scheme, this recombination factor rises with larger k, i.e. the fraction of recombinable candidates increases. The increase is remarkably pronounced for Arabic→English with deep grammar. The steep slope of the recombination factor may be interpreted as an indicator for undesired overgeneration of the deep grammar on the Arabic→English task.

6 Conclusion

We systematically studied three key aspects of hierarchical phrase-based translation with cube pruning: Deep vs. shallow-1 grammars, the k-best generation size, and the hypothesis recombination scheme. In a series of empirical experiments, we revealed the trade-offs between translation quality and resource requirements to a more fine-grained degree than this is typically done in the literature.

Table 2: Average amount of hypernodes per sentence and average length of the preprocessed input sentences on the NIST Chinese→English (MT08) and Arabic→English (MT08) tasks.

	Chinese→English		Arabic→English	
	deep	**shallow-1**	**deep**	**shallow-1**
avg. #hypernodes per sentence	480.5	200.7	896.4	308.4
avg. source sentence length	25.4		33.2	

Table 3: Detailed statistics about the actual amount of derivations on the NIST Chinese→English task (MT08).

deep						
	recombination T			**recombination LM**		
k	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor
10	10.0	11.7	1.17	10.0	18.2	1.82
100	99.9	120.1	1.20	99.9	275.8	2.76
1000	950.1	1142.3	1.20	950.1	4246.9	4.47
10000	9429.8	11262.8	1.19	9418.1	72008.4	7.65

shallow-1						
	recombination T			**recombination LM**		
k	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor
10	9.7	11.3	1.17	9.6	13.6	1.41
100	90.8	105.2	1.16	90.4	168.6	1.86
1000	707.3	811.3	1.15	697.4	2143.4	3.07
10000	6478.1	7170.4	1.11	6202.8	34165.6	5.51

Table 4: Detailed statistics about the actual amount of derivations on the NIST Arabic→English task (MT08).

deep						
	recombination T			**recombination LM**		
k	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor
10	10.0	18.3	1.83	10.0	71.5	7.15
100	98.0	177.4	1.81	98.0	1726.0	17.62
500	482.1	849.0	1.76	482.1	14622.1	30.33
1000	961.8	1675.0	1.74	–	–	–

shallow-1						
	recombination T			**recombination LM**		
k	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor	avg. #derivations per hypernode (after recombination)	avg. #derivations per hypernode (incl. recombined)	factor
10	9.6	12.1	1.26	9.6	16.6	1.73
100	80.9	105.2	1.30	80.2	193.8	2.42
1000	690.1	902.1	1.31	672.1	2413.0	3.59
10000	5638.6	7149.5	1.27	5275.1	31283.6	5.93

Acknowledgments

This work was partly achieved as part of the Quaero Programme, funded by OSEO, French State agency for innovation. This material is also partly based upon work supported by the DARPA BOLT project under Contract No. HR0011-12-C-0015. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the DARPA. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 287658.

References

Hala Almaghout, Jie Jiang, and Andy Way. 2012. Extending CCG-based Syntactic Constraints in Hierarchical Phrase-Based SMT. In *Proc. of the Annual Conf. of the European Assoc. for Machine Translation (EAMT)*, pages 193–200, Trento, Italy, May.

Kathryn Baker, Michael Bloodgood, Chris Callison-Burch, Bonnie Dorr, Nathaniel Filardo, Lori Levin, Scott Miller, and Christine Piatko. 2010. Semantically-Informed Syntactic Machine Translation: A Tree-Grafting Approach. In *Proc. of the Conf. of the Assoc. for Machine Translation in the Americas (AMTA)*, Denver, CO, USA, October/November.

Jean-Cédric Chappelier and Martin Rajman. 1998. A Generalized CYK Algorithm for Parsing Stochastic CFG. In *Proc. of the First Workshop on Tabulation in Parsing and Deduction*, pages 133–137, Paris, France, April.

Stanley F. Chen and Joshua Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Computer Science Group, Harvard University, Cambridge, MA, USA, August.

David Chiang. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 263–270, Ann Arbor, MI, USA, June.

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228, June.

Adrià de Gispert, Gonzalo Iglesias, Graeme Blackwood, Eduardo R. Banga, and William Byrne. 2010. Hierarchical Phrase-Based Translation with Weighted Finite-State Transducers and Shallow-n Grammars. *Computational Linguistics*, 36(3):505–533.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A Decoder, Alignment, and Learning framework for finite-state and context-free translation models. In *Proc. of the ACL 2010 System Demonstrations*, pages 7–12, Uppsala, Sweden, July.

Kenneth Heafield, Hieu Hoang, Philipp Koehn, Tetsuo Kiso, and Marcello Federico. 2011. Left Language Model State for Syntactic Machine Translation. In *Proc. of the Int. Workshop on Spoken Language Translation (IWSLT)*, pages 183–190, San Francisco, CA, USA, December.

Kenneth Heafield, Philipp Koehn, and Alon Lavie. 2012. Language Model Rest Costs and Space-Efficient Storage. In *Proc. of the 2012 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1169–1178, Jeju Island, Korea, July.

