Statistical Machine Translation by Parsing

I. Dan Melamed
Computer Science Department
New York University
New York, NY, U.S.A.
10003-6806
{lastname}@cs.nyu.edu

Abstract

In an ordinary syntactic parser, the input is a string, and the grammar ranges over strings. This paper explores generalizations of ordinary parsing algorithms that allow the input to consist of string tuples and/or the grammar to range over string tuples. Such algorithms can infer the synchronous structures hidden in parallel texts. It turns out that these generalized parsers can do most of the work required to train and apply a syntax-aware statistical machine translation system.

1 Introduction

A parser is an algorithm for inferring the structure of its input, guided by a grammar that dictates what structures are possible or probable. In an ordinary parser, the input is a string, and the grammar ranges over strings. This paper explores generalizations of ordinary parsing algorithms that allow the input to consist of string tuples and/or the grammar to range over string tuples. Such inference algorithms can perform various kinds of analysis on parallel texts, also known as multitexts.

Figure 1 shows some of the ways in which ordinary parsing can be generalized. A **synchronous parser** is an algorithm that can infer the syntactic structure of each component text in a multitext and simultaneously infer the correspondence relation between these structures.[1] When a parser's input can have fewer dimensions than the parser's grammar, we call it a **translator**. When a parser's grammar can have fewer dimensions than the parser's input, we call it a **synchronizer**. The corresponding processes are called **translation** and **synchronization**. To our knowledge, synchronization has never been explored as a class of algorithms. Neither has the relationship between parsing and word alignment. The relationship between translation and ordinary parsing was noted a long time ago (Aho & Ullman, 1969), but here we articulate it in more detail: ordinary parsing is a special case of synchronous parsing, which is a special case of translation. This paper offers an informal guided tour of the generalized parsing algorithms in Figure 1. It culminates with a recipe for using these algorithms to train and apply a syntax-aware statistical machine translation (SMT) system.

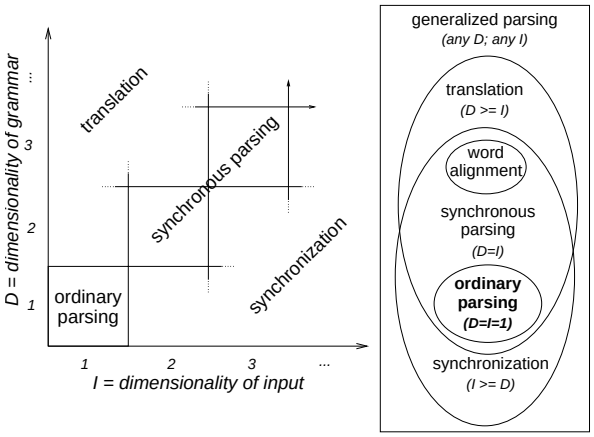


Figure 1: Generalizations of ordinary parsing.

2 Multitext Grammars and Multitrees

The algorithms in this paper can be adapted for any synchronous grammar formalism. The vehicle for the present guided tour shall be multitext grammar (MTG), which is a generalization of context-free grammar to the synchronous case (Melamed, 2003). We shall limit our attention to MTGs in Generalized Chomsky Normal Form (GCNF) (Melamed *et al.*, 2004). This normal form allows simpler algorithm descriptions than the normal forms used by Wu (1997) and Melamed (2003).

In GCNF, every production is either a terminal production or a nonterminal production. A nonterminal production might look like this:

$$\begin{array}{lll} X & [1,2] & \begin{pmatrix} \text{A} & \text{B} \\ () & C \\ \text{D(2)} & \text{E} \end{pmatrix} \\ Y \Rightarrow\bowtie & [1] & \\ Z & [1,2,1] & \end{array} \qquad (1)$$

[1] A suitable set of ordinary parsers can also infer the syntactic structure of each component, but cannot infer the correspondence relation between these structures.

There are nonterminals on the left-hand side (LHS) and in parentheses on the right-hand side (RHS). Each row of the production describes rewriting in a different component text of a multitext. In each row, a **role template** describes the relative order and contiguity of the RHS nonterminals. E.g., in the top row, [1,2] indicates that the first nonterminal (A) precedes the second (B). In the bottom row, [1,2,1] indicates that the first nonterminal both precedes and follows the second, i.e. D is discontinuous. Discontinuous nonterminals are annotated with the number of their contiguous segments, as in $D(2)$. The $\bowtie$ ("join") operator rearranges the nonterminals in each component according to their role template. The nonterminals on the RHS are written in columns called **links**. Links express translational equivalence. Some nonterminals might have no translation in some components, indicated by (), as in the 2nd row. Terminal productions have exactly one "active" component, in which there is exactly one terminal on the RHS. The other components are inactive. E.g.,

$$\begin{matrix}() \\ Y\end{matrix} \Rightarrow \begin{matrix}() \\ a\end{matrix} \tag{2}$$

The semantics of $\Rightarrow$ are the usual semantics of rewriting systems, i.e., that the expression on the LHS can be rewritten as the expression on the RHS. However, all the nonterminals in the same link must be rewritten simultaneously. In this manner, MTGs generate tuples of parse trees that are isomorphic up to reordering of sibling nodes and deletion. Figure 2 shows two representations of a tree that might be generated by an MTG in GCNF for the imperative sentence pair *Wash the dishes / Pasudu moy* . The tree exhibits both deletion and inversion in translation. We shall refer to such multidimensional trees as **multitrees**.

