Breaking the barrier of context-freeness. Towards a linguistically adequate probabilistic dependency model of parallel texts.

Matthias Buch-Kromann
ISV Computational Linguistics Group
Copenhagen Business School
DK-2000 Frederiksberg, Denmark
mbk.isv@cbs.dk

Abstract

This paper presents a generative probabilistic dependency model of parallel texts that can be used for statistical machine translation and parallel parsing. Unlike syntactic models that are based on context-free dependency grammars, the dependency model proposed in this paper is based on a sophisticated notion of dependency grammar that is capable of modelling non-projective word order and island constraints, the complement-adjunct distinction, as well as deletions and additions in translations.

1 Introduction

Dependency grammar has attracted much attention in computational linguistics in recent years. In statistical machine translation, several researchers have proposed SMT systems that are based on dependency grammars, including (Fox, 2005; Quirk et al., 2005; Ding, 2006; Smith and Eisner, 2006; Hall and Němec, 2007). However, the dependency-based SMT systems that have been proposed in the literature are almost uniformly based on projective (usually context-free) dependency grammars, ie, grammars that disallow the kind of crossing dependencies shown in Figure 1 and explained in section 3.

From a linguistic point of view, the projectivity assumption is unfortunate because non-projectivity is a high-frequent phenomenon that manifests itself in long-distance phenomena such as topicalization, scrambling, and extraposition.

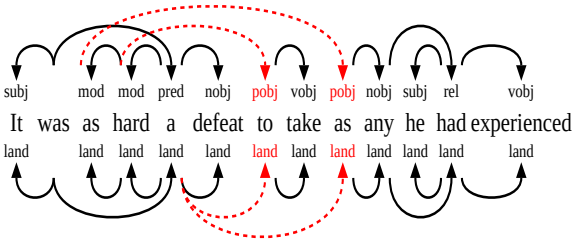

Figure 1: Authentic example with a doubly non-projective dependency tree and corresponding surface structure. Dependency and landing edges for non-projective nodes are shown with dashes.

Eg, in the dependency treebanks for Slovene, Arabic, Dutch, Czech, and Danish, 0.4–5.4% of all dependencies are non-projective, and 11.2–36.4% of all sentences contain a non-projective dependency (Nilsson et al., 2007). Since it is difficult to model non-projective word orders correctly with projective syntax models, and such errors often result in meaning-disturbing translation errors, non-projectivity is more important than its relatively small contribution to precision and recall in monolingual parsing suggests. (Buch-Kromann, 2006, sections 1.4, 2.4, 4.2) gives a more comprehensive list of linguistic constructions that are difficult to model within a projective setting.

Within a monolingual setting, there are many dependency frameworks that account for most of these phenomena, including Word Grammar (Hudson, 2007), Functional-Generative Description (Sgall et al., 1986), Weighted Constraint Dependency Grammar (Schröder, 2002), Extensible Dependency Grammar (Debusmann et al., 2004), and Discontinuous Grammar (Buch-Kromann, 2006). But, as far as we know, none of these de-

pendency frameworks have so far provided a linguistically well-motivated non-projective dependency framework for parallel texts, and done so within a probabilistic setting. This is a gap that we hope to fill with the present paper.

The paper is structured as follows. In section 2, we describe how machine translation and parallel parsing can be viewed as optimization problems within a generative probabilistic dependency model of parallel texts. In section 3, we describe our notion of parallel dependency analyses and how they are used to control word order. In section 4, we introduce our notion of translation units. In section 5, we describe our generative probabilistic dependency model of parallel texts. In section 6, we briefly outline some ideas for how grammar induction can be carried out within our framework. Section 7 presents our conclusions.

2 Statistical dependency-based translation and parallel parsing

From an abstract point of view, a *parallel probabilistic dependency grammar* can be viewed as a probability measure $P(\mathcal{A})$ on the space $\mathbb{A}$ of all conceivable parallel dependency analyses. In this setting, machine translation and parallel parsing can be reduced to the problem of optimizing $P(\mathcal{A})$ with different side conditions.

In *translation*, we know a source text t and need to find the most probable parallel dependency analysis, $\mathrm{Trans}(t)$, that matches t. That is, we must find:

$$\mathrm{Trans}(t) = \arg\max_{\substack{\mathcal{A}\in\mathbb{A} \\ Y(\mathcal{A})=t}} P(\mathcal{A})$$

where $Y(\mathcal{A})$ denotes the source text associated with $\mathcal{A}$, and $Y'(\mathcal{A})$ the target text. Once we have computed $\mathrm{Trans}(t)$, it is easy to compute the optimal translation by extracting the target text from $\mathrm{Trans}(t)$ by means of Y'.[1]

Similarly, in *parallel (synchronous) parsing* — which is essential for turning a parallel corpus into a parallel dependency treebank — we know a source text t and a target text t', and need to find the most probable parallel dependency analysis, $\mathrm{Parse}(t, t')$, that matches the given source and target texts t, t'. That is, we must find:

$$\mathrm{Parse}(t, t') = \arg\max_{\substack{\mathcal{A}\in\mathbb{A} \\ Y(\mathcal{A})=t \\ Y'(\mathcal{A})=t'}} P(\mathcal{A}).$$

In our generative probability model, we assume that a parallel dependency analysis $\mathcal{A}$ consists of a source text analysis D, a target text analysis D', and a word alignment W. We will factor:

$$P(\mathcal{A}) = P(D, D', W) = P(D) \cdot P(D', W|D)$$

and model the monolingual source analysis probability $P(D)$ and the translation probability $P(D', W|D)$ separately. Note that unlike the probability model in phrase-based SMT (Koehn et al., 2003), where the source text is generated from the target text, our probability model follows the natural direction of translation. This is also the approach used in the probability model by (Smith and Eisner, 2006), but for projective rather than non-projective dependency grammars.

