Scalable Inference and Training of Context-Rich Syntactic Translation Models

Michel Galley[*], **Jonathan Graehl**[†], **Kevin Knight**[†‡], **Daniel Marcu**[†‡], **Steve DeNeefe**[†], **Wei Wang**[‡] and **Ignacio Thayer**[†]

[*]Columbia University, Dept. of Computer Science, New York, NY 10027

[†]University of Southern California, Information Sciences Institute, Marina del Rey, CA 90292

[‡]Language Weaver, Inc., 4640 Admiralty Way, Marina del Rey, CA 90292

galley@cs.columbia.edu, {graehl,knight,marcu,sdeneefe}@isi.edu, wwang@languageweaver.com, thayer@google.com

Abstract

Statistical MT has made great progress in the last few years, but current translation models are weak on re-ordering and target language fluency. Syntactic approaches seek to remedy these problems. In this paper, we take the framework for acquiring multi-level syntactic translation rules of (Galley et al., 2004) from aligned tree-string pairs, and present two main extensions of their approach: first, instead of merely computing a single derivation that minimally explains a sentence pair, we construct a large number of derivations that include contextually richer rules, and account for multiple interpretations of unaligned words. Second, we propose probability estimates and a training procedure for weighting these rules. We contrast different approaches on real examples, show that our estimates based on multiple derivations favor phrasal re-orderings that are linguistically better motivated, and establish that our larger rules provide a 3.63 BLEU point increase over minimal rules.

1 Introduction

While syntactic approaches seek to remedy word-ordering problems common to statistical machine translation (SMT) systems, many of the earlier models—particularly child re-ordering models—fail to account for human translation behavior. Galley et al. (2004) alleviate this modeling problem and present a method for acquiring millions of syntactic transfer rules from bilingual corpora, which we review below. Here, we make the following new contributions: (1) we show how to acquire larger rules that crucially condition on more syntactic context, and show how to compute multiple derivations for each training example, capturing both large and small rules, as well as multiple interpretations for unaligned words; (2) we develop probability models for these multi-level transfer rules, and give estimation methods for assigning probabilities to very large rule sets. We contrast our work with (Galley et al., 2004), highlight some severe limitations of probability estimates computed from single derivations, and demonstrate that it is critical to account for many derivations for each sentence pair. We also use real examples to show that our probability models estimated from a large number of derivations favor phrasal re-orderings that are linguistically well motivated. An empirical evaluation against a state-of-the-art SMT system similar to (Och and Ney, 2004) indicates positive prospects. Finally, we show that our contextually richer rules provide a 3.63 BLEU point increase over those of (Galley et al., 2004).

2 Inferring syntactic transformations

We assume we are given a source-language (e.g., French) sentence $\mathbf{f}$, a target-language (e.g., English) parse tree π, whose yield $\mathbf{e}$ is a translation of $\mathbf{f}$, and a word alignment $\mathbf{a}$ between $\mathbf{f}$ and $\mathbf{e}$. Our aim is to gain insight into the process of transforming π into $\mathbf{f}$ and to discover grammatically-grounded translation rules. For this, we need a formalism that is expressive enough to deal with cases of syntactic divergence between source and target languages (Fox, 2002): for any given $(\pi, \mathbf{f}, \mathbf{a})$ triple, it is useful to produce a derivation that minimally explains the transformation between π and $\mathbf{f}$, while remaining consistent with $\mathbf{a}$. Galley et al. (2004) present one such formalism (henceforth "GHKM").

2.1 Tree-to-string alignments

It is appealing to model the transformation of π into $\mathbf{f}$ using tree-to-string (**xRs**) transducers, since their theory has been worked out in an extensive literature and is well understood (see, e.g., (Graehl and Knight, 2004)). Formally, transformational rules r_i presented in (Galley et al., 2004) are equivalent to 1-state **xRs** transducers mapping a given pattern (subtree to match in π) to a right hand side string. We will refer to them as $lhs(r_i)$ and $rhs(r_i)$, respectively. For example, some **xRs**

rules may describe the transformation of *does not* into *ne ... pas* in French. A particular instance may look like this:

VP(AUX(*does*), RB(*not*), x_0:VB) → *ne*, x_0, *pas*

$lhs(r_i)$ can be any arbitrary syntax tree fragment. Its leaves are either lexicalized (e.g. *does*) or variables (x_0, x_1, etc). $rhs(r_i)$ is represented as a sequence of target-language words and variables.

Now we give a brief overview of how such transformational rules are acquired automatically in GHKM.[1] In Figure 1, the $(\pi, \mathbf{f}, \mathbf{a})$ triple is represented as a directed graph G (edges going downward), with no distinction between edges of π and alignments. Each node of the graph is labeled with its **span** and **complement span** (the latter in italic in the figure). The span of a node n is defined by the indices of the first and last word in **f** that are reachable from n. The complement span of n is the union of the spans of all nodes n' in G that are neither descendants nor ancestors of n. Nodes of G whose spans and complement spans are non-overlapping form the **frontier set** $F \in G$.