The different classes of generalized parsing algorithms in this paper differ only in their grammars and in their logics. They are all compatible with the same parsing semirings and search strategies. Therefore, we shall describe these algorithms in terms of their underlying logics and grammars, abstracting away the semirings and search strategies, in order to elucidate how the different classes of algorithms are related to each other. Logical descriptions of inference algorithms involve inference rules: $\frac{Y, Z}{X}$ means that X can be inferred from Y and Z. An item that appears in an inference rule stands for the proposition that the item is in the parse chart. A production rule that appears in an inference rule stands for the proposition that the production is in the grammar. Such specifications are nondeterministic: they do not indicate the order in which a parser should attempt inferences. A deterministic parsing strategy can always be chosen later, to suit the application. We presume that readers are familiar with declarative descriptions of inference algorithms, as well as with semiring parsing (Goodman, 1999).

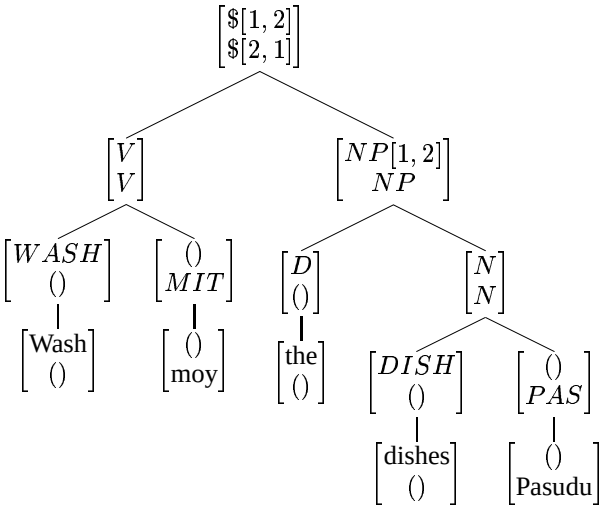
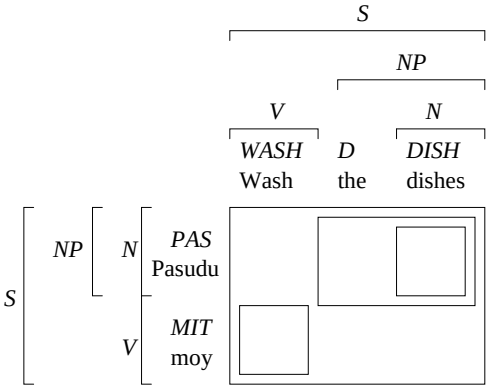


Figure 2: Above: A tree generated by a 2-MTG in English and (transliterated) Russian. Every internal node is annotated with the linear order of its children, in every component where there are two children. Below: A graphical representation of the same tree. Rectangles are 2D constituents.



3 A Synchronous CKY Parser

Figure 3 shows Logic C. Parser C is any parser based on Logic C. As in Melamed (2003)'s Parser A, Parser C's items consist of a D-dimensional label vector X_D^1 and a D-dimensional d-span vector σ_D^1.[2] The items contain d-spans, rather than ordinary spans, because

[2] Superscripts and subscripts indicate the range of dimensions of a vector. E.g., X_D^1 is a vector spanning dimensions 1 through D. See Melamed (2003) for definitions of cardinality, d-span, and the operators $+$ and $\otimes$.

Parser C needs to know all the boundaries of each item, not just the outermost boundaries. Some (but not all) dimensions of an item can be inactive, denoted $()$, and have an empty d-span $()$.

The input to Parser C is a tuple of D parallel texts, with lengths $n_1, \ldots, n_D$. The notation $(0, n_d)^1_D$ indicates that the Goal item must span the input from the left of the first word to the right of the last word in each component $d, 1 \leq d \leq D$. Thus, the Goal item must be contiguous in all dimensions.

Parser C begins with an empty chart. The only inferences that can fire in this state are those with no antecedent items (though they can have antecedent production rules). In Logic C, $G_T(\alpha;\beta)$ is the value that the grammar assigns to the terminal production $\alpha \Rightarrow \beta$. The range of this value depends on the semiring used. A *Scan* inference can fire for the ith word $w_{d,i}$ in component d for every terminal production in the grammar where $w_{d,i}$ appears in the dth component. Each *Scan* consequent has exactly one active d-span, and that d-span always has the form $(i-1, i)$ because such items always span one word, so the distance between the item's boundaries is always one.

