16

Alignment

Dekai Wu
The Hong Kong University of Science and Technology

16.1 Introduction

In this chapter, we discuss the work done on automatic alignment of parallel texts for various purposes. Fundamentally, an alignment algorithm accepts as input a *bitext* and produces as output a *bisegmentation* relation that identifies corresponding segments between the texts. A bitext consists of two texts that are translations of each other.* Bitext alignment fundamentally lies at the heart of all data-driven machine translation methods, and the rapid research progress on alignment since 1990 reflects the advent of statistical machine translation (SMT) and example-based machine translation (EBMT) approaches. Yet the importance of alignment extends as well to many other practical applications for translators, bilingual lexicographers, and even ordinary readers.

* In a "Terminological note" prefacing his book, Veronis (2000) cites Alan Melby pointing out that the alternative term *parallel text* creates an unfortunate and confusing clash with the translation theory and terminological community, who use the same term instead to mean what NLP and computational linguistics researchers typically refer to as *non-parallel corpora* or *comparable corpora*—texts in different languages from the same domain, but not necessarily translations of each other.

Automatically learned resources for MT, NLP, and humans.
Bitext alignment methods are the core of many methods for machine learning of language resources to be used by SMT or other NLP applications, as well as human translators and linguists. The side effects of alignment are often of more interest than the aligned text itself. Alignment algorithms offer the possibility of extracting various sorts of knowledge resources, such as (a) phrasal bilexicons listing word or collocation translations; (b) translation examples at the sentence, constituent, and/or phrase level; or (c) tree-structured translation patterns such as transfer rules, translation frames, or treelets. Such resources constitute a database that may be used by SMT and EBMT systems (Nagao 1984), or they may be taken as training data for further machine learning to mine deeper patterns. Alignment has also been employed to infer sentence bracketing or constituent structure as a side effect.

Biconcordances.
Historically, the first application for bitext alignment algorithms was to automate the production of the cross-indexing for bilingual concordances (Warwick and Russell 1990; Karlgren et al. 1994; Church and Hovy 1993). Such concordances are consulted by human translators to find the previous contexts in which a term, idiom, or phrase was translated, thereby helping the translator to maintain consistency with preexisting translations, which is important in government and legal documents. An additional benefit of biconcordances is a large increase in navigation ease in bitexts.

Bitext for readers.
Aligned bitexts, in addition to their use in translators' concordances, are also useful for bilingual readers and language learners.

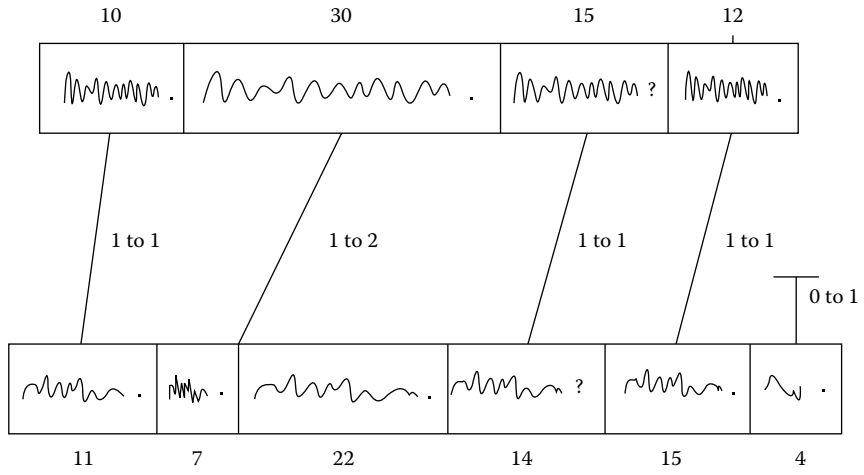
Linked biconcordances and bilexicons.
An aligned bitext can be automatically linked with a bilexicon, providing a more effective interface for lexicographers, corpus annotators, as well as human translators. This was implemented in the BICORD system (Klavans and Tzoukermann 1990).

Translation validation.
A word-level bitext alignment system can be used within a translation checking tool that attempts to automatically flag possible errors, in the same way that spelling and style checkers operate (Macklovitch 1994). The alignment system in this case is primed to search for deceptive cognates (*faux amis*) such as *library*/*librarie* in English and French.