Kenneth Heafield, Philipp Koehn, and Alon Lavie. 2013. Grouping Language Model Boundary Words to Speed K-Best Extraction from Hypergraphs. In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, Atlanta, GA, USA, June.

Hieu Hoang, Philipp Koehn, and Adam Lopez. 2009. A Unified Framework for Phrase-Based, Hierarchical, and Syntax-Based Statistical Machine Translation. In *Proc. of the Int. Workshop on Spoken Language Translation (IWSLT)*, pages 152–159, Tokyo, Japan, December.

Liang Huang and David Chiang. 2005. Better k-best Parsing. In *Proc. of the 9th Int. Workshop on Parsing Technologies*, pages 53–64, October.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 144–151, Prague, Czech Republic, June.

Matthias Huck, David Vilar, Daniel Stein, and Hermann Ney. 2011. Advancements in Arabic-to-English Hierarchical Machine Translation. In *15th Annual Conference of the European Association for Machine Translation*, pages 273–280, Leuven, Belgium, May.

Gonzalo Iglesias, Adrià de Gispert, Eduardo R. Banga, and William Byrne. 2009a. Rule Filtering by Pattern for Efficient Hierarchical Translation. In *Proc. of the 12th Conf. of the Europ. Chapter of the Assoc. for Computational Linguistics (EACL)*, pages 380–388, Athens, Greece, March.

Gonzalo Iglesias, Adrià de Gispert, Eduardo R. Banga, and William Byrne. 2009b. Hierarchical Phrase-Based Translation with Weighted Finite State Trans-

ducers. In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, pages 433–441, Boulder, CO, USA, June.

Reinhard Kneser and Hermann Ney. 1995. Improved Backing-Off for M-gram Language Modeling. In *Proc. of the International Conf. on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184, Detroit, MI, USA, May.

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. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 177–180, Prague, Czech Republic, June.

Zhifei Li and Sanjeev Khudanpur. 2008. A Scalable Decoder for Parsing-Based Machine Translation with Equivalent Language Model State Maintenance. In *Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation*, SSST '08, pages 10–18, Columbus, OH, USA, June.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. 2009a. Joshua: An Open Source Toolkit for Parsing-Based Machine Translation. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 135–139, Athens, Greece, March.

Zhifei Li, Chris Callison-Burch, Sanjeev Khudanpur, and Wren Thornton. 2009b. Decoding in Joshua: Open Source, Parsing-Based Machine Translation. *The Prague Bulletin of Mathematical Linguistics*, (91):47–56, January.

Junhui Li, Zhaopeng Tu, Guodong Zhou, and Josef van Genabith. 2012. Using Syntactic Head Information in Hierarchical Phrase-Based Translation. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 232–242, Montréal, Canada, June.

NIST. 2008. Open Machine Translation 2008 Evaluation. `http://www.itl.nist.gov/iad/mig/tests/mt/2008/`.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, March.

Franz Josef Och. 2003. Minimum Error Rate Training for Statistical Machine Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, PA, USA, July.

Baskaran Sankaran, Majid Razmara, and Anoop Sarkar. 2012. Kriya - An end-to-end Hierarchical Phrase-based MT System. *The Prague Bulletin of Mathematical Linguistics*, (97):83–98, April.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2010. String-to-Dependency Statistical Machine Translation. *Computational Linguistics*, 36(4):649–671, December.

Andreas Stolcke. 2002. SRILM – an Extensible Language Modeling Toolkit. In *Proc. of the Int. Conf. on Spoken Language Processing (ICSLP)*, volume 3, Denver, CO, USA, September.

Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. 2009. Preference Grammars: Softening Syntactic Constraints to Improve Statistical Machine Translation. In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, pages 236–244, Boulder, CO, USA, June.

David Vilar and Hermann Ney. 2012. Cardinality pruning and language model heuristics for hierarchical phrase-based translation. *Machine Translation*, 26(3):217–254, September.

David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. 2010. Jane: Open Source Hierarchical Translation, Extended with Reordering and Lexicon Models. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 262–270, Uppsala, Sweden, July.

David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. 2012. Jane: an advanced freely available hierarchical machine translation toolkit. *Machine Translation*, 26(3):197–216, September.

David Vilar. 2011. *Investigations on Hierarchical Phrase-Based Machine Translation*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, November.

Philip Williams and Philipp Koehn. 2012. GHKM Rule Extraction and Scope-3 Parsing in Moses. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 388–394, Montréal, Canada, June.

Tong Xiao, Jingbo Zhu, Hao Zhang, and Qiang Li. 2012. NiuTrans: An Open Source Toolkit for Phrase-based and Syntax-based Machine Translation. In *Proc. of the ACL 2012 System Demonstrations*, pages 19–24, Jeju, Republic of Korea, July.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A Novel Dependency-to-String Model for Statistical Machine Translation. In *Proc. of the Conf. on Empirical Methods for Natural Language Processing (EMNLP)*, pages 216–226, Edinburgh, Scotland, UK, July.

Wenduan Xu and Philipp Koehn. 2012. Extending Hiero Decoding in Moses with Cube Growing. *The Prague Bulletin of Mathematical Linguistics*, (98):133–142, October.