The asymmetry between source and target language in our model is sensible from a linguistic point of view, since it is well-known among translation scholars that translations tend to differ significantly from normal texts in the target language. This asymmetry means that our translation model resembles a transfer-based system in important respects. However, unlike traditional transfer systems, the model does not require the parallel parser or translation system to make a hard choice about the source language analysis before deciding on a target language analysis.

Several problems must be solved in order to build a functioning parallel parser or machine translation system that uses these ideas to circumvent the linguistic limitations of projective dependency grammars: we must (a) formulate a linguistically sensible notion of parallel dependency analyses and parallel probabilistic dependency grammars; (b) specify a method for inducing such grammars from parallel corpora and/or parallel dependency treebanks; and (c) identify computationally efficient optimization algorithms

[1] In the SMT literature, the translation t' of t is often defined as the target text t' that maximizes $P(t'|t) = \sum_{\mathcal{A}\in\mathbb{A}\text{ s.t. }Y(\mathcal{A})=t, Y'(\mathcal{A})=t'} P(\mathcal{A}|t)$. From a linguistic point of view, there is no solid argument for preferring one definition over the other, and by looking for the optimal parallel analysis rather than the optimal target text, we avoid the computationally difficult problem of calculating the sum.

for translation and parallel parsing that normally succeed in finding optimal or near-optimal translations and parallel parses. This paper focuses on (a), and largely ignores (b) and (c). More information about our solution to (b) and (c) is presented in (Buch-Kromann, 2007a; Buch-Kromann, 2007b). Our analyses are based on the dependency framework Discontinuous Grammar (Buch-Kromann, 2006).

3 Parallel dependency analyses

In a parallel dependency analysis $\mathcal{A} = (D, D', W)$, each word alignment $w \leftrightarrow w'$ in W is assumed to encode a translational correspondence between the word clusters w and w' in the source text and target text, ie, the word alignment encodes the intuition that the subset w of words in the source text corresponds roughly in meaning or function to the subset w' of words in the target text. The translations may contain additions or deletions, ie, w and w' may be empty.

The monolingual dependency analyses D and D' are assumed to consist of dependency edges linking the words in the text. Each dependency edge $d \xleftarrow{r} g$ encodes a complement or adjunct relation between a word g (the *governor*) and a complement or adjunct phrase headed by the word d (the *dependent*), where the edge label r specifies the complement or adjunct dependency role.[2] In our analyses, the dependencies in the source analysis are required to form a tree (or a forest), and similarly with the dependencies in the target analysis. Moreover, our parallel dependency analyses must be well-formed with respect to translation units, in a sense that is described briefly in section 4 and defined formally in (Buch-Kromann, 2007a).

Figure 2 shows an example of this kind of analysis, based on the annotation conventions used in the Copenhagen Danish-English Dependency Treebank (Buch-Kromann, 2007a). In the example, word alignments are indicated by lines connecting Danish word clusters with English word

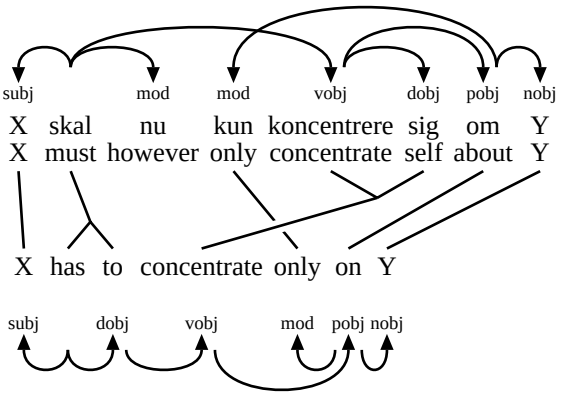

Figure 2: Parallel dependency treebank analysis with word alignment and two monolingual dependency analyses (with non-projective word order).

clusters, and dependencies are indicated by means of arrows that point from the governor to the dependent, with the dependency role written at the arrow tip. For example, the Danish word cluster "koncentrere sig" ("concentrate self") has been aligned with the English word "concentrate", and the English phrase headed by "on" is analyzed as a prepositional object of the verb "concentrate."[3]

