Large-scale Exact Decoding: The IMS-TTT submission to WMT14[*]

Daniel Quernheim
IMS
University of Stuttgart
daniel@ims.uni-stuttgart.de

Fabienne Cap
CIS
Ludwig-Maximilian University of Munich
cap@cis.uni-muenchen.de

Abstract

We present the IMS-TTT submission to WMT14, an experimental statistical tree-to-tree machine translation system based on the multi-bottom up tree transducer including rule extraction, tuning and decoding. Thanks to input parse forests and a "no pruning" strategy during decoding, the obtained translations are competitive. The drawbacks are a restricted coverage of 70% on test data, in part due to exact input parse tree matching, and a relatively high runtime. Advantages include easy redecoding with a different weight vector, since the full translation forests can be stored after the first decoding pass.

1 Introduction

In this contribution, we present an implementation of a translation model that is based on ℓMBOT (the multi bottom-up tree transducer of Arnold and Dauchet (1982) and Lilin (1978)). Intuitively, an MBOT is a synchronous tree sequence substitution grammar (STSSG, Zhang et al. (2008a); Zhang et al. (2008b); Sun et al. (2009)) that has discontiguities only on the target side (Maletti, 2011). From an algorithmic point of view, this makes the MBOT more appealing than STSSG as demonstrated by Maletti (2010). Formally, MBOT is expressive enough to express all sensible translations (Maletti, 2012)[1]. Figure 2 displays sample rules of the MBOT variant, called ℓMBOT, that we use (in a graphical representation of the trees and the alignment). Recently, a shallow version of MBOT has been integrated into the popular Moses toolkit (Braune et al., 2013). Our implementation is exact in the sense that it does absolutely no pruning during decoding and thus preserves all translation candidates, while having no mechanism to handle unknown structures. (We added dummy rules that leave unseen lexical material untranslated.) The coverage is thus limited, but still considerably high. Source-side and target-side syntax restrict the search space so that decoding stays tractable. Only the language model scoring is implemented as a separate reranker[2]. This has several advantages: (1) We can use input parse forests (Liu et al., 2009). (2) Not only is the output optimal with regard to the theoretical model, also the space of translation candidates can be efficiently stored as a weighted regular tree grammar. The best translations can then be extracted using the k-best algorithm by Huang and Chiang (2005). Rule weights can be changed without the need for explicit redecoding, the parameters of the log-linear model can be changed, and even new features can be added. These properties are especially helpful in tuning, where only the k-best algorithm has to be re-run in each iteration. A model in similar spirit has been described by Huang et al. (2006); however, it used target syntax only (using a top-down tree-to-string transducer backwards), and was restricted to sentences of length at most 25. We do not make such restrictions.

The theoretical aspects of ℓMBOT and their use in our translation model are presented in Section 2. Based on this, we implemented a machine translation system that we are going to make available to

[*]This work was supported by Deutsche Forschungsgemeinschaft grants Models of Morphosyntax for Statistical Machine Translation (Phase 2) and MA/4959/1–1.

[1]A translation is sensible if it is of linear size increase and can be computed by some (potentially copying) top-down tree transducer.

[2]Strictly speaking, this does introduce pruning into the pipeline.

the public. Section 4 presents the most important components of our ℓMBOT implementation, and Section 5 presents our submission to the WMT14 shared translation task.

2 Theoretical Model

In this section, we present the theoretical generative model that is used in our approach to syntax-based machine translation: the multi bottom-up tree transducer (Maletti, 2011). We omit the technical details and give graphical examples only to illustrate how the device works, but refer to the literature for the theoretical background. Roughly speaking, a local multi bottom-up tree transducer (ℓMBOT) has rules that replace one nonterminal symbol N on the source side by a tree, and a sequence of nonterminal symbols on the target side linked to N by one tree each. These trees again have linked nonterminals, thus allowing further rule applications.

Our ℓMBOT rules are obtained automatically from data like that in Figure 1. Thus, we (word) align the bilingual text and parse it in both the source and the target language. In this manner we obtain sentence pairs like the one shown in Figure 1. To these sentence pairs we apply the rule extraction method of Maletti (2011). The rules extracted from the sentence pair of Figure 1 are shown in Figure 2. Note the discontiguous alignment of *went* to *ist* and *gegangen*, resulting in discontiguous rules.