The *Compose* inference in Logic C is the same as in Melamed's Parser A, using slightly different notation: In Logic C, the function $G_N(\alpha;\rho,\beta,\gamma)$ represents the value that the grammar assigns to the nonterminal production $\alpha \Rightarrow \bowtie \rho(\beta\gamma)$. Parser C can compose two items if their labels appear on the RHS of a production rule in the grammar, and if the contiguity and relative order of their intervals is consistent with the role templates of that production rule.

Item Form: $\left[X^1_D;\sigma^1_D\right]$ **Goal:** $\left[\$^1_D;(0,n_d)^1_D\right]$

Inference Rules

Scan component d, $1 \leq d \leq D$:

$$\frac{G_T\begin{pmatrix}()^1_{d-1} & ()^1_{d-1}\\ X & ;\ w_{d,i}\\ ()^{d+1}_D & ()^{d+1}_D\end{pmatrix}}{\begin{bmatrix}()^1_{d-1} & ()^1_{d-1}\\ X & ;\ (i-1,i)\\ ()^{d+1}_D & ()^{d+1}_D\end{bmatrix}}$$

Compose:

$$\frac{\left[Y^1_D;\tau^1_D\right]\left[Z^1_D;\sigma^1_D\right]G_N(X^1_D;\tau^1_D\otimes\sigma^1_D,Y^1_D,Z^1_D)}{\left[X^1_D;\tau^1_D+\sigma^1_D\right]}$$

Figure 3: Logic C ("C" for CKY)

These constraints are enforced by the d-span operators $+$ and $\otimes$.

Parser C is conceptually simpler than the synchronous parsers of Wu (1997), Alshawi *et al.* (2000), and Melamed (2003), because it uses only one kind of item, and it never composes terminals. The inference rules of Logic C are the multidimensional generalizations of inference rules with the same names in ordinary CKY parsers. For example, given a suitable grammar and the input (imperative) sentence pair *Wash the dishes / Pasudu moy*, Parser C might make the 9 inferences in Figure 4 to infer the multitree in Figure 2. Note that there is one inference per internal node of the multitree.

Goodman (1999) shows how a parsing logic can be combined with various semirings to compute different kinds of information about the input. Depending on the chosen semiring, a parsing logic can compute the single most probable derivation and/or its probability, the k most probable derivations and/or their total probability, all possible derivations and/or their total probability, the number of possible derivations, etc. All the parsing semirings catalogued by Goodman apply the same way to synchronous parsing, and to all the other classes of algorithms discussed in this paper.

The class of synchronous parsers includes some algorithms for word alignment. A translation lexicon (weighted or not) can be viewed as a degenerate MTG (not in GCNF) where every production has a link of terminals on the RHS. Under such an MTG, the logic of word alignment is the one in Melamed (2003)'s Parser A, but without *Compose* inferences. The only other difference is that, instead of a single item, the Goal of word alignment is any set of items that covers all dimensions of the input. This logic can be used with the expectation semiring (Eisner, 2002) to find the maximum likelihood estimates of the parameters of a word-to-word translation model.

An important application of Parser C is parameter estimation for probabilistic MTGs (PMTGs). Eisner (2002) has claimed that parsing under an expectation semiring is equivalent to the Inside-Outside algorithm for PCFGs. If so, then there is a straightforward generalization for PMTGs. Parameter estimation is beyond the scope of this paper, however. The next section assumes that we have an MTG, probabilistic or not, as required by the semiring.

4 Translation

A D-MTG can guide a synchronous parser to infer the hidden structure of a D-component multitext. Now suppose that we have a D-MTG and an input multitext with only I components, $I < D$.

1. $\dfrac{G_T\begin{pmatrix} WASH & ; & Wash \\ () & ; & () \end{pmatrix}}{\begin{bmatrix} WASH & ; & (0,1) \\ () & ; & () \end{bmatrix}}$ 2. $\dfrac{G_T\begin{pmatrix} D & ; & the \\ () & ; & () \end{pmatrix}}{\begin{bmatrix} D & ; & (1,2) \\ () & ; & () \end{bmatrix}}$

3. $\dfrac{G_T\begin{pmatrix} DISH & ; & dishes \\ () & ; & () \end{pmatrix}}{\begin{bmatrix} DISH & ; & (2,3) \\ () & ; & () \end{bmatrix}}$

4. $\dfrac{G_T\begin{pmatrix} () & ; & () \\ PAS & ; & Pasudu \end{pmatrix}}{\begin{bmatrix} () & ; & () \\ PAS & ; & (0,1) \end{bmatrix}}$ 5. $\dfrac{G_T\begin{pmatrix} () & ; & () \\ MIT & ; & moy \end{pmatrix}}{\begin{bmatrix} () & ; & () \\ MIT & ; & (1,2) \end{bmatrix}}$

$$\begin{bmatrix} DISH & ; & (2,3) \\ () & ; & () \end{bmatrix}\begin{bmatrix} () & ; & () \\ PAS & ; & (0,1) \end{bmatrix}$$