What is particularly interesting about the frontier set? For any frontier of graph G containing a given node $n \in F$, spans on that frontier define an ordering between n and each other frontier node n'. For example, the span of VP[4-5] either precedes or follows, but never overlaps the span of any node n' on any graph frontier. This property does not hold for nodes outside of F. For instance, PP[4-5] and VBG[4] are two nodes of the same graph frontier, but they cannot be ordered because of their overlapping spans.

The purpose of **xRs** rules in this framework is to order constituents along sensible frontiers in G, and all frontiers containing undefined orderings, as between PP[4-5] and VBG[4], must be disregarded during rule extraction. To ensure that **xRs** rules are prevented from attempting to re-order any such pair of constituents, these rules are designed in such a way that variables in their lhs can only match nodes of the frontier set. Rules that satisfy this property are said to be **induced** by G.[2] For example, rule **(d)** in Table 1 is valid according to GHKM, since the spans corresponding to its *rhs* constituents (VBP[3] and NP[4-8]) do not overlap. Conversely, NP(x_0:DT, x_1:CD:, x_2:NNS) is not the lhs of any rule extractible from G, since its frontier constituents CD[2] and NNS[2] have overlapping spans.[3] Finally, the GHKM procedure produces a single derivation from G, which is shown in Table 1.

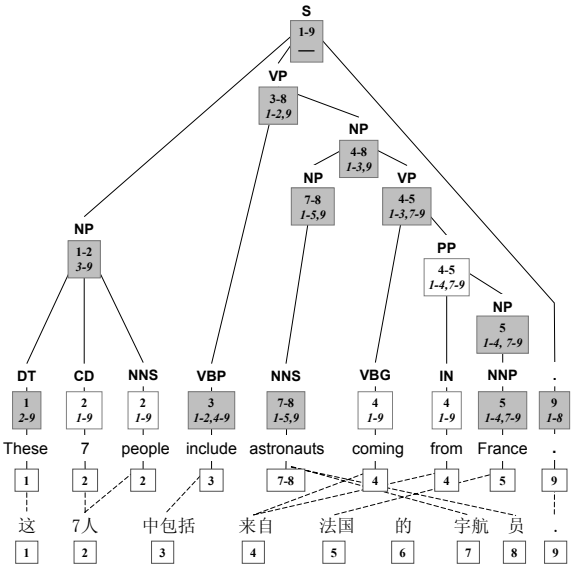


Figure 1: **Spans** and **complement-spans** determine what rules are extracted. Constituents in gray are members of the **frontier set**; a minimal rule is extracted from each of them.

(a)	S(x_0:NP, x_1:VP, x_2:.) → x_0, x_1, x_2
(b)	NP(x_0:DT, CD(*7*), NNS(*people*)) → x_0, 7人
(c)	DT(*these*) → 这
(d)	VP(x_0:VBP, x_1:NP) → x_0, x_1
(e)	VBP(*include*) → 中包括
(f)	NP(x_0:NP, x_1:VP) → x_1, 的, x_0
(g)	NP(x_0:NNS) → x_0
(h)	NNS(*astronauts*) → 宇航, 员
(i)	VP(VBG(*coming*), PP(IN(*from*), x_0:NP)) → 来自, x_0
(j)	NP(x_0:NNP) → x_0
(k)	NNP(*France*) → 法国
(l)	.(.) → .

Table 1: A minimal derivation corresponding to Figure 1.

The concern in GHKM was to extract minimal rules, whereas ours is to extract rules of any arbitrary size. **Minimal** rules defined over G are those that cannot be decomposed into simpler rules induced by the same graph G, e.g., all rules in Table 1. We call minimal a derivation that only contains minimal rules. Conversely, a **composed** rule results from the composition of two or more minimal rules, e.g., rule (b) and (c) compose into:

NP(DT(*these*), CD(*7*), NNS(*people*)) → 这, 7人

[1] Note that we use a slightly different terminology.

[2] Specifically, an **xRs** rule r_i is extracted from G by taking a subtree $\gamma \in \pi$ as $lhs(r_i)$, appending a variable to each leaf node of γ that is internal to π, adding those variables to $rhs(r_i)$, ordering them in accordance to **a**, and if necessary inserting any word of **f** to ensure that $rhs(r_i)$ is a sequence of contiguous spans (e.g., [4-5][6][7-8] for rule **(f)** in Table 1).

[3] It is generally reasonable to also require that the root n of $lhs(r_i)$ be part of F, because no rule induced by G can compose with r_i at n, due to the restrictions imposed on the extraction procedure, and r_i wouldn't be part of any valid derivation.