The application of those rules is illustrated in Figure 3 (a *pre-translation* is a pair consisting of a source tree and a sequence of target trees). While it shows a synchronous derivation, our main use case of ℓMBOT rules is *forward application* or *input restriction*, that is the calculation of all target trees that can be derived given a source tree. For a given synchronous derivation d, the source tree generated by d is $s(d)$, and the target tree is $t(d)$. The yield of a tree is the string obtained by concatenating its leaves.

Apart from ℓMBOT application to input trees, we can even apply ℓMBOT to *parse forests* and even *weighted regular tree grammars* (RTGs) (Fülöp and Vogler, 2009). RTGs offer an efficient representation of weighted forests, which are sets of trees such that each individual tree is equipped with a weight. This representation is even more efficient than packed forests (Mi et al., 2008) and moreover can represent an infinite number of weighted trees. The most important property that we utilize is that the output tree language is regular, so we can represent it by an RTG (cf. preservation of regularity (Maletti, 2011)). Indeed, every input tree can only be transformed into finitely many output trees by our model, so for a given finite input forest (which the output of the parser is) the computed output forest will also be finite and thus regular.

3 Translation Model

Given a source language sentence e and corresponding weighted parse forest $F(e)$, our translation model aims to find the best corresponding target language translation $\hat{g}$;[3] i.e.,

$$\hat{g} = \arg\max_g p(g|e) \ .$$

We estimate the probability $p(g|e)$ through a log-linear combination of component models with parameters λ_m scored on the derivations d such that the source tree of d is in the parse forest of e and the yield of the target tree reads g. With

$$D(e,g) = \{d \mid s(d) \in F(e) \text{ and } \operatorname{yield}(t(d)) = g\},$$

we thus have: [4]

$$p(g|e) \propto \sum_{d \in D(e,g)} \prod_{m=1}^{11} h_m(d)^{\lambda_m}$$

Our model uses the following features $h_m(\cdot)$ for a derivation:

(1) Translation weight normalized by source root symbol
(2) Translation weight normalized by all root symbols
(3) Translation weight normalized by leaves on the source side
(4) Lexical translation weight source $\rightarrow$ target
(5) Lexical translation weight target $\rightarrow$ source
(6) Target side language model: $p(g)$
(7) Number of words in g
(8) Number of rules used in the derivation
(9) Number of gaps in the target side sequences
(10) Penalty for rules that have more lexical material on the source side than on the target side or vice versa (absolute value)

[3]Our main translation direction is English to German.

[4]While this is the clean theoretical formulation, we make two approximations to $D(e,g)$: (1) The parser we use returns a pruned parse forest. (2) We only sum over derivations with the same target sentence that actually appear in the k-best list.

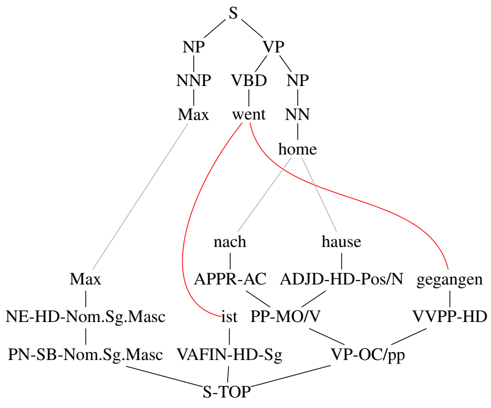

Figure 1: Aligned parsed sentences.

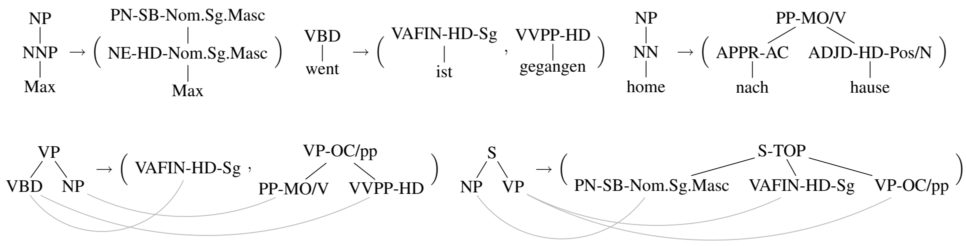

Figure 2: Extracted rules.