In order to model word order and island constraints, each word w in the source and target dependency trees is assigned a *landing site* l, defined as the lowest transitive governor of w that dominates all words between w and l; a node w that has l as its landing site is called a *landed node* of l, and the landing relation between w and l is encoded by means of a *landing edge* $w \xleftarrow{\text{land}} l$. If the governor g and landing site l of a word w do not coincide ($g \neq l$), then the dependency edge $w \leftarrow g$ is called *non-projective*; otherwise, it is called *projective*. In projective dependency grammars, we always have $g = l$. Figure 1 shows an example of a dependency tree with two non-projective dependency edges ('to $\xleftarrow{\text{pobj}}$ hard' and 'as $\xleftarrow{\text{pobj}}$ as'). The word "a" functions as the landing site for both "hard" and "as" because it is the lowest transitive governor that dominates all the nodes between these two words and their respec-

[2] Following standard dependency theoretic assumptions, we assume: (a) complements are lexically licensed by their governor, whereas adjuncts license their adjunct governor; (b) in the functor-argument structure, complements act as arguments of their governor, whereas adjuncts act as modifiers; (c) a governor can have several adjuncts with the same adjunct role, whereas complement roles must be unique.

[3] Dependency analyses differ from phrase-structure analyses in that phrases are a derived notion: in a dependency tree, each word has a derived phrase that consists of all the words that can be reached from the word by following the arrows. For example, the English word "concentrate" heads the phrase "concentrate only on Y," and the Danish word "om" heads the discontinuous phrase "kun . . . om Y."

tive governors.

It can be shown that the landing edges associated with a dependency tree always form a projective tree, called the *surface tree*. The projectivity allows landing sites to control the global word order by controlling the local relative word order of their landed nodes — ie, landing sites have the word ordering responsibility assigned to governors in projective dependency grammars.

The *extraction path* for a word w is defined as the shortest path from the governor g to the landing site l of w. For example, in Figure 1, the word "to" has extraction path 'hard $\xleftarrow{\text{mod}}$ a', and the second "as" ("as$_2$") has extraction path 'as$_1$ $\xleftarrow{\text{mod}}$ hard $\xleftarrow{\text{mod}}$ a'. As argued by (Buch-Kromann, 2006, p. 98), extraction paths are useful for modelling island constraints in a dependency-based setting. For example, the adjunct island constraint states that nothing may be moved out of an adverbial adjunct, which corresponds to the claim that an extraction path cannot contain an adjunct edge of the form $x \leftarrow y$ where y is a verb.

4 Syntactic translation units[4]

In order to define our notion of syntactic translation units, we need to introduce the following terminology. The definitions below apply to both source and target words and dependencies. Two words are said to be *coaligned* if they belong to the same alignment edge. A dependency edge $d \xleftarrow{r} g$ is called *internal* if d and g are coaligned, and *external* otherwise. A word w is called *singular* if it fails to be coaligned with at least one word in the other language. By an abuse of terminology, we will say that a word d is a *dependent* of an alignment edge $w \leftrightarrow w'$ provided d is a dependent of some word in $w \cup w'$ and d is not itself contained in $w \cup w'$. For example, in Figure 2, the words "has", "to", and "skal" are coaligned, the dependency 'to $\xleftarrow{\text{dobj}}$ has' is internal, the dependency 'concentrate $\xleftarrow{\text{vobj}}$ to' is external, the word "nu" is singular, and the word "X" is a dependent of the alignment edge "skal $\leftrightarrow$ has to".

The *translation unit* corresponding to the word alignment $w \leftrightarrow w'$ is defined as the subgraph of the analysis $\mathcal{A}$ consisting of all nodes in $w \cup w'$, all internal dependency and alignment edges within $w \leftrightarrow w'$, and all external dependencies of $w \leftrightarrow w'$ except for parallel and singular adjuncts, where the external dependents are replaced with argument variables $x_1, \ldots, x_n$ and $x'_1, \ldots, x'_{n'}$. Figure 3 shows the six translation units that can be derived from the parallel dependency analysis in Figure 2 in this way. Each translation unit can be interpreted as a bidirectional translation rule: eg, the first translation unit in Figure 3 can be interpreted as a translation rule stating that a Danish dependency tree with terminals "x_1 skal x_2" can be translated into an English dependency tree with terminals "x'_1 has to x'_2" where the English phrases x'_1, x'_2 are translations of the Danish phrases x_1, x_2, and vice versa.

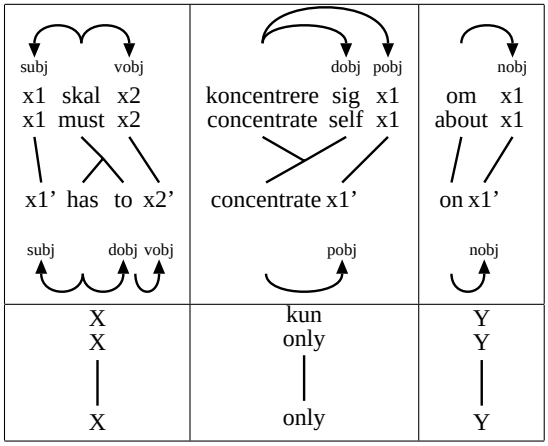

Figure 3: The six translation units derived from the parallel dependency analysis in Figure 2.