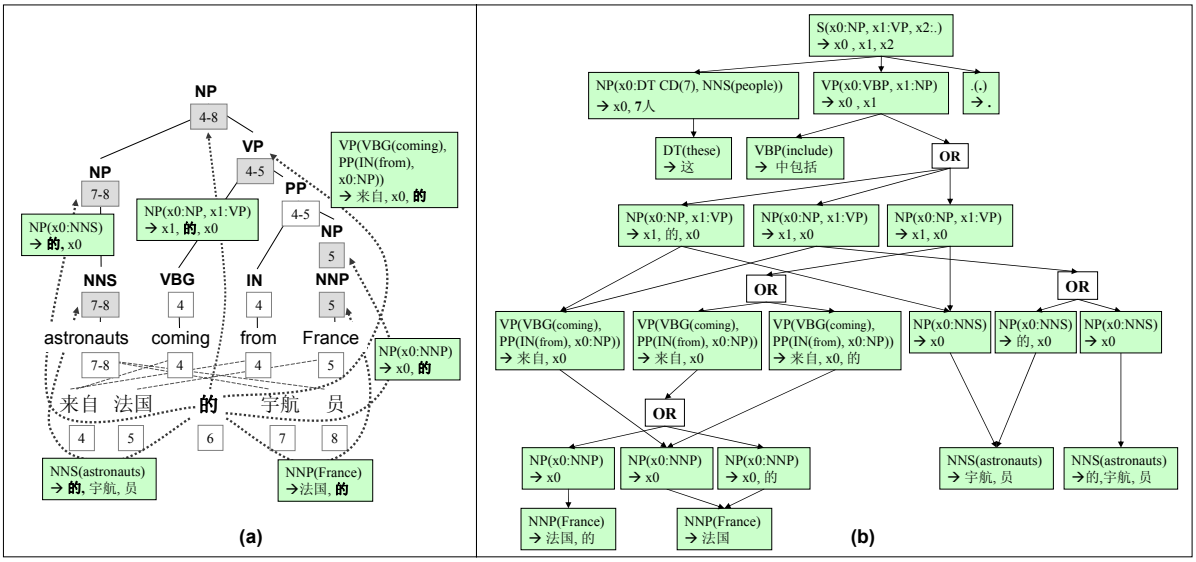


Figure 2: **(a)** Multiple ways of aligning 的 to constituents in the tree. **(b)** Derivation corresponding to the parse tree in Figure 1, which takes into account all alignments of 的 pictured in **(a)**.

Note that these properties are dependent on G, and the above rule would be considered a minimal rule in a graph G' similar to G, but additionally containing a word alignment between 7 and 这. We will see in Sections 3 and 5 why extracting only minimal rules can be highly problematic.

2.2 Unaligned words

While the general theory presented in GHKM accounts for any kind of derivation consistent with G, it does not particularly discuss the case where some words of the source-language string **f** are not aligned to any word of **e**, thus disconnected from the rest of the graph. This case is highly frequent: 24.1% of Chinese words in our 179 million word English-Chinese bilingual corpus are unaligned, and 84.8% of Chinese sentences contain at least one unaligned word. The question is what to do with such lexical items, e.g., 的 in Figure 2(a). The approach of building one minimal derivation for G as in the algorithm described in GHKM assumes that we commit ourselves to a particular heuristic to attach the unaligned item to a certain constituent of π, e.g., highest attachment (in the example, 的 is attached to NP[4-8] and the heuristic generates rule **(f)**). A more reasonable approach is to invoke the principle of insufficient reason and make no a priori assumption about what is a "correct" way of assigning the item to a constituent, and return *all* derivations that are consistent with G. In Section 4, we will see how to use corpus evidence to give preference to unaligned-word attachments that are the most consistent across the data. Figure 2(a) shows the six possible ways of attaching 的 to constituents of π: besides the highest attachment (rule **(f)**), 的 can move along the ancestors of *France*, since it is to the right of the translation of that word, and be considered to be part of an NNP, NP, or VP rule. We make the same reasoning to the left: 的 can either start the NNS of *astronauts*, or start an NP.

Our account of all possible ways of consistently attaching 的 to constituents means we must extract more than one derivation to explain transformations in G, even if we still restrict ourselves to minimal derivations (a minimal derivation for G is unique if and only if no source-language word in G is unaligned). While we could enumerate all derivations separately, it is much more efficient both in time and space to represent them as a **derivation forest**, as in Figure 2(b). Here, the forest covers all minimal derivations that correspond to G. It is necessary to ensure that for each derivation, each unaligned item (here 的) appears only once in the rules of that derivation, as shown in Figure 2 (which satisfies the property). That requirement will prove to be critical when we address the problem of estimating probabilities for our rules: if we allowed in our example to spuriously generate 的s in multiple successive steps of the same derivation, we would not only represent the transformation incorrectly, but also 的-rules would be disproportionately represented, leading to strongly biased estimates. We will now see how to ensure this constraint is satisfied in our rule extraction and derivation building algorithm.