(11) Input parse tree probability assigned to $s(t)$ by the parser of e

The rule weights required for (1) are relative frequencies normalized over all extracted rules with the same root symbol on the left-hand side. In the same fashion the rule weights required for (2) are relative frequencies normalized over all rules with the same root symbols on both sides. The lexical weights for (4) and (5) are obtained by multiplying the word translations $w(g_i|e_j)$ [respectively, $w(e_j|g_i)$] of lexically aligned words (g_i, e_j) across (possibly discontiguous) target side sequences.[5] Whenever a source word e_j is aligned to multiple target words, we average over the word translations:[6]

$$h_4(d) = \prod_{\substack{\text{lexical item} \\ e \text{ occurs in } s(d)}} \text{average}\,\{w(g|e) \mid g \text{ aligned to } e\}$$

4 Implementation

Our implementation is very close to the theoretical model and consists of several independent components, most of which are implemented in Python. The system does not have any dependencies other than the need for parsers for the source and target language, a word alignment tool and optionally an implementation of some tuning algorithm. A schematic depiction of the training and decoding pipeline can be seen in Figure 4.

Rule extraction From a parallel corpus of which both halves have been parsed and word aligned, multi bottom-up tree transducer rules are extracted according to the procedure laid out in (Maletti, 2011). In order to handle unknown words, we add dummy identity translation rules for lexical material that was not present in the training data.

Translation model building Given a set of rules, translation weights (see above) are computed for each unique rule. The translation model is then converted into a source, a weight and a target model. The source model (an RTG represented in an efficient binary format) is used for decoding and maps input trees to trees over rule identifiers representing derivations. The weight model and the target model can be used to reconstruct the weight and the target realization of a given derivation.

[5]The lexical alignments are different from the links used to link nonterminals.

[6]If the word e_j has no alignment to a target word, then it is assumed to be aligned to a special NULL word and this alignment is scored.

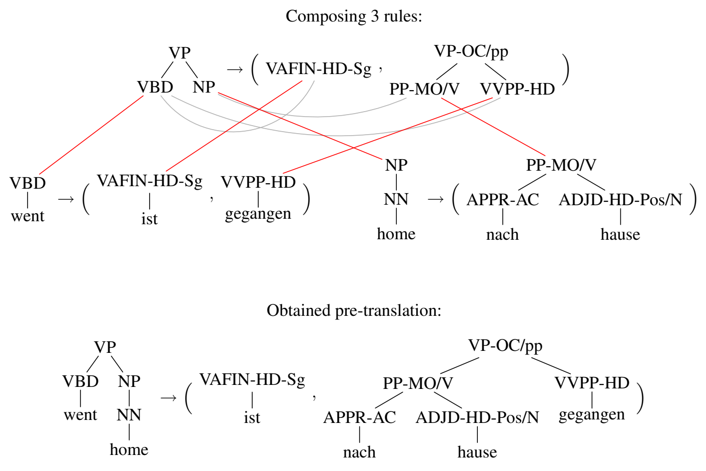

Figure 3: Synchronous rule application.