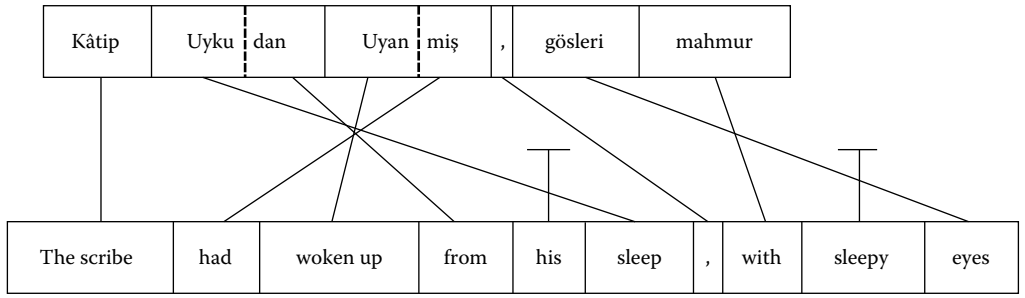
A wide variety of techniques now exist, ranging from the most simple (counting characters or words) to the more sophisticated, sometimes involving linguistic data (lexicons) that may or may not have been automatically induced themselves. Some techniques work on precisely translated parallel corpora, while others work on noisy, comparable, or nonparallel corpora. Some techniques make use of apparent morphological features, while others rely on cognates and loan-words; of particular interest is work done on languages that do not have a common writing system. Some techniques align only shallow, flat chunks, while others align compositional, hierarchical structures. The robustness and generality of different techniques have generated much discussion.

Techniques have been developed for aligning segments at various granularities: documents, paragraphs, sentences, constituents, collocations or phrases, words, and characters, as seen in the examples in Figure 16.1. In the following sections, we first discuss the general concepts underlying alignment techniques. Each of the major categories of alignment techniques are considered in turn in subsequent sections: document-structure alignment, sentence alignment, alignment for noisy bitexts, word alignment, constituent and tree alignment, and biparsing alignment.

We attempt to use a consistent notation and conceptual orientation throughout. Particularly when discussing algorithms, we stick to formal data constructs such as "token," "sequence," and "segment," rather than linguistic entities. Many algorithms for alignment are actually quite general and can often be applied at many levels of granularity. Unfortunately, this is often obscured by the use of linguistically



(a) Sentence/paragraph alignment



(b) Turkish–English character/word/phrase alignment

FIGURE 16.1 Alignment examples at various granularities.

loaded terms such as "word," "phrase," or "sentence." For these reasons, some techniques we discuss may appear superficially quite unlike the original works from which our descriptions were derived.

16.2 Definitions and Concepts

16.2.1 Alignment

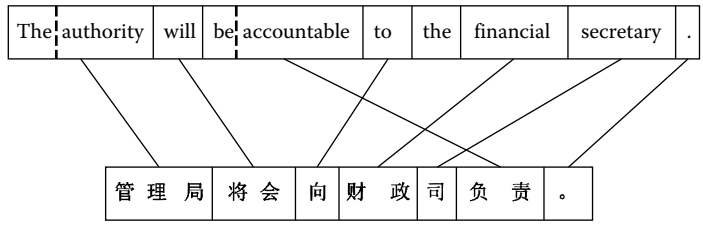
The general problem of aligning a parallel text is, more precisely, to find its optimal parallel segmentation or *bisegmentation* under some set of constraints:

Input.