2.3 Algorithm

The linear-time algorithm presented in GHKM is only a particular case of the more general one we describe here, which is used to extract all rules, minimal and composed, induced by G. Similarly to the GHKM algorithm, ours performs a top-down traversal of G, but differs in the operations it performs at each node $n \in F$: we must explore all subtrees rooted at n, find all consistent ways of attaching unaligned words of **f**, and build valid derivations in accordance to these attachments.

We use a table **or-dforest**$[x, y, c]$ to store OR-nodes, in which each OR-node can be uniquely defined by a syntactic category c and a span $[x, y]$ (which may cover unaligned words of **f**). This table is used to prevent the same partial derivation to be followed multiple times (the in-degrees of OR-nodes generally become large with composed rules). Furthermore, to avoid over-generating unaligned words, the root and variables in each rule are represented with their spans. For example, in Figure 2(b), the second and third child of the topmost OR-node respectively span across [4-5][6-8] and [4-6][7-8] (after constituent reordering). In the former case, 的 will eventually be realized in an NP, and in the latter case, in a VP.

The preprocessing step consists of assigning spans and complement spans to nodes of G, in the first case by a bottom-up exploration of the graph, and in the latter by a top-down traversal. To assign complement spans, we assign the complement span of any node n to each of its children, and for each of them, add the span of the child to the complement span of all other children. In another traversal of G, we determine the minimal rule extractible from each node in F.

We explore all tree fragments rooted at n by maintaining an open and a closed queue of rules extracted from n (q_o and q_c). At each step, we pick the smallest rule in q_o, and for each of its variable nodes, try to discover new rules ('successor rules') by means of composition with minimal rules, until a given threshold on rule size or maximum number of rules in q_c is reached. There may be more that one successor per rule, since we must account for all possible spans than can be assigned to non-lexical leaves of a rule. Once a threshold is reached, or if the open queue is empty, we connect a new OR-node to all rules that have just been extracted from n, and add it to **or-dforest**. Finally, we proceed recursively, and extract new rules from each node at the frontier of the minimal rule rooted at n. Once all nodes of F have been processed, the **or-dforest** table contains a representation encoding only valid derivations.

3 Probability models

The overall goal of our translation system is to transform a given source-language sentence $\mathbf{f}$ into an appropriate translation $\mathbf{e}$ in the set $\mathbf{E}$ of all possible target-language sentences. In a noisy-channel approach to SMT, we uses Bayes' theorem and choose the English sentence $\hat{\mathbf{e}} \in \mathbf{E}$ that maximizes:[4]

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e} \in \mathbf{E}} \Big\{ Pr(\mathbf{e}) \cdot Pr(\mathbf{f}|\mathbf{e}) \Big\} \quad (1)$$

$Pr(\mathbf{e})$ is our language model, and $Pr(\mathbf{f}|\mathbf{e})$ our translation model. In a grammatical approach to MT, we hypothesize that syntactic information can help produce good translation, and thus introduce dependencies on target-language syntax trees. The function to optimize becomes:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e} \in \mathbf{E}} \Big\{ Pr(\mathbf{e}) \sum_{\pi \in \tau(\mathbf{e})} Pr(\mathbf{f}|\pi) \cdot Pr(\pi|\mathbf{e}) \Big\} \quad (2)$$

$\tau(\mathbf{e})$ is the set of all English trees that yield the given sentence $\mathbf{e}$. Estimating $Pr(\pi|\mathbf{e})$ is a problem equivalent to syntactic parsing and thus is not discussed here. Estimating $Pr(\mathbf{f}|\pi)$ is the task of syntax-based translation models (SBTM).

Given a rule set R, our SBTM makes the common assumption that left-most compositions of **xRs** rules $\theta_i = r_1 \circ ... \circ r_n$ are independent from one another in a given derivation $\theta_i \in \Theta$, where Θ is the set of all derivations constructible from $G = (\pi, \mathbf{f}, \mathbf{a})$ using rules of R. Assuming that Λ is the set of all subtree decompositions of π corresponding to derivations in Θ, we define the estimate:

$$Pr(\mathbf{f}|\pi) = \frac{1}{|\Lambda|} \sum_{\theta_i \in \Theta} \prod_{r_j \in \theta_i} p(rhs(r_j)|lhs(r_j)) \quad (3)$$

under the assumption:

$$\sum_{r_j \in R : lhs(r_j) = lhs(r_i)} p(rhs(r_j)|lhs(r_j)) = 1 \quad (4)$$

It is important to notice that the probability distribution defined in Equation 3 requires a normalization factor ($|\Lambda|$) in order to be tight, i.e., sum to 1 over all strings $\mathbf{f_i} \in \mathbf{F}$ that can be derived

[4] We denote general probability distributions with $Pr(\cdot)$ and use $p(\cdot)$ for probabilities assigned by our models.