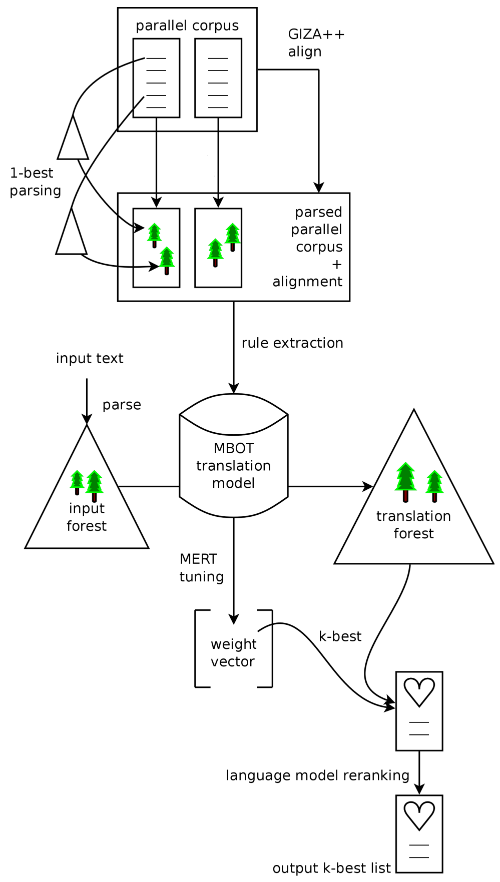

Figure 4: Our machine translation system.

Decoder The decoder transforms a forest of input sentence parse trees to a forest of translation derivations by means of forward application. These derivations are trees over the set of rules (represented by rule identifiers). One of the most useful aspects of our model is the fact that decoding is completely independent of the weights, as no pruning is performed and all translation candidates are preserved in the translation forest. Thus, even after decoding, the weight model can be changed, augmented by new features, etc.; even the target model can be changed, e.g. to support parse tree output instead of string output. In all of our experiments, we used string output, but it is conceivable to use other realizations. For instance, a syntactic language model could be used for output tree scoring. Also, recasing is extremely easy when we have part-of-speech tags to base our decision on (proper names are typically uppercase, as are all nouns in German).

Another benefit of having a packed representation of all candidates is that we can easily check whether the reference translation is included in the candidate set ("force decoding"). The freedom to allow arbitrary target models that rewrite derivations is related to current work on interpreted regular tree grammars (Koller and Kuhlmann, 2011), where arbitrary algebras can be used to compute a realization of the output tree.

k-best extractor From the translation derivation RTGs, a k-best list of derivations can be extracted (Huang and Chiang, 2005) very efficiently. This is the only step that has to be repeated if the rule weights or the parameters of the log-linear model change. The derivations are then mapped to target language sentences (if several derivations realize the same target sentence, their weights are summed) and reranked according to a language model (as was done in Huang et al. (2006)). This is the only part of the pipeline where we deviate from the theoretical log-linear model, and this is where we might make search errors. In principle, one could integrate the language model by intersection with the translation model (as the stateful MBOT model is closed under intersection with finite automata), but this is (currently) not computationally feasible due to the size of models.

Tuning Minimum error rate training (Och, 2003) is implemented using Z-MERT[7] (Zaidan, 2009). A set of source sentences has to be (forest-)parsed and decoded; the translation forests are stored on disk. Then, in each iteration of Z-MERT, it suffices to extract k-best lists from the translation forests according to the current weight vector.

5 WMT14 Experimental setup

We used the training data that was made available for the WMT14 shared translation task on English–German[8]. It consists of three parallel corpora (1.9M sentences of European parliament proceedings, 201K sentences of newswire text, and 2M sentences of web text) and additional monolingual news data for language model training.

The English half of the parallel data was parsed using Egret[9] which is a re-implementation of the Berkeley parser (Petrov et al., 2006). For the German parse, we used the BitPar parser (Schmid, 2004; Schmid, 2006). The BitPar German grammar is highly detailed, which makes the syntactic information contained in the parses extremely useful. Part-of-speech tags and category label are augmented by case, number and gender information, as can be seen in the German parse tree in Figure 1. We only kept the best parse for each sentence during training. After parsing, we prepared three versions of the German corpus: a) RAW, with no morphological post-processing; b) UNSPLIT, using SMOR, a rule-based morphological analyser (Schmid et al., 2004), to reduce words to their base form; c) SPLIT, using SMOR to reduce words to their base form and split compound nouns. After translation, compounds were merged again, and words were re-inflected. Previous experiments using SMOR to lemmatise and split compounds in phrase-based SMT showed improved translation performances, see (Cap et al., 2014a) for details.