In order to have a meaningful interpretation as a translation rule, a translation unit must have a parallel set of source and target argument variables, and a well-formed source and target dependency analysis, as defined formally in (Buch-Kromann, 2007a). In general, parallel dependency treebanks are not guaranteed to lead to translation units that satisfy these requirements. However, (Buch-Kromann, 2007a) has defined an algorithm that can compute a *minimal reduction* that is computed by merging word alignments in a minimal way, in which the resulting translation units satisfy the requirements. As an example of how this procedure works, Figure 4 shows a head-switching example (left) borrowed from (Way, 2001), and the corresponding minimal reduction (right) computed by the merging algo-

[4]This section is based on (Buch-Kromann, 2007a).

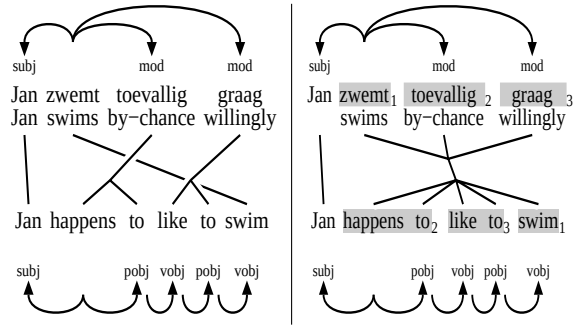

Figure 4: A head-switching example (left) and the associated minimal reduction (right).

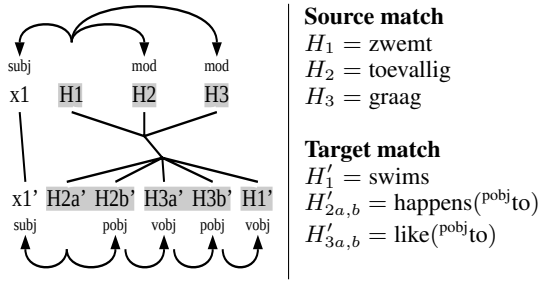

Figure 5: Syntactic translation template induced from Figure 4, with source and target match.

procedure probabilistic graph generation
begin
recursively expand source root TOP (cf. Figure 7)
recursively translate source root TOP (cf. Figure 8)
return *generated graph and probability*
end

Figure 6: Our probabilistic graph generation procedure (a Markov process).

Top-down expansion of source node w_i
S1. Identify landing site and relative word order
S2. Select complement frame
S3. Generate and recursively expand complements
S4. Generate and recursively expand adjuncts

Figure 7: The steps in the top-down expansion of a source word w_i in our generative probabilistic dependency model.

rithm, with the original word alignments indicated by means of the numbered boxes. We can think of the original word alignments in the treebank as *lexical translation units* (the smallest lexically meaningful units of translation), and of the merged word alignments as *syntactic translation units* (the smallest syntactically meaningful units of translation).

In this paper, we will for simplicity assume that each syntactic translation unit consists of a single lexical translation unit. However, a more elegant and general account of head-switching phenomena can be provided by decomposing syntactic translation units by means of their original lexical translation units. Eg, instead of using zwemt($^{\text{mod}}$toevallig,$^{\text{mod}}$graag) $\leftrightarrow$ happens($^{\text{pobj}}$to($^{\text{vobj}}$like($^{\text{pobj}}$to($^{\text{vobj}}$swim)))) as an atomic lexical translation unit in the translation of the example in Figure 4, we can decompose the translation into several steps by first matching the source analysis with the abstract syntactic translation template shown in Figure 5, and then deciding on the choice of lexical translation units in a target language top-down manner: ie, we first select "toevallig $\leftrightarrow$ happen($^{\text{pobj}}$to)" as a translation of "toevallig", then "graag $\leftrightarrow$ like($^{\text{pobj}}$to)", and finally "zwemt $\leftrightarrow$ swim."

5 A generative probabilistic dependency model of parallel texts

We will now present a generative probabilistic dependency model of parallel texts that models complements, adjuncts, landing sites, local word order, island constraints, and additions and deletions during translation. The source dependency model is a simplification of (Buch-Kromann, 2006, ch. 6) in that we ignore secondary dependencies, gapping coordinations, antecedents, and punctuation. We assume that the source and target analyses have formal root nodes TOP and TOP′ (aligned with each other), and that all words in the source and target text are transitive dependents of the top nodes; in particular, the root of a sentence in the source and target analysis is assumed to be a *root* adjunct of the top node.

The generative procedure is modelled as a top-down Markov process (Figure 6). The generative procedure first creates the source tree by recursively expanding TOP in steps S1–S4, and then creates the target tree and the word alignments by recursively translating TOP in steps T1–T5. The individual steps in the source and target node expansion are shown in Figures 7 and 8, and described in detail below. In our dependency model, the probability of a parallel dependency analysis

Top-down translation of source node w_i
T1. Identify landing sites and word order in target tunit
T2. Generate and recursively expand tunit arguments
T3. Identify deleted source adjuncts
T4. Generate and recursively translate parallel adjuncts
T5. Generate added target adjuncts

Figure 8: The steps in the top-down translation of a source word w_i in our generative probabilistic dependency model.