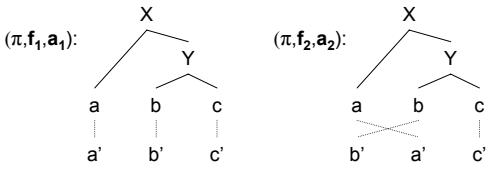


Figure 3: Example corpus.

from π. A simple example suffices to demonstrate it is not tight without normalization. Figure 3 contains a sample corpus from which four rules can be extracted:

r_1: X(a, Y(b, c)) → a', b', c'
r_2: X(a, Y(b, c)) → b', a', c'
r_3: X(a, x_0:Y) → a', x_0
r_4: Y(b, c) → b', c'

From Equation 4, the probabilities of r_3 and r_4 must be 1, and those of r_1 and r_2 must sum to 1. Thus, the total probability mass, which is distributed across two possible output strings a'b'c' and b'a'c', is: p(a'b'c'$|\pi$) + p(b'a'c'$|\pi$) = $p_1 + p_3 \cdot p_4 + p_2$ = 2, where $p_i = p(rhs(r_i)|lhs(r_i))$.

It is relatively easy to prove that the probabilities of all derivations that correspond to a given decomposition $\lambda_i \in \Lambda$ sum to 1 (the proof is omitted due to constraints on space). From this property we can immediately conclude that the model described by Equation 3 is tight.[5]

We examine two estimates $p(rhs(r)|lhs(r))$. The first one is the relative frequency estimator conditioning on left hand sides:

$$p(rhs(r)|lhs(r)) = \frac{f(r)}{\sum_{r':lhs(r')=lhs(r)} f(r')} \quad (5)$$

$f(r)$ represents the number of times rule r occurred in the derivations of the training corpus.

One of the major negative consequences of extracting only minimal rules from a corpus is that an estimator such as Equation 5 can become extremely biased. This again can be observed from Figure 3. In the minimal-rule extraction of GHKM, only three rules are extracted from the example corpus, i.e. rules r_2, r_3, and r_4. Let's assume now that the triple $(\pi, \mathbf{f}_1, \mathbf{a}_1)$ is represented 99 times, and $(\pi, \mathbf{f}_2, \mathbf{a}_2)$ only once. Given a tree π, the model trained on that corpus can generate the two strings a'b'c' and b'a'c' only through two derivations, $r_3 \circ r_4$ and r_2, respectively. Since all rules in that example have probability 1, and given that the normalization factor $|\Lambda|$ is 2, both probabilities p(a'b'c'$|\pi$) and p(b'a'c'$|\pi$) are 0.5. On the other hand, if all rules are extracted and incorporated into our relative-frequency probability model, r_1 seriously counterbalances r_2 and the probability of a'b'c' becomes: $\frac{1}{2} \cdot (\frac{99}{100} + 1) = .995$ (since it differs from .99, the estimator remains biased, but to a much lesser extent).

An alternative to the conditional model of Equation 3 is to use a joint model conditioning on the root node instead of the entire left hand side:

$$p(r|root(r)) = \frac{f(r)}{\sum_{r':root(r')=root(r)} f(r')} \quad (6)$$

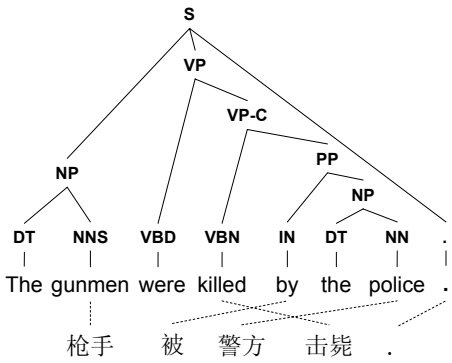
This can be particularly useful if no parser or syntax-based language model is available, and we need to rely on the translation model to penalize ill-formed parse trees. Section 6 will describe an empirical evaluation based on this estimate.

4 EM training

In our previous discussion of parameter estimation, we did not explore the possibility that one derivation in a forest may be much more plausible than the others. If we knew which derivation in each forest was the "true" derivation, then we could straightforwardly collect rule counts off those derivations. On the other hand, if we had good rule probabilities, we could compute the most likely (Viterbi) derivations for each training example. This is a situation in which we can employ EM training, starting with uniform rule probabilities. For each training example, we would like to: (1) score each derivation θ_i as a product of the probabilities of the rules it contains, (2) compute a conditional probability p_i for each derivation θ_i (conditioned on the observed training pair) by normalizing those scores to add to 1, and (3) collect weighted counts for each rule in each θ_i, where the weight is p_i. We can then normalize the counts to get refined probabilities, and iterate; the corpus likelihood is guaranteed to improve with each iteration. While it is infeasible to enumerate the millions of derivations in each forest, Graehl and Knight (2004) demonstrate an efficient algorithm. They also analyze how to train arbitrary tree transducers into two steps. The first step is to build a derivation forest for each training example, where the forest contains those derivations licensed by the (already supplied) transducer's rules. The second step employs EM on those derivation forests, running in time proportional to the size of the

[5] If each tree fragment in π is the lhs of some rule in R, then we have $|\Lambda| = 2^n$, where n is the number of nodes of the frontier set $F \in G$ (each node is a binary choice point).