6. $$\frac{G_N\begin{pmatrix} N & ; & [1] & , & DISH & , & () & , & PAS \\ N & ; & [1] & , & () & , & PAS \end{pmatrix}}{\begin{bmatrix} N & ; & (2,3) \\ N & ; & (0,1) \end{bmatrix}}$$

$$\begin{bmatrix} D & ; & (1,2) \\ () & ; & () \end{bmatrix}\begin{bmatrix} N & ; & (2,3) \\ N & ; & (0,1) \end{bmatrix}$$

7. $$\frac{G_N\begin{pmatrix} NP & ; & [1,2] & , & D & , & N \\ NP & ; & [1] & , & () & , & N \end{pmatrix}}{\begin{bmatrix} NP & ; & (1,3) \\ NP & ; & (0,1) \end{bmatrix}}$$

$$\begin{bmatrix} WASH & ; & (0,1) \\ () & ; & () \end{bmatrix}\begin{bmatrix} () & ; & () \\ MIT & ; & (1,2) \end{bmatrix}$$

8. $$\frac{G_N\begin{pmatrix} V & ; & [1] & , & WASH & , & () & , & MIT \\ V & ; & [1] & , & () & , & MIT \end{pmatrix}}{\begin{bmatrix} V & ; & (0,1) \\ V & ; & (1,2) \end{bmatrix}}$$

$$\begin{bmatrix} V & ; & (0,1) \\ V & ; & (1,2) \end{bmatrix}\begin{bmatrix} NP & ; & (1,3) \\ NP & ; & (0,1) \end{bmatrix}$$

9. $$\frac{G_N\begin{pmatrix} \$ & ; & [1,2] & , & V & , & NP \\ \$ & ; & [2,1] & , & V & , & NP \end{pmatrix}}{\begin{bmatrix} \$ & ; & (0,3) \\ \$ & ; & (0,2) \end{bmatrix}}$$

Figure 4: Possible sequence of inferences of Parser C on input *Wash the dishes / Pasudu moy*.

When some of the component texts are missing, we can ask the parser to infer a D-dimensional multitree that includes the missing components. The resulting multitree will cover the I **input components/dimensions** among its D dimensions. It will also express the $D - I$ **output components/dimensions**, along with their syntactic structures.

Item Form: $[X_D^1; \sigma_I^1]$ **Goal:** $[\$_D^1; (0, n_d)_I^1]$

Inference Rules

Scan component d $1 \le d \le I$:

$$\frac{G_T\begin{pmatrix} ()_{d-1}^1 & ; & ()_{d-1}^1 \\ X & ; & w_{d,i} \\ ()_I^{d+1} & ; & ()_I^{d+1} \end{pmatrix}}{\begin{bmatrix} ()_{d-1}^1 & ; & ()_{d-1}^1 \\ X & ; & (i-1,i) \\ ()_I^{d+1} & ; & ()_I^{d+1} \\ ()_D^{I+1} & & \end{bmatrix}}$$

Load component d, $I < d \le D$:

$$\frac{G_T\begin{pmatrix} ()_{d-1}^{I+1} & ; & ()_{d-1}^{I+1} \\ X & ; & t \\ ()_D^{d+1} & ; & ()_D^{d+1} \end{pmatrix}}{\begin{bmatrix} ()_I^1 & ; & ()_I^1 \\ ()_{d-1}^{I+1} & ; & \\ X & ; & \\ ()_D^{d+1} & & \end{bmatrix}}$$

Compose:

$$\frac{\left[Y_D^1;\tau_I^1\right]\left[Z_D^1;\sigma_I^1\right] G_N\begin{pmatrix} X_D^1; & \tau_I^1 \otimes \sigma_I^1 \\ & \rho_D^{I+1} \end{pmatrix}, Y_D^1, Z_D^1}{[X_D^1; \tau_I^1 + \sigma_I^1]}$$

Figure 5: Logic CT ("T" for Translation)

Figure 5 shows Logic CT, which is a generalization of Logic C. Translator CT is any parser based on Logic CT. The items of Translator CT have a D-dimensional label vector, as usual. However, their d-span vectors are only I-dimensional, because it is not necessary to constrain absolute word positions in the output dimensions. Instead, we need only constrain the cardinality of the output nonterminals, which is accomplished by the role templates ρ_D^{I+1} in the G_N term. Translator CT scans only the input components. Terminal productions with active output components are simply loaded from the grammar, and their LHSs are added to the chart without d-span information. Composition proceeds as before, except that there are no constraints on the role templates in the output dimensions – the role templates in ρ_D^{I+1} are free variables.

In summary, Logic CT differs from Logic C as follows:

- Items store no position information (d-spans) for the output components.
- For the output components, the *Scan* inferences are replaced by *Load* inferences, which are not constrained by the input.
- The *Compose* inference does not constrain the d-spans of the output components. (Though it still constrains their cardinality.)