We then trained three 5-gram language models on monolingual data using KenLM[10] (Heafield, 2011; Heafield et al., 2013 to appear) for the three setups. For SPLIT and UNSPLIT, we were only able to use the German side of the parallel data, since parsing is a prerequisite for our morphological post-processing and we did not have the resources to parse more data. For RAW, we additionally used the monolingual German data

[7]http://cs.jhu.edu/~ozaidan/zmert/

[8]http://www.statmt.org/wmt14/translation-task.html

[9]https://sites.google.com/site/zhangh1982/egret

[10]http://kheafield.com/code/kenlm/

system	BLEU	BLEU-cased	TER
RAW	17.0	16.4	.770
UNSPLIT	16.4	15.8	.773
SPLIT	16.3	15.7	.773

Table 1: BLEU and TER scores of the submitted systems.

that was distributed for the shared task. Word alignment for all three setups was achieved using GIZA++[11]. As usual, we discarded sentence pairs where one sentence was significantly longer than the other, as well as those that were too long or too short.

For tuning, we chose the WMT12 test set (3,003 sentences of newswire text), available as part of the development data for the WMT13 shared translation task. Since our system had limited coverage on this tuning set, we limited ourselves to the first a subset of sentences we could translate.

When translating the test set, our models used parse trees delivered by the Egret parser. After translation, recasing was done by examining the output syntax tree, using a simple heuristics looking for nouns and sentence boundaries. Since coverage on the test set was also limited, we used the systems as described in (Cap et al., 2014b)[12] as a fallback to translate sentences that our system was not able to translate.

6 Results

We report the overall translation quality, as listed on `http://matrix.statmt.org/`, measured using BLEU (Papineni et al., 2002) and TER (Snover et al., 2006), in Table 1.

We assume that the poor performance of UNSPLIT and SPLIT compared to RAW is due to the fact that we use a significantly smaller language model (as explained above) for these two settings. A detailed analysis will follow after the end of the manual evaluation period.

7 Conclusion and further work

We presented our submission to the WMT14 shared translation task based on a novel, promising "full syntax, no pruning" tree-to-tree approach to statistical machine translation, inspired by Huang et al. (2006). There are, however, still major drawbacks and open problems associated with our approach. Firstly, the coverage can still be significantly improved. In these experiments, our model was able to translate only 70% of the test sentences. To some extent, this number can be improved by providing more training data. Also, more rules can be extracted if we not only use the best parse for rule extraction, but multiple parse trees, or even switch to forest-based rule extraction (Mi and Huang, 2008). Finally, the size of the input parse forest plays a role. For instance, if we only supply the best parse to our model, translation will fail for approximately half of the input.

However, there are inherent coverage limits. Since our model is extremely strict, it will never be able to translate sentences whose parse trees contain structures it has never seen before, since it has to match at least one input parse tree exactly. While we implemented a simple solution to handle unknown words, the issue with unknown structures is not so easy to solve without breaking the otherwise theoretically sound approach. Possibly, glue rules can help.

The second drawback is runtime. We were able to translate about 15 sentences per hour on one processor. Distributing the translation task on different machines, we were able to translate the WMT14 test set (10k sentences) in roughly four days. Given that the trend goes towards parallel programming, and considering the fact that our decoder is written in the rather slow language Python, we are confident that this is not a major problem. We were able to run the whole pipeline of training, tuning and evaluation on the WMT14 shared task data in less than one week. We are currently investigating whether A* k-best algorithms (Pauls and Klein, 2009; Pauls et al., 2010) can help to guide the translation process while maintaining optimality.

Thirdly, currently the language model is not integrated, but implemented as a separate reranking component. We are aware that this might introduce search errors, and that an integrated language model might improve translation quality (see e.g. Chiang (2007) where 3–4 BLEU points are gained by LM integration). Some research on this topic already exists, e.g. (Rush and Collins, 2011) who use dual decomposition, and (Aziz et al., 2013) who replace intersection with an upper bound which is easier to compute.

[11]`https://code.google.com/p/giza-pp/`

[12]We use raw as described in (Cap et al., 2014b) as a fallback for RAW, RI for UNSPLIT and CoRI for SPLIT.