	Best minimal-rule derivation (C_m)	$p(r)$
(a)	S(x_0:NP-C x_1:VP x_2:.) → x_0 x_1 x_2	.845
(b)	NP-C(x_0:NPB) → x_0	.82
(c)	NPB(DT(*the*) x_0:NNS) → x_0	.507
(d)	NNS(*gunmen*) → 枪手	.559
(e)	VP(VBD(*were*) x_0:VP-C) → x_0	.434
(f)	VP-C(x_0:VBN x_1:PP) → x_1 x_0	.374
(g)	PP(x_0:IN x_1:NP-C) → x_0 x_1	.64
(h)	IN(*by*) → 被	.0067
(i)	NP-C(x_0:NPB) → x_0	.82
(j)	NPB(DT(*the*) x_0:NN) → x_0	.586
(k)	NN(*police*) → 警方	.0429
(l)	VBN(*killed*) → 击毙	.0072
(m)	.(.) → .	.981



	Best composed-rule derivation (C_4)	$p(r)$
(o)	S(NP-C(NPB(DT(*the*) NNS(*gunmen*))) x_0:VP .(.)) → 枪手 x_0 .	1
(p)	VP(VBD(*were*) VP-C(x_0:VBN PP(IN(*by*) x_1:NP-C))) → 被 x_1 x_0	0.00724
(q)	NP-C(NPB(DT(*the*) NN(*police*))) → 警方	0.173
(r)	VBN(*killed*) → 击毙	0.00719

Figure 4: Two most probable derivations for the graph on the right: the top table restricted to minimal rules; the bottom one, much more probable, using a large set of composed rules. Note: the derivations are constrained on the $(\pi, \mathbf{f}, \mathbf{a})$ triple, and thus include some non-literal translations with relatively low probabilities (e.g. *killed*, which is more commonly translated as 死亡).

rule set	nb. of rules	nb. of nodes	deriv-time	EM-time
C_m	4M	192M	2 h.	4 h.
C_3	142M	1255M	52 h.	34 h.
C_4	254M	2274M	134 h.	60 h.

Table 2: Rules and derivation nodes for a 54M-word, 1.95M sentence pair English-Chinese corpus, and time to build derivations (on 10 cluster nodes) and run 50 EM iterations.

forests. We only need to borrow the second step for our present purposes, as we construct our own derivation forests when we acquire our rule set.

A major challenge is to scale up this EM training to large data sets. We have been able to run EM for 50 iterations on our Chinese-English 54-million word corpus. The derivation forests for this corpus contain 2.2 billion nodes; the largest forest contains 1.1 million nodes. The outcome is to assign probabilities to over 254 million rules. Our EM runs with either lhs normalization or lhs-$root$ normalization. In the former case, each lhs has an average of three corresponding rhs's that compete with each other for probability mass.

5 Model coverage

We now present some examples illustrating the benefit of composed rules. We trained three $p(rhs(r_i)|lhs(r_i))$ models on a 54 million-word English-Chinese parallel corpus (Table 2): the first one (C_m) with only minimal rules, and the two others (C_3 and C_4) additionally considering composed rules with no more than three, respectively four, internal nodes in $lhs(r_i)$. We evaluated these models on a section of the NIST 2002 evaluation corpus, for which we built derivation forests and

lhs: S(x_0:NP-C VP(x_1:VBD x_2:NP-C) x_3:.)		
corpus	rhs_i	$p(rhs_i\|lhs)$
Chinese (minimal)	x_1 x_0 x_2 x_3	.3681
	x_0 x_1 第 x_3 x_2	.0357
	x_2 , x_0 x_1 x_3	.0287
	x_0 x_1 第 x_3 x_2 .	.0267
Chinese (composed)	x_0 x_1 x_2 x_3	.9047
	x_0 x_1 , x_2 x_3	.016
	x_0 , x_1 x_2 x_3	.0083
	x_0 x_1 了 x_2 x_3	.0072
Arabic (composed)	x_1 x_0 x_2 x_3	.5874
	x_0 x_1 x_2 x_3	.4027
	x_1 x_2 x_0 x_3	.0077
	x_1 x_0 x_2 " x_3	.0001

Table 3: Our model transforms English subject-verb-object (SVO) structures into Chinese SVO and into Arabic VSO. With only minimal rules, Chinese VSO is wrongly preferred.