We have constructed a translator from a synchronous parser merely by relaxing some constraints on the output dimensions. Logic C is just Logic CT for the special case where $I = D$. The relationship between the two classes of algorithms is easier to see from their declarative logics than it would be from their procedural pseudocode or equations.

Like Parser C, Translator CT can *Compose* items that have no dimensions in common. If one of the items is active only in the input dimension(s), and the other only in the output dimension(s), then the inference is, de facto, a translation. The possible translations are determined by consulting the grammar. Thus, in addition to its usual function of evaluating syntactic structures, the grammar simultaneously functions as a translation model.

Logic CT can be coupled with any parsing semiring. For example, under a boolean semiring, this logic will succeed on an I-dimensional input if and only if it can infer a D-dimensional multitree whose root is the goal item. Such a tree would contain a $(D - I)$-dimensional translation of the input. Thus, under a boolean semiring, Translator CT can determine whether a translation of the input exists.

Under an inside-probability semiring, Translator CT can compute the total probability of all multitrees containing the input and its translations in the $D-I$ output components. All these derivation trees, along with their probabilities, can be efficiently represented as a packed parse forest, rooted at the goal item. Unfortunately, finding the most probable output string still requires summing probabilities over an exponential number of trees. This problem was shown to be NP-hard in the one-dimensional case (Sima'an, 1996). We have no reason to believe that it is any easier when $D > 1$.

The Viterbi-derivation semiring would be the most often used with Translator CT in practice. Given a D-PMTG, Translator CT can use this semiring to find the single most probable D-dimensional multitree that covers the I-dimensional input. The multitree inferred by the translator will have the words of both the input and the output components in its leaves. For example, given a suitable grammar and the input *Pasudu moy*, Translator CT could infer the multitree in Figure 2. The set of inferences would be exactly the same as those listed in Figure 4, except that the items would have no d-spans in the English component.

In practice, we usually want the output as a string tuple, rather than as a multitree. Under the various derivation semirings (Goodman, 1999), Translator CT can store the output role templates ρ_D^{I+1} in each internal node of the tree. The intended ordering of the terminals in each output dimension can be assembled from these templates by a linear-time **linearization** post-process that traverses the finished multitree in postorder.

To the best of our knowledge, Logic CT is the first published translation logic to be compatible with all of the semirings catalogued by Goodman (1999), among others. It is also the first to simultaneously accommodate multiple input components and multiple output components. When a source document is available in multiple languages, a translator can benefit from the disambiguating information in each. Translator CT can take advantage of such information without making the strong independence assumptions of Och & Ney (2001). When output is desired in multiple languages, Translator CT offers all the putative benefits of the interlingual approach to MT, including greater efficiency and greater consistency across output components. Indeed, the language of multitrees can be viewed as an interlingua.

5 Synchronization

We have explored inference of I-dimensional multitrees under a D-dimensional grammar, where $D \geq I$. Now we generalize along the other axis of Figure 1(a). Multitext synchronization is most often used to infer I-dimensional multitrees without the benefit of an I-dimensional grammar. One application is inducing a parser in one language from a parser in another (Lü *et al.*, 2002). The application that is most relevant to this paper is bootstrapping an I-dimensional grammar. In theory, it is possible to induce a PMTG from multitext in an unsupervised manner. A more reliable way is to start from a corpus of multitrees — a **multitreebank**.[3]

We are not aware of any multitreebanks at this time. The most straightforward way to create one is to parse some multitext using a synchronous parser, such as Parser C. However, if the goal is to bootstrap an I-PMTG, then there is no I-PMTG that can evaluate the G terms in the parser's logic. Our solution is to orchestrate lower-dimensional knowledge sources to evaluate the G terms. Then, we can use the same parsing logic to synchronize multitext into a multitreebank.

To illustrate, we describe a relatively simple synchronizer, using the Viterbi-derivation semiring.[4] Under this semiring, a synchronizer computes the single most probable multitree for a given multitext.

[3] In contrast, a parallel treebank might contain no information about translational equivalence.

[4] The inside-probability semiring would be required for maximum-likelihood synchronization.

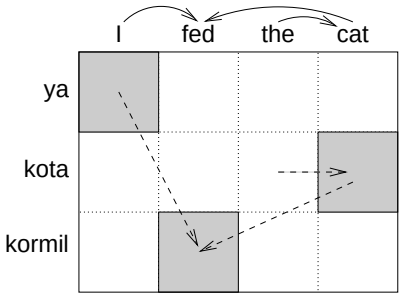


Figure 6: Synchronization. Only one synchronous dependency structure (dashed arrows) is compatible with the monolingual structure (solid arrows) and word alignment (shaded cells).