References

André Arnold and Max Dauchet. 1982. Morphismes et bimorphismes d'arbres. *Theoret. Comput. Sci.*, 20(1):33–93.

Wilker Aziz, Marc Dymetman, and Sriram Venkatapathy. 2013. Investigations in exact inference for hierarchical translation. In *Proc. 8th WMT*, pages 472–483.

Fabienne Braune, Nina Seemann, Daniel Quernheim, and Andreas Maletti. 2013. Shallow local multi-bottom-up tree transducers in statistical machine translation. In *Proc. 51th ACL*, pages 811–821.

Fabienne Cap, Alexander Fraser, Marion Weller, and Aoife Cahill. 2014a. How to Produce Unseen Teddy Bears: Improved Morphological Processing of Compounds in SMT. In *Proc. 14th EACL*.

Fabienne Cap, Marion Weller, Anita Ramm, and Alexander Fraser. 2014b. CimS – The CIS and IMS joint submission to WMT 2014 translating from English into German. In *Proc. 9th WMT*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computat. Linguist.*, 33(2):201–228.

Zoltán Fülöp and Heiko Vogler. 2009. Weighted tree automata and tree transducers. In Manfred Droste, Werner Kuich, and Heiko Vogler, editors, *Handbook of Weighted Automata*, EATCS Monographs on Theoret. Comput. Sci., chapter 9, pages 313–403. Springer.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013 (to appear). Scalable modified Kneser-Ney language model estimation. In *Proc. 51st ACL*.

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proc. 6th WMT*, pages 187–197.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proc. IWPT*, pages 53–64.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. 7th Conf. AMTA*, pages 66–73.

Alexander Koller and Marco Kuhlmann. 2011. A generalized view on parsing and translation. In *Proc. IWPT*, pages 2–13.

Eric Lilin. 1978. *Une généralisation des transducteurs d'états finis d'arbres: les S-transducteurs*. Thèse 3ème cycle, Université de Lille.

Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proc. 47th ACL*, pages 558–566.

Andreas Maletti. 2010. Why synchronous tree substitution grammars? In *Proc. HLT-NAACL*, pages 876–884.

Andreas Maletti. 2011. How to train your multi bottom-up tree transducer. In *Proc. 49th ACL*, pages 825–834.

Andreas Maletti. 2012. Every sensible extended top-down tree transducer is a multi bottom-up tree transducer. In *Proc. HLT-NAACL*, pages 263–273.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In *Proc. EMNLP*, pages 206–214.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. 46th ACL*, pages 192–199. ACL.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. 41st ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. 40th ACL*, pages 311–318.

Adam Pauls and Dan Klein. 2009. K-best A* parsing. In *Proc. 47th ACL*, pages 958–966.

Adam Pauls, Dan Klein, and Chris Quirk. 2010. Top-down k-best A* parsing. In *Proc. 48th ACL*, pages 200–204.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. COLING-ACL*, pages 433–440.

Alexander M. Rush and Michael Collins. 2011. Exact decoding of syntactic translation models through lagrangian relaxation. In *Proc. 49th ACL*, pages 72–82.

Helmut Schmid, Arne Fitschen, and Ulrich Heid. 2004. SMOR: A German Computational Morphology Covering Derivation, Composition and Inflection. In *Proc. 4th LREC*.

Helmut Schmid. 2004. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proc. 20th COLING*, pages 162–168.

Helmut Schmid. 2006. Trace prediction and recovery with unlexicalized PCFGs and slash features. In *Proc. 44th ACL*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proc. AMTA*.

Jun Sun, Min Zhang, and Chew Lim Tan. 2009. A non-contiguous tree sequence alignment-based model for statistical machine translation. In *Proc. 47th ACL*, pages 914–922.

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

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008a. A tree sequence alignment-based tree-to-tree translation model. In *Proc. 46th ACL*, pages 559–567.

Min Zhang, Hongfei Jiang, Haizhou Li, Aiti Aw, and Sheng Li. 2008b. Grammar comparison study for translational equivalence modeling and statistical machine translation. In *Proc. 22nd COLING*, pages 1097–1104.