Notation	Meaning
w_i	ith word (source > 0, target < 0)
d_i	dependency role of ith word
$cframe_i$	complement frame at w_i
$aframe_i$	adjunct roles at w_i
g_i	governor of w_i
l_i	landing site of w_i
o_i	relative word order of w_i at l_i
path(w_i, w_j)	upwards path from node w_i to transitive governor w_j
τ_i	syntactic translation unit for w_i
S_i	source analysis for τ_i
T_i	target analysis for τ_i
$w_{i'}$	target root of τ_i
int_i	internal source nodes in τ_i
int'_i	internal target nodes in τ_i
$extadj_i$	external adjuncts of τ_i
$added_i$	added external target adjuncts of τ_i
$args_i$	source arguments of τ_i

Figure 9: The notation used to refer to the governor, landing site, word order, etc. of a source or target node N.

$\mathcal{A}$ is computed by

$$P(\mathcal{A}) = P_{\text{S}}(\text{TOP}) \cdot P_{\text{T}}(\text{TOP})$$

where P_{S} and P_{T} are defined recursively by $P_{\text{S}}(w_i) = P_{\text{S1}}(w_i) \cdots P_{\text{S4}}(w_i)$ and $P_{\text{T}}(w_i) = P_{\text{T1}}(w_i) \cdots P_{\text{T5}}(w_i)$, using the probabilities for steps S1–S4 and T1–T5 defined below.

In the following, given a source or target node w_i (with source nodes having $i > 0$, target nodes $i < 0$), we will use the notation shown in Figure 9. By an abuse of notation, we will use w_i^* to denote the set of all relevant covariates associated with w_i when w_i is expanded or translated; the covariates may include any aspects of the structure that have been generated at the given point in the generation, including (but not necessarily restricted to) all relevant node features and dependency roles of w_i, l_i, g_i, etc. Determining the right set of covariates for each of the distributions in our model is an empirical question which we will ignore in the rest of this paper.

5.1 Modelling source analyses

The steps S1–S4 are used to model node expansion in source analyses. Steps S2–S4 are similar in spirit to the steps proposed by (Eisner, 1996; Collins, 1997) for statistical dependency parsing, whereas the submodel S1 for island constraints and local word order is new.

S1. Identify landing site and word order

The first step in the source expansion of w_i is to choose a landing site l_i among the transitive governors of w_i, and a linear ordering o_i that indicates the word order of w_i at l_i relative to the previously landed nodes at l_i.[5] For each possible landing site l and word order o, we want to quantify how well-formed that choice of landing site and word order is with respect to (a) island constraints expressed in terms of the extraction path from g_i to l, and (b) the local word order position o assigned to w_i at l.

As noted in section 3, an extraction can be blocked by the presence of island edges in the extraction path (eg, adjunct edges with verbal governors). Island edges can be detected statistically by observing that if an edge $x \xleftarrow{r} y$ occurs less often in extraction paths than in the treebank in general, then the edge is likely to be an island edge, ie, the blocking effect of an edge $x \xleftarrow{r} y$ for the word w_i can be modelled by means of:

$$\min\left(1, \frac{P_{\text{extpath}}(x \xleftarrow{r} y | w_i)}{P_{\text{deptree}}(x \xleftarrow{r} y)}\right)$$

where the minimum ensures that non-island edges cannot improve the global extraction probability. P_{extpath} is the probability distribution of edges in extraction paths, and P_{deptree} is the probability distribution of edges in dependency trees. Ie, the relative probability $E_{w_i,l}$ of the extraction path produced by choosing l as the landing site of w_i is expressed by:

$$E_{w_i,l} = \prod_{\substack{(x \xleftarrow{r} y) \in \\ \text{path}(g_i,l)}} \min\left(1, \frac{P_{\text{extpath}}(x \xleftarrow{r} y | w_i)}{P_{\text{deptree}}(x \xleftarrow{r} y)}\right)$$

[5] Following (Buch-Kromann, 2006, pp. 276-277), we assume that dependencies are generated in a predefined derivation order. Nodes that precede the current landed node in the derivation order are called *previously landed nodes*.

In order to model the probability of the local word order position, we note that the choice of local word order o for w_i at l can be modelled as a process where w_i is inserted at position o, and the dummy node STOP is inserted at all other positions so that we can detect the absence of an obligatorily present node. If we let $P_{\text{worder}}(w|c_{l,o})$ denote the probability of inserting word or dummy word w as a landed node at a position o with word order context $c_{l,o}$, then the relative probability $O_{w_i,l,o}$ of the choice of local word order o for w_i at l is expressed by:

$$O_{w_i,l,o} = P_{\text{worder}}(w|c_{l,o}) \prod_{o' \neq o} P_{\text{worder}}(\text{STOP}|c_{l,o'})$$

(Buch-Kromann, 2006, section 6.2) has proposed a local word order context that only includes the neighbouring complements, the neighbouring adjuncts, the landing site, and a binary variable that indicates whether the position is to the left or right of the landing site. These covariates suffice to encode a wide range of local word order constraints, such as "adverbials cannot be inserted between a verb and an adjacent subject," "a verb does not allow two simultaneous complements on its left," and "a finite verb requires a subject to its left," but in probabilistic rather than absolute terms.