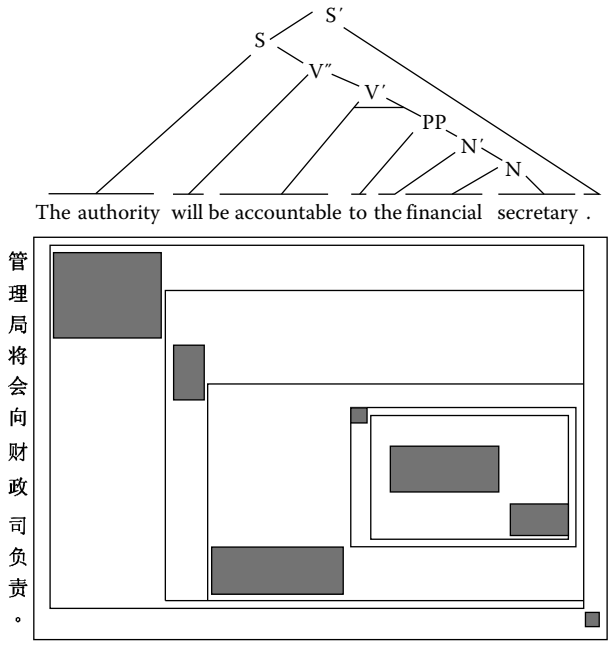
A bitext $(\mathbf{e}, \mathbf{f})$. Assume that the vector $\mathbf{e}$ contains a sequence of T tokens $\mathbf{e}_0, \ldots, \mathbf{e}_{T-1}$ and $\mathcal{E}$ is the set $\{0, 1, 2, \ldots, T-1\}$. Similarly, the vector $\mathbf{f}$ contains a sequence of V tokens $\mathbf{f}_0, \ldots, \mathbf{f}_{V-1}$ and $\mathcal{F}$ is the set $\{0, 1, 2, \ldots, V-1\}$.* When the direction of translation is relevant, $\mathbf{f}$ is the input language (*foreign*) string, and $\mathbf{e}$ is the output language (*emitted*) string.†

* We use the following notational conventions: bold letters are vectors, calligraphic letters are sets, and capital (non-bold) letters are constants.

† The **e** and **f** convention dates back to early statistical MT work where the input foreign language was French and the output language emitted was English.



(c) English–Chinese character/word/phase alignment



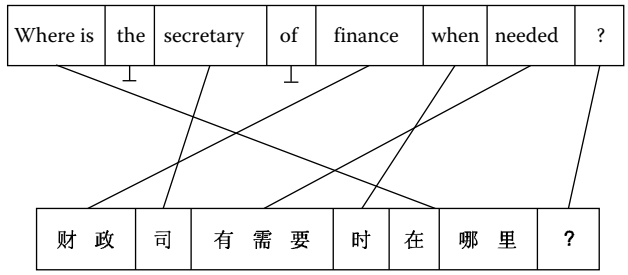
(d) English–Chinese character/word/phase tree alignment

FIGURE 16.1 (continued)

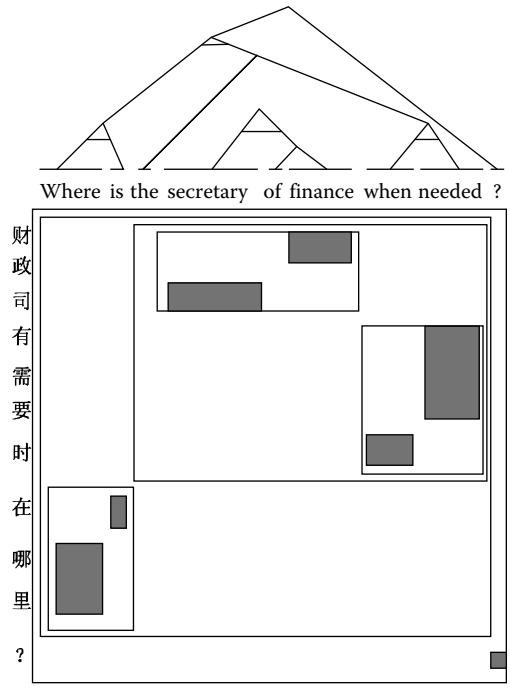
Output.

A *bisegmentation* of the bitext, designated by a set $\mathcal{A}$ of *bisegments*, where each bisegment (p, r) *couples* an emitted segment p to a foreign segment r and often can instead more conveniently be uniquely identified by its *span pair* (s, t, u, v). Any emitted segment p has a span $\mathbf{span}(p) = (s, t)$ that bounds the passage $\mathbf{e}_{s..t}$ formed by the subsequence of tokens $\mathbf{e}_s, \ldots, \mathbf{e}_{t-1}$ where $0 \leq s \leq t \leq T$. Similarly, any foreign segment r has a span $\mathbf{span}(r) = (u, v)$, which bounds the passage $\mathbf{f}_{u..v}$ formed by the subsequence of tokens $\mathbf{f}_u, \ldots, \mathbf{f}_{v-1}$ where $0 \leq u \leq v \leq V$. Conversely, we write $p = \mathbf{seg}(s, t)$ and $r = \mathbf{seg}(u, v)$ when the span uniquely identifies a segment. Otherwise, for models that permit more than one segment to label the same span, we instead write $\mathbf{segs}(s, t)$ or $\mathbf{segs}(u, v)$ to denote the set of segments that label the span.