If we have no suitable PMTG, then we can use other criteria to search for trees that have high probability. We shall consider the common synchronization scenario where a lexicalized monolingual grammar is available for at least one component.[5] Also, given a tokenized set of I-tuples of parallel sentences, it is always possible to estimate a word-to-word translation model $\Pr(u_1^D|u_{D+1}^I)$ (e.g., Och & Ney, 2003).[6]

A word-to-word translation model and a lexicalized monolingual grammar are sufficient to drive a synchronizer. For example, in Figure 6 a monolingual grammar has allowed only one dependency structure on the English side, and a word-to-word translation model has allowed only one word alignment. The syntactic structures of all dimensions of a multitree are isomorphic up to reordering of sibling nodes and deletion. So, given a fixed correspondence between the tree leaves (i.e. words) across components, choosing the optimal structure for one component is tantamount to choosing the optimal synchronous structure for all components.[7] Ignoring the nonterminal labels, only one dependency structure is compatible with these constraints – the one indicated by dashed arrows. Bootstrapping a PMTG from a lower-dimensional PMTG and a word-to-word translation model is similar in spirit to the way that regular grammars can help to estimate CFGs (Lari & Young, 1990), and the way that simple translation models can help to bootstrap more sophisticated ones (Brown *et al.*, 1993).

[5] Such a grammar can be induced from a treebank, for example. We are currently aware of treebanks for English, Spanish, German, Chinese, Czech, Arabic, and Korean.

[6] Although most of the literature discusses word translation models between only two languages, it is possible to combine several 2D models into a higher-dimensional model (Mann & Yarowsky, 2001).

[7] Except where the unstructured components have words that are linked to nothing.

We need only redefine the G terms in a way that does not rely on an I-PMTG. Without loss of generality, we shall assume a D-PMTG that ranges over the first D components, where $D < I$. We shall then refer to the D structured components and the $I - D$ unstructured components.

We begin with G_T. For the structured components $d, 1 \leq d \leq D$, we retain the grammar-based definition: $G_T(X_d[h_d]; h_d) = \Pr(h_d|X_d)$,[8] where the latter probability can be looked up in our D-PMTG. For the unstructured components, there are no useful nonterminal labels. Therefore, we assume that the unstructured components use only one (dummy) nonterminal label λ, so that $G_T(X_d[h_d]; h_d) = 1$ if $X = \lambda$ and undefined otherwise for $D < d \leq I$.

Our treatment of nonterminal productions begins by applying the chain rule[9]

$$\begin{aligned} & G_N(X_I^1[h_I^1]; \rho_I^1, Y_I^1[g_I^1], Z_I^1[h_I^1]) \\ & \quad = \Pr(\rho_I^1, g_I^1, Y_I^1, Z_I^1|X_I^1, h_I^1) \qquad (3) \\ & \quad = \Pr(\rho_D^1, g_D^1, Y_D^1, Z_D^1|X_I^1, h_I^1) \\ & \quad \times \Pr(Y_I^{D+1}, Z_I^{D+1}|\rho_D^1, g_D^1, Y_D^1, Z_D^1, X_I^1, h_I^1) \\ & \quad \times \Pr(g_I^{D+1}|\rho_D^1, g_D^1, Y_I^1, Z_I^1, X_I^1, h_I^1) \\ & \quad \times \Pr(\rho_I^{D+1}|\rho_D^1, g_I^1, Y_I^1, Z_I^1, X_I^1, h_I^1) \qquad (4) \end{aligned}$$

and continues by making independence assumptions. The first assumption is that the structured components of the production's RHS are conditionally independent of the unstructured components of its LHS:

$$\begin{aligned} & \Pr(\rho_D^1, g_D^1, Y_D^1, Z_D^1|X_I^1, h_I^1) = \\ & \qquad = \Pr(\rho_D^1, g_D^1, Y_D^1, Z_D^1|X_D^1, h_D^1) \qquad (5) \end{aligned}$$

The above probability can be looked up in the D-PMTG. Second, since we have no useful nonterminals in the unstructured components, we let

$$\Pr(Y_I^{D+1}, Z_I^{D+1}|\rho_D^1, g_D^1, Y_D^1, Z_D^1, X_I^1, h_I^1) = 1 \qquad (6)$$

if $Y_I^{D+1} = Z_I^{D+1} = \lambda_I^{D+1}$ and 0 otherwise. Third, we assume that the word-to-word translation probabilities are independent of anything else:

$$\begin{aligned} & \Pr(g_I^{D+1}|\rho_D^1, g_D^1, Y_I^1, Z_I^1, X_I^1, h_I^1) = \\ & \qquad = \Pr(g_I^{D+1}|g_D^1) \qquad (7) \end{aligned}$$

[8] We have ignored lexical heads so far, but we need them for this synchronizer.

[9] The procedure is analogous when the heir is the first nonterminal link on the RHS, rather than the second.