With the relative probability of the extraction path quantified by $E_{w_i,l}$ and the relative probability of the local word order quantified by $O_{w_i,l,o}$, we can compute the probability of the actual choice of l_i, o_i by normalizing the probabilities, ie by setting:

$$P_{\text{S1}}(w_i) = \frac{E_{w_i,l_i} \cdot O_{w_i,l_i,o_i}}{\sum_{l,o} E_{w_i,l} \cdot O_{w_i,l,o}}.$$

As argued by (Buch-Kromann, 2006, section 7.3), under linguistically reasonable assumptions about island constraints and the number of complements and adjuncts that a word can have, a landing site has a bounded number of landing positions, and a word has at most $\log n$ landing sites where n is the number of words in the graph. The sum can therefore be computed efficiently in $O(\log n)$ time.

S2. Select complement frame

In step 2 of the source expansion, we must choose a complement frame *cframe*$_i$ for w_i. This choice can be modelled by means of

$$P_{\text{S2}}(w_i) = P_{\text{cframe}}(\textit{cframe}_i|w_i^*)$$

where $P_{\text{cframe}}(\textit{cframe}|w_i^*)$ is the probability of generating the complement frame *cframe* at w_i.

S3. Generate and expand complements

In step 3 of the source expansion, we must choose a complement word w_j for each complement role d_j in *cframe*$_i$, and expand the complement recursively. We model this by:

$$P_{\text{S3}}(w_i) = \prod_{d_j \in \textit{cframe}_i} P_{\text{comp}}(w_j|d_j, w_i^*) P_{\text{S}}(w_j)$$

where $P_{\text{comp}}(w|d, w_i^*)$ is the probability of generating the complement w for complement role d at w_i.[6]

S4. Generate and expand adjuncts

In step 4 of the source expansion, we must generate the adjuncts of w_i and expand them recursively. We model this as a process where the governor generates a list of adjunct roles *aframe*$_i$ at w_i one by one with probability $P_{\text{arole}}(d_j|w_i^*)$, until the special adjunct role STOP is generated with probability $P_{\text{arole}}(\text{STOP}|w_i^*)$. As each adjunct role d_j is generated, we generate an adjunct word w_j with probability $P_{\text{adj}}(w_j|d_j, w_i^*)$ and expand w_j recursively, ie, the adjuncts of w_i are generated with probability:

$$P_{\text{S4}}(w_i) = P_{\text{arole}}(\text{STOP}|w_i^*) \cdot \prod_{d_j \in \textit{aframe}_i} P_{\text{arole}}(d_j|w_i^*) P_{\text{adj}}(w_j|d_j, w_i^*) P_{\text{S}}(w_j)$$

5.2 Modelling the translation from source analyses to target analyses

The steps T1–T5 are used to model the translation from source analyses to target analyses. Probability distributions for the target language are indicated by means of primes. Eg, $P_{\text{S1}'}(w_i)$ denotes the probability of the monolingual expansion step S1 at the target word w_i, but for the target language rather than the source language.

[6] Although we could have designed a model that can learn statistical dependencies between different complement slots, we use a simpler model where the complements are generated independently of each other. The simple model is justified by (Li and Abe, 1999), who report that the statistical dependencies between complement roles are rather weak, and therefore difficult to detect.

T1. Identify landing site and relative word order in target unit

In step T1, we must identify landing sites and relative word order for the internal target nodes int'_i in the syntactic translation unit τ_i with source root w_i. If the target word order is assumed to be completely independent of the source word order, we can simply define:

$$P_{\text{T1}}(w_i) = \prod_{w_j \in int'_i} P_{\text{S1}'}(w_j)$$

where int'_i is processed in the target language derivation order.

However, languages tend to place discourse-old material in the beginning of sentences, and discourse-new material in the end. It therefore often makes sense to use the source word order as a guide to target word order. This can be accomplished by including the relative ordering of the source nodes corresponding to the target nodes within the target word order context $c'_{l,o}$.

T2. Generate and translate tunit arguments

In step T2, we need to recursively translate the source arguments $args_i$ of the translation unit τ_i. For each $w_j \in args_i$ we select a translation unit τ_j that matches the source analysis at w_j. Like in noisy-channel SMT, we must balance the adequacy A and fluency F of our choice of τ_j at w_j, ie, we must try to find a compromise between the admissibility of τ_j as a translation of the source tree in τ_j (*adequacy*) and the admissibility of the target tree in τ_j as a target subtree at the target root $w_{j'}$ of τ_j (*fluency*).

We can model the adequacy of τ_j as a translation of the source tree at w_j by means of the probability:

$$A(w_j, \tau_j) = P_{\text{tunit}}(\tau_j | w_j^*)$$

where $P_{\text{tunit}}(\tau | w_j^*)$ is the probability of translating a source structure at w_j by means of the matching translation unit τ.