Note that a bisegmentation inherently defines two monolingual segmentations, on both of the monolingual texts **e** and **f**.



(e) English–Chinese character/word/phase alignment



(f) English–Chinese character/word/phase tree alignment

FIGURE 16.1 (continued)

16.2.1.1 Monotonic Alignment

The term "alignment" has become something of a misnomer in computational linguistics. Technically, in an alignment the coupled passages must occur in the same order in both texts, that is, with no crossings. Many alignment techniques have their roots in speech recognition applications, where acoustic waveforms need to be aligned to transcriptions that are of course in the same order. In bitext research, the term "alignment" originally described the reasonable approximating assumption that paragraph and sentence translations always preserve the original order.

However, "word alignment" was subsequently co-opted to mean coupling of words within sentences, even when word-coupling models do not assume that order is monotonically preserved across translation. Since this permits permutations where the word segments are reordered in translation, properly speaking, such a non-monotonic "alignment" is rather a bisegmentation, which can be seen as a (partial) binary relation from the segments $\mathbf{e}_{s..t}$ to the segments $\mathbf{f}_{u..v}$. To avoid confusion, we will use the term *monotonic alignment* or *monotone alignment* whenever we mean "alignment" in its proper sense.

16.2.1.2 Disjoint (Flat) Alignment

As shown in Figure 16.1a through c and e, a common type of alignment is the simple, flat case of *disjoint alignment* where all segments are restricted to be *disjoint* in both languages, meaning that no two segments $\mathbf{e}_{s..t}$ and $\mathbf{e}_{s'..t'}$ participating in the alignment overlap and, likewise, no two segments $\mathbf{e}_{s..t}$ and $\mathbf{e}_{s'..t'}$ participating in the alignment overlap. That is, for any two bisegments (s, t, u, v) and (s', t', u', v'), either $s \leq t'$ or $t \leq s'$, and either $u \leq v'$ or $v \leq u'$.

In the simplest case of disjoint alignment where every segment is exactly one token long, i.e., where $t - s = 1$ and $v - u = 1$, then $\mathcal{A}$ can represent a non-total many-to-many map from $\mathcal{E}$ to $\mathcal{F}$.

16.2.1.3 Compositional (Hierarchical) Tree Alignment

Another common type of alignment is the more general, hierarchical case of *compositional alignment* or *tree alignment* where smaller bisegments may be nested within larger bisegments, as shown in Figure 16.1d and f. The simplest case is to align sentences within paragraphs that are themselves aligned. A more complex case is to couple nested constituents in sentences.

More precisely, two segments $\mathbf{e}_{s..t}$ and $\mathbf{e}_{s'..t'}$ participating in the alignment may be either disjoint or nested, i.e., for any two bisegments $q = (s, t, u, v)$ and $q' = (s', t', u', v')$, either q and q' are disjoint in $\mathcal{E}$ so that $s \leq t'$ or $t \leq s'$, or they are nested so that $s \leq s' \leq t' \leq t$ or $s' \leq s \leq t \leq t'$; and similarly for $\mathcal{F}$.

A compositional alignment forms a tree, where the external leaf nodes are a set of disjoint (bi)segments, and internal nodes are bisegments whose children are nested (bi)segments.

16.2.1.4 Subtokens and Subtokenization

Some models are designed to align tokens primarily at one granularity, yet the tokens can be further broken into even finer pieces for secondary purposes. Tokens, after all, are just segments at a specific level of disjoint flat segmentation that has been designated as primitive with respect to some algorithm. Subtokenization is especially common with compositional and hierarchical alignments, in applications such as paragraph/sentence alignment or tree-structured word alignment.