extracted the most probable one (Viterbi) for each sentence pair (based on an automatic alignment produced by GIZA). We noticed in general that Viterbi derivations according to C_4 make extensive usage of composed rules, as it is the case in the example in Figure 4. It shows the best derivation according to C_m and C_4 on the unseen (π,**f**,**a**) triple displayed on the right. The second derivation ($\log p = -11.6$) is much more probable than the minimal one ($\log p = -17.7$). In the case of C_m, we can see that many small rules must be applied to explain the transformation, and at each step, the decision regarding the re-ordering of constituents is made with little syntactic context. For example, from the perspective of a decoder, the word *by* is immediately transformed into a preposition (IN), but it is in general useful to know which particular function word is present in the sentence to motivate good re-orderings in the up-

***lhs*$_1$**: NP-C(x_0:NPB PP(IN(*of*) x_1:NP-C))	(NP-*of*-NP)
***lhs*$_2$**: PP(IN(*of*) NP-C(x_0:NPB PP(IN(*of*) NP-C(x_1:NPB x_2:VP))))	(*of*-NP-*of*-NP-VP)
***lhs*$_3$**: VP(VBD(*said*) SBAR-C(IN(*that*) x_0:S-C))	(*said-that*-S)
***lhs*$_4$**: SBAR(WHADVP(WRB(*when*)) S-C(x_0:NP-C VP(VBP(*are*) x_1:VP-C)))	(*when*-NP-*are*-VP)

rhs_{1i}	$p(rhs_{1i}\|$***lhs*$_1$**$)$	rhs_{2i}	$p(rhs_{2i}\|$***lhs*$_2$**$)$	rhs_{3i}	$p(rhs_{3i}\|$***lhs*$_3$**$)$	rhs_{4i}	$p(rhs_{4i}\|$***lhs*$_4$**$)$
x_1 x_0	.54	x_2 的 x_1 的 x_0	.6754	说 , x_0	.6062	在 x_1 x_0 时	.6618
x_0 x_1	.2351	在 x_2 的 x_1 的 x_0	.035	说 x_0	.1073	当 x_1 x_0 时	.0724
x_1 的 x_0	.0334	x_2 的 x_1 的 x_0 ,	.0263	表示 , x_0	.0591	在 x_1 x_0 时 ,	.0579
x_1 x_0 的	.026	x_2 的 x_1 的 x_0 有	.0116	他 说 , x_0	.0234	, 在 x_1 x_0 时	.0289

Table 4: Translation probabilities promote linguistically motivated constituent re-orderings (for lhs_1 and lhs_2), and enable non-constituent (lhs_3) and non-contiguous (lhs_4) phrasal translations.

per levels of the tree. A rule like **(e)** is particularly unfortunate, since it allows the word *were* to be added without any other evidence that the VP should be in passive voice. On the other hand, the composed-rule derivation of C_4 incorporates more linguistic evidence in its rules, and re-orderings are motivated by more syntactic context. Rule **(p)** is particularly appropriate to create a passive VP construct, since it expects a Chinese passive marker (被), an NP-C, and a verb in its rhs, and creates the *were ... by* construction at once in the left hand side.

5.1 Syntactic translation tables

We evaluate the promise of our SBTM by analyzing instances of translation tables (t-table). Table 3 shows how a particular form of SVO construction is transformed into Chinese, which is also an SVO language. While the t-table for Chinese composed rules clearly gives good estimates for the "correct" x_0 x_1 ordering ($p = .9$), i.e. subject before verb, the t-table for minimal rules unreasonably gives preference to verb-subject ordering (x_1 x_0, $p = .37$), because the most probable transformation (x_0 x_1) does not correspond to a minimal rule. We obtain different results with Arabic, an VSO language, and our model effectively learns to move the subject after the verb ($p = .59$).

lhs_1 in Table 4 shows that our model is able to learn large-scale constituent re-orderings, such as re-ordering NPs in a NP-*of*-NP construction, and put the modifier first as it is more commonly the case in Chinese ($p = .54$). If more syntactic context is available as in lhs_2, our model provides much sharper estimates, and appropriately reverses the order of three constituents with high probability ($p = .68$), inserting modifiers first (possessive markers 的 are needed here for better syntactic disambiguation).

A limitation of earlier syntax-based systems is their poor handling of non-constituent phrases. Table 4 shows that our model can learn rules for such phrases, e.g., *said that* (lhs_3). While the *that* has no direct translation, our model effectively learns to separate 说 (said) from the relative clause with a comma, which is common in Chinese.

Another promising prospect of our model seems to lie in its ability to handle non-contiguous phrases, a feature that state of the art systems such as (Och and Ney, 2004) do not incorporate. The *when*-NP-*are*-VP construction of lhs_4 presents such a case. Our model identifies that *are* needs to be deleted, that *when* translates into the phrase 在 ... 时, and that the NP needs to be moved after the VP in Chinese ($p = .66$).