Similarly, we can model the fluency of the source tree T_j at the target root $w_{j'}$ by means of the probability:

$$F(w_j, \tau_j) = P_{\text{comp}'/\text{adj}'}(w_{j'} | d_{j'}, w_{j'}^*) \cdot P_{\text{S}'_{234}}(T_j)$$

where $P_{\text{comp}'/\text{adj}'}$ denotes either $P_{\text{comp}'}$ or $P_{\text{adj}'}$, depending on whether $w_{j'}$ is a complement or an adjunct, and where $P_{\text{S234}'}(T_j)$ denotes the monolingual target language probability of the target dependency tree T_j without any word order (ie, steps S2′–S4′ only).

Like in noisy-channel SMT, we can compromise between adequacy and fluency by weighing them by means of the formula $A^\lambda F^{1-\lambda}$ for some $\lambda \in [0, 1]$. Setting λ close to 1 results in translations with high adequacy and low fluency, and vice versa when setting λ close to 0. We can therefore model the probability of choosing the translation unit τ_j to transfer the source tree at w_j by means of:

$$P_{\text{transfer}}(w_j, \tau_j) = \frac{A(w_j, \tau_j)^\lambda F(w_j, \tau_j)^{1-\lambda}}{\sum_\tau A(w_j, \tau)^\lambda F(w_j, \tau)^{1-\lambda}}.$$

This allows us to model:

$$P_{\text{T2}}(w_i) = \prod_{w_j \in args_i} P_{\text{transfer}}(w_j, \tau_j) P_{\text{T}}(w_j).$$

T3. Identify deleted source adjuncts

In step T3, we need to decide for each external source adjunct w_j in $extadj_i$ whether w_j should be deleted in the translation ($\delta_j = 1$) or translated into the target language ($\delta_j = 0$). In general, it is not a good idea to delete content words in the translation. However, there are sometimes mismatches in the translation, and there are also some aspects of syntax, especially discourse particles and punctuation, that are language-specific and consequently often ignored in the translation. We will therefore include deletions in our model, by defining:

$$P_{\text{T3}}(w_i) = \prod_{w_j \in extadj_i} P_{\text{del}}(\delta_j | w_j^*)$$

where $P_{\text{del}}(\delta_j = 1 | w_j^*)$ is the probability of deleting the adjunct w_j in the translation.

T4. Generate and translate parallel adjuncts

For each non-deleted external source adjunct w_j in $extadj_i$ (ie, each w_j where $\delta_j = 0$), we need to (a) select a target adjunct role $d_{j'}$ and a target adjunct governor $g_{j'}$ within the target tree T_i, (b) select a translation unit τ_j that matches the source analysis at w_j, and (c) expand w_j recursively.

In step T4a, we want to quantify the probability of the chosen target adjunct governor $g_{j'}$ and role $d_{j'}$, given the corresponding source adjunct governor g_j and role d_j. The relative probability of a particular choice (g', d') can be modelled statistically by assigning large weight to choices of (g', d') that occur above chance level, and low weight to choices that occur below chance level, ie, the relative probability of the choice (g', d') can be expressed by the quantity

$$I_{g',d'|g,d} = \frac{P_{\text{adjtrans}}(g', d'|g, d)}{P_{\text{adjtrans}}(g', d')}$$

where $P_{\text{adjtrans}}(g', d')$ is the probability that a parallel adjunct has target governor g' and target role d', and $P_{\text{adjtrans}}(g', d'|g, d)$ is the same probability with the conditional knowledge that the parallel adjunct has source governor g and source role d. By normalizing the weights, we can compute:

$$P_{\text{T4a}}(w_j) = \frac{I_{g_{j'},d_{j'}|g_j,d_j}}{\sum_{g',d'} I_{g',d'|g_j,d_j}}.$$

In step T4b, we must select a translation unit τ_j for each non-deleted adjunct w_j, given the target adjunct role $d_{j'}$ and target adjunct governor $g_{j'}$. This is modelled exactly as in step T2, but for non-deleted external source adjuncts rather than translation unit arguments.

Combining (a) and (b), we therefore define:

$$P_{\text{T4}}(w_i) = \prod_{\substack{w_j \in \mathit{extadj}_i \\ \delta_j = 0}} P_{\text{T4a}}(w_j) P_{\text{T4b}}(w_j) P_{\text{T}}(w_j).$$

T5. Generate added adjuncts

In step T5, we must generate the added target adjuncts in the target analysis. We do this by traversing the internal target nodes in int'_i in target derivation order: for each internal target node w_j in int'_i, we (a) generate a sequence added_j of added target adjunct phrases one at a time, until the special stop symbol STOP is generated, and (b) assign landing sites to the generated target adjunct phrases in the process.

Step T5a can therefore be computed by:

$$P_{\text{T5a}}(w_j) = P_{\text{add-arole}}(\text{STOP}|w_j^*) \cdot \prod_{w_k \in \mathit{added}_j} P_{\text{add-arole}}(d_k|w_j^*) P_{\text{add-adj}}(T_k|d_k, w_j^*)$$

where $P_{\text{add-arole}}(d|w_j^*)$ is the probability of creating an added target adjunct with adjunct role d at w_j, and $P_{\text{add-adj}}(T|d_k, w_j^*)$ is the probability of creating the added target adjunct tree T given adjunct role d_k at w_j. T5b can be computed by means of:

$$P_{\text{T5b}}(w_j) = \prod_{w_k \in \mathit{added}_j} P_{\text{T1'}}(T_k)$$

where $P_{\text{T1'}}(T_k)$ is the probability of the target landing sites assigned to the words in the target adjunct phrase T_k.