These probabilities can be obtained from our word-to-word translation model, which would typically be estimated under exactly such an independence assumption. Finally, we assume that the output role templates are independent of each other and uniformly distributed, up to some maximum cardinality m. Let $v(m)$ be the number of unique role templates of cardinality m or less. Then

$$\Pr(\rho_I^{D+1}, |\rho_D^1, g_I^1, Y_I^1, Z_I^1, X_I^1, h_I^1) = \quad (8)$$
$$= \Pr(\rho_I^{D+1}) = \prod_{d=D+1}^{I} \frac{1}{v(m)} = \frac{1}{v(m)^{I-D}}$$

Under Assumptions 5–8,

$$G_N(X_I^1[h_I^1]; \rho_I^1, Y_I^1[g_I^1], Z_I^1[h_I^1]) = \quad (9)$$
$$= \frac{\Pr(\rho_D^1, g_D^1, Y_D^1, Z_D^1 | X_D^1, h_D^1) \cdot \Pr(g_I^{D+1} | g_D^1)}{v(m)^{I-D}}$$

if $Y_I^{D+1} = Z_I^{D+1} = \lambda_I^{D+1}$ and 0 otherwise. We can use these definitions of the grammar terms in the inference rules of Logic C to synchronize multitexts into multitreebanks.

More sophisticated synchronization methods are certainly possible. For example, we could project a part-of-speech tagger (Yarowsky & Ngai, 2001) to improve our estimates in Equation 6. Yet, despite their relative simplicity, the above methods for estimating production rule probabilities use all of the available information in a consistent manner, without double-counting. This kind of synchronizer stands in contrast to more ad-hoc approaches (e.g., Matsumoto, 1993; Meyers, 1996; Wu, 1998; Hwa *et al.*, 2002). Some of these previous works fix the word alignments first, and then infer compatible parse structures. Others do the opposite. Information about syntactic structure can be inferred more accurately given information about translational equivalence, and vice versa. Commitment to either kind of information without consideration of the other increases the potential for compounded errors.

6 Multitree-based Statistical MT

Multitree-based statistical machine translation (MTSMT) is an architecture for SMT that revolves around multitrees. Figure 7 shows how to build and use a rudimentary MTSMT system, starting from some multitext and one or more monolingual treebanks. The recipe follows:

T1. Induce a word-to-word translation model.

T2. Induce PCFGs from the relative frequencies of productions in the monolingual treebanks.

T3. Synchronize some multitext, e.g. using the approximations in Section 5.

T4. Induce an initial PMTG from the relative frequencies of productions in the multitreebank.

T5. Re-estimate the PMTG parameters, using a synchronous parser with the expectation semiring.

A1. Use the PMTG to infer the most probable multitree covering new input text.

A2. Linearize the output dimensions of the multitree.

Steps T2, T4 and A2 are trivial. Steps T1, T3, T5, and A1 are instances of the generalized parsers described in this paper.

Figure 7 is only an architecture. Computational complexity and generalization error stand in the way of its practical implementation. Nevertheless, it is satisfying to note that all the non-trivial algorithms in Figure 7 are special cases of Translator CT. It is therefore possible to implement an MTSMT system using just one inference algorithm, parameterized by a grammar, a semiring, and a search strategy. An advantage of building an MT system in this manner is that improvements invented for ordinary parsing algorithms can often be applied to all the main components of the system. For example, Melamed (2003) showed how to reduce the computational complexity of a synchronous parser by $O(n^D)$, just by changing the logic. The same optimization can be applied to the inference algorithms in this paper. With proper software design, such optimizations need never be implemented more than once. For simplicity, the algorithms in this paper are based on CKY logic. However, the architecture in Figure 7 can also be implemented using generalizations of more sophisticated parsing logics, such as those inherent in Earley or Head-Driven parsers.

7 Conclusion

This paper has presented generalizations of ordinary parsing that emerge when the grammar and/or the input can be multidimensional. Along the way, it has elucidated the relationships between ordinary parsers and other classes of algorithms, some previously known and some not. It turns out that, given some multitext and a monolingual treebank, a rudimentary multitree-based statistical machine translation system can be built and applied using only generalized parsers and some trivial glue.

There are three research benefits of using generalized parsers to build MT systems. First, we can