6 Empirical evaluation

The task of our decoder is to find the most likely English tree π that maximizes all models involved in Equation 2. Since **xRs** rules can be converted to context-free productions by increasing the number of non-terminals, we implemented our decoder as a standard CKY parser with beam search. Its rule binarization is described in (Zhang et al., 2006).

We compare our syntax-based system against an implementation of the alignment template (AlTemp) approach to MT (Och and Ney, 2004), which is widely considered to represent the state of the art in the field. We registered both systems in the NIST 2005 evaluation; results are presented in Table 5. With a difference of 6.4 BLEU points for both language pairs, we consider the results of our syntax-based system particularly promising, since these are the highest scores to date that we know of using linguistic syntactic transformations. Also, on the one hand, our AlTemp system represents quite mature technology, and incorporates highly tuned model parameters. On the other hand, our syntax decoder is still work in progress: only one model was used during search, i.e., the EM-trained root-normalized SBTM, and as yet no language model is incorporated in the search (whereas the search in the AlTemp system uses two phrase-based translation models and

	Syntactic	AlTemp
Arabic-to-English	40.2	46.6
Chinese-to-English	24.3	30.7

Table 5: BLEU-4 scores for the 2005 NIST test set.

	C_m	C_3	C_4
Chinese-to-English	24.47	27.42	28.1

Table 6: BLEU-4 scores for the 2002 NIST test set, with rules of increasing sizes.

12 other feature functions). Furthermore, our decoder doesn't incorporate any syntax-based language model, and admittedly our ability to penalize ill-formed parse trees is still limited.

Finally, we evaluated our system on the NIST-02 test set with the three different rule sets (see Table 6). The performance with our largest rule set represents a 3.63 BLEU point increase (14.8% relative) compared to using only minimal rules, which indicates positive prospects for using even larger rules. While our rule inference algorithm scales to higher thresholds, one important area of future work will be the improvement of our decoder, conjointly with analyses of the impact in terms of BLEU of contextually richer rules.

7 Related work

Similarly to (Poutsma, 2000; Wu, 1997; Yamada and Knight, 2001; Chiang, 2005), the rules discussed in this paper are equivalent to productions of synchronous tree substitution grammars. We believe that our tree-to-string model has several advantages over tree-to-tree transformations such as the ones acquired by Poutsma (2000). While tree-to-tree grammars are richer formalisms that provide the potential benefit of rules that are linguistically better motivated, modeling the syntax of both languages comes as an extra cost, and it is admittedly more helpful to focus our syntactic modeling effort on the target language (e.g., English) in cases where it has syntactic resources (parsers and treebanks) that are considerably more available than for the source language. Furthermore, we think there is, overall, less benefit in modeling the syntax of the source language, since the input sentence is fixed during decoding and is generally already grammatical.

With the notable exception of Poutsma, most related works rely on models that are restricted to synchronous context-free grammars (SCFG). While the state-of-the-art hierarchical SMT system (Chiang, 2005) performs well despite stringent constraints imposed on its context-free grammar, we believe its main advantage lies in its ability to extract hierarchical rules across phrasal boundaries. Context-free grammars (such as Penn Treebank and Chiang's grammars) make independence assumptions that are arguably often unreasonable, but as our work suggests, relaxations of these assumptions by using contextually richer rules results in translations of increasing quality. We believe it will be beneficial to account for this finding in future work in syntax-based SMT and in efforts to improve upon (Chiang, 2005).

8 Conclusions

In this paper, we developed probability models for the multi-level transfer rules presented in (Galley et al., 2004), showed how to acquire larger rules that crucially condition on more syntactic context, and how to pack multiple derivations, including interpretations of unaligned words, into derivation forests. We presented some theoretical arguments for not limiting extraction to minimal rules, validated them on concrete examples, and presented experiments showing that contextually richer rules provide a 3.63 BLEU point increase over the minimal rules of (Galley et al., 2004).

Acknowledgments

We would like to thank anonymous reviewers for their helpful comments and suggestions. This work was partially supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022.

References

D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL*.

H. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proc. of EMNLP*, pages 304–311.

M. Galley, M. Hopkins, K. Knight, and D. Marcu. 2004. What's in a translation rule? In *Proc. of HLT/NAACL-04*.

J. Graehl and K. Knight. 2004. Training tree transducers. In *Proc. of HLT/NAACL-04*, pages 105–112.

F. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

A. Poutsma. 2000. Data-oriented translation. In *Proc. of COLING*, pages 635–641.

D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.

K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL*, pages 523–530.

H. Zhang, L. Huang, D. Gildea, and K. Knight. 2006. Synchronous binarization for machine translation. In *Proc. of HLT/NAACL*.