We therefore have:

$$P_{\text{T5}}(w_i) = \prod_{w_j \in \mathit{int}'_i} P_{\text{T5a}}(w_j) P_{\text{T5b}}(w_j).$$

6 Statistical estimation and optimization

Our generative probabilistic dependency model decomposes the probability of the entire analysis into probabilities associated with individual steps in the generative procedure, such as P_{cframe}, P_{extpath}, $P_{\text{add-adj}}$, etc. Each of these distributions can be estimated from parallel dependency treebank data by means of any suitable density estimator, including Generalized Linear Models and Generalized Additive Models (which have log-linear models as a special case) and the XHPM estimator proposed by (Buch-Kromann, 2006, ch. 5,6). The XHPM estimator is a generalization of (Li and Abe, 1999) that is designed specifically for categorical data equipped with classification hierarchies. As a correction estimator, the XHPM estimator may be particularly suited to estimating probability ratios of the form $P(x|y)/P(x)$, which is needed in steps S1 and T4.

7 Conclusions

In this paper, we have presented a generative probabilistic dependency model of parallel texts that can be used for machine translation and parallel parsing. Unlike previous dependency models used in machine translation, the proposed model is not based on context-free dependency grammar, but builds on a more sophisticated notion of dependency theory that is capable of modelling complements and adjuncts, non-projective dependencies and island constraints, as well as deletions and additions in the translation. In this respect, our model can be seen as a step towards

translation models that are more realistic from a linguistic point of view. By allowing syntactic translation units to be arbitrarily large parallel tree structures, and decomposing syntactic translation units into lexical translation units, the model may even provide an elegant account of head-switching.

There are many issues that need to be addressed before the dependency model we have presented can be used to build a functioning machine translation or parallel parsing system. First of all, we have not described how to estimate the many probabilities in our dependency model from parallel treebank data. Secondly, some empirical work remains to be done with respect to choosing the relevant covariates in each generative step. Finally, although (Buch-Kromann, 2007b) has started work in this direction, we still need to develop a computationally efficient algorithm that is capable of computing optimal or near-optimal solutions to the optimization problems posed by parallel parsing and machine translation.

8 Acknowledgements

This work was supported by a grant from the Danish Research Council for the Humanities. Thanks to the anonymous reviewers for their careful reviews and highly valuable advice.

References

Matthias Buch-Kromann. 2006. Discontinuous Grammar. A dependency-based model of human parsing and language learning. Dr.ling.merc. dissertation, Copenhagen Business School. http://www.id.cbs.dk/∼mbk/thesis.

Matthias Buch-Kromann. 2007a. Computing translation units and quantifying parallelism in parallel dependency treebanks. In *Proc. of Linguistic Annotation Workshop, ACL-2007*. See errata on http://www.isv.cbs.dk/∼mbk/pub/2007-law.html.

Matthias Buch-Kromann. 2007b. Dependency-based machine translation and parallel parsing without the projectivity and edge-factoring assumptions. a white paper. Working paper, ISV Computational Linguistics Group, Copenhagen Business School. URL www.isv.cbs.dk/ mbk/pub/2007-white.pdf.

Michael Collins. 1997. Three generative, lexicalized models for statistical parsing. In *Proc. ACL-1997*, pages 16–23.

Ralph Debusmann, Denys Duchier, and Geert-Jan Kruijff. 2004. Extensible Dependency Grammar: A new methodology. In *Proc. Recent Advances in Dependency Grammar, COLING-2004*.

Yuan Ding. 2006. *Machine translation using Probabilistic Synchronous Dependency Insertion Grammars*. Ph.D. thesis, Univ. of Pennsylvania.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proc. COLING-96*, pages 340–345.

Heidi J. Fox. 2005. Dependency-based statistical machine translation. In *Proc. 2005 ACL Student Workshop*.

Keith Hall and Petr Němec. 2007. Generation in machine translation from deep syntactic trees. In *Proc. CoNLL-2007*.

Richard Hudson. 2007. *Language Networks. The new Word Grammar*. Oxford University Press.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT-NAACL-2003*.

Hang Li and Naoki Abe. 1999. Learning dependencies between case frame slots. *Computational Linguistics*, 25(2):283–291.

Jens Nilsson, Joakim Nivre, and Johan Hall. 2007. Generalizing tree transformations for inductive dependency parsing. In *Proc. ACL-2007*.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proc. ACL-2005*.

Ingo Schröder. 2002. *Natural language parsing with graded constraints*. Ph.D. thesis, Univ. of Hamburg.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The meaning of the sentence in its semantic and pragmatic aspects*. Reidel, Dordrecht.

David A. Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proc. HLT-NAACL Workshop on Statistical Machine Translation*, pages 23–30.

Andy Way. 2001. Solving headswitching translation cases in LFG-DOT. In *Proc. LFG-2001*.