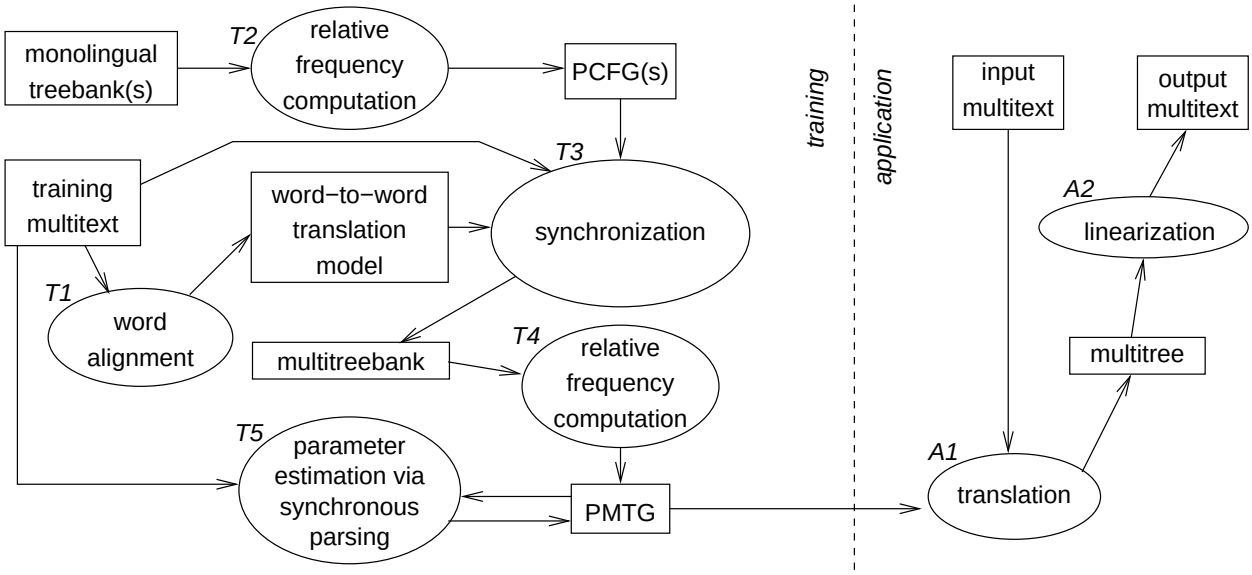


Figure 7: Data-flow diagram for a rudimentary MTSMT system based on generalizations of parsing.

take advantage of past and future research on making parsers more accurate and more efficient. Therefore, second, we can concentrate our efforts on better models, without worrying about MT-specific search algorithms. Third, more generally and most importantly, this approach encourages MT research to be less specialized and more transparently related to the rest of computational linguistics.

Acknowledgments

Thanks to Joseph Turian, Wei Wang, Ben Wellington, and the anonymous reviewers for valuable feedback. This research was supported by an NSF CAREER Award, the DARPA TIDES program, and an equipment gift from Sun Microsystems.

References

A. Aho & J. Ullman (1969) "Syntax Directed Translations and the Pushdown Assembler," *Journal of Computer and System Sciences 3*, 37-56.

H. Alshawi, S. Bangalore, & S. Douglas (2000) "Learning Dependency Translation Models as Collections of Finite State Head Transducers," *Computational Linguistics 26(1)*:45-60.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, & R. L. Mercer (1993) "The Mathematics of Statistical Machine Translation: Parameter Estimation," *Computational Linguistics 19*(2):263–312.

J. Goodman (1999) "Semiring Parsing," *Computational Linguistics 25(4)*:573–305.

R. Hwa, P. Resnik, A. Weinberg, & O. Kolak (2002) "Evaluating Translational Correspondence using Annotation Projection," *Proceedings of the ACL*.

J. Eisner (2002) "Parameter Estimation for Probabilistic Finite-State Transducers," *Proceedings of the ACL*.

K. Lari & S. Young (1990) "The Estimation of Stochastic Context-Free Grammars using the Inside-Outside Algorithm," *Computer Speech and Language Processing 4*:35–56.

Y. Lü, S. Li, T. Zhao, & M. Yang (2002) "Learning Chinese Bracketing Knowledge Based on a Bilingual Language Model," *Proceedings of COLING*.

G. S. Mann & D. Yarowsky (2001) "Multipath Translation Lexicon Induction via Bridge Languages," *Proceedings of HLT/NAACL*.

Y. Matsumoto (1993) "Structural Matching of Parallel Texts," *Proceedings of the ACL*.

I. D. Melamed (2003) "Multitext Grammars and Synchronous Parsers," *Proceedings of HLT/NAACL*.

I. D. Melamed, G. Satta, & B. Wellington (2004) "Generalized Multitext Grammars," *Proceedings of the ACL* (this volume).

A. Meyers, R. Yangarber, & R. Grishman (1996) "Alignment of Shared Forests for Bilingual Corpora," *Proceedings of COLING*.

F. Och & H. Ney (2001) "Statistical Multi-Source Translation," *Proceedings of MT Summit VIII*.

F. Och & H. Ney (2003) "A Systematic Comparison of Various Statistical Alignment Models," *Computational Linguistics 29(1)*:19-51.

K. Sima'an (1996) "Computational Complexity of Probabilistic Disambiguation by means of Tree-Grammars," *Proceedings of COLING*.

D. Wu (1996) "A polynomial-time algorithm for statistical machine translation," *Proceedings of the ACL*.

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

D. Wu & H. Wong (1998) "Machine translation with a stochastic grammatical channel," *Proceedings of the ACL*.

K. Yamada & K. Knight (2002) "A Decoder for Syntax-based Statistical MT," *Proceedings of the ACL*.

D. Yarowsky & G. Ngai (2001) "Inducing Multilingual POS Taggers and NP Bracketers via Robust Projection Across Aligned Corpora," *Proceedings of the NAACL*.