Intuitively, $\mathbf{e}'$ is a *subtokenization* of $\mathbf{e}$ if it refines the tokens in $\mathbf{e}$ into finer-grained *subtokens* (Guo 1997). More precisely, let the tokens in $\mathbf{e}$ and $\mathbf{e}'$ be both defined on some primitive alphabet Σ. Let $G(\mathbf{e})$ be the string generation operation that maps any token $\mathbf{e}_s$ into the string in Σ^* that the token represents, and assume both $\mathbf{e}$ and $\mathbf{e}'$ generate the same string, i.e., $G(\mathbf{e}_{0..T}) = G(\mathbf{e}'_{0..T'})$. Then $\mathbf{e}'$ is a subtokenization of $\mathbf{e}$ if every token $\mathbf{e}_s$ corresponds to the concatenation of one or more tokens $\mathbf{e}'_{s'..t'}$, i.e., $G(\mathbf{e}_s) = G(\mathbf{e}'_{s'..t'})$.

16.2.1.5 Bijective, Injective, Partial, and Many-to-Many Alignments

When we are speaking with respect to particular *monolingual* disjoint segmentations of (both sides of) a bitext, a few other concepts are often useful.

Where the pair of monolingual segmentations comes from depends on our assumptions. In some situations, we might assume monolingual segmentations that are fixed by monolingual preprocessors such as morphological analyzers or word/phrase segmenters.

In other situations, we may instead assume whatever monolingual segmentations result from aligning a bitext. In the trivial case, we can simply assume the two monolingual segmentations that are inherently defined by the alignment's output bisegmentation. A more useful case, under compositional tree alignment, is to assume the monolingual segmentations imposed by the leaf nodes of the tree.

Whatever assumptions we use to arrive at a pair of monolingual segmentations, then, in a *1-to-1 alignment* or *bijective alignment*, every segment in each text is coupled to exactly one segment in the other text. A bijective alignment is *total*, meaning that no segment remains uncoupled. In practice, bijective alignments are almost never achievable except at the chapter/section granularity, or perhaps at the paragraph granularity for extremely tight translations. Ordinarily, we aim for a *partial alignment*, in which some segments remain uncoupled *singletons*; and/or we aim for a *many-to-many alignment*, in which segments may be coupled to multiple segments. Another often-useful approximating assumption

is that the alignment is a (partial) function from a language-0 position to a language-1 position (but not necessarily vice versa). Such a relation is a *many-to-1* or *right-unique* alignment and is written $v = \mathbf{a}'(t)$. Similarly, a *1-to-many* or *left-unique* or *injective* relation is written $t = \mathbf{a}(v)$. A bijective 1-to-1 alignment is injective in both directions, i.e., both left-unique and right-unique. For convenience, we will sometimes freely switch between the set notation $\mathcal{A}$ and the function notation $\mathbf{a}(\cdot)$ to refer to the same injective alignment.

Unless we specify otherwise, "alignments" are non-monotonic, many-to-many, partial, and non-compositional.

16.2.2 Constraints and Correlations

Every alignment algorithm inputs a bitext and outputs a set of couplings. The techniques are nearly all statistical in nature, due to the need for robustness in the face of imperfect translations. Much of the variation between techniques lies in the other kinds of information—constraints and correlations—that play a role in alignment. Some alignment techniques require one or more of these as inputs, and bring them to bear on the alignment hypothesis space. Others derive or learn such kinds of information as by-products. In some cases the by-products are of sufficient quality as to be taken as outputs in their own right, as mentioned above. Important kinds of information include the following.

Bijectivity constraint

Bijectivity is the assumption that the coupling between passages is 1-to-1 (usually in the sense of partial bijective maps, which allow some passages to remain uncoupled). This assumption is inapplicable at coarser granularities than the sentence-level. However, it is sometimes useful for word-level alignment, despite being clearly inaccurate. For example, consider the case where the words within a sentence pair are being aligned (Melamed 1997). If only one word in the language-0 sentence remains uncoupled and similarly for the language-1 sentence, then the bijectivity assumption implies a preference for coupling those two words, by the process of elimination. Such benefits can easily outweigh errors caused by the inaccuracies of the assumption.

Monoticity constraint

This assumption reduces the problem of coupling bitext passages to a properly monotonic alignment problem: coupled passages occur in the same order in both sides of the bitext.

Segment constraints (disjoint and compositional)

A *segment constraint* prevents an alignment algorithm from outputting any bisegment that crosses the monolingual segment's boundaries. That is, if a segment $\mathbf{e}_{s..t}$ is taken as a constraint, then for any bisegmentation (s', t', u', v') that forms part of an alignment, either (s, t) and (s', t') are disjoint in $\mathbf{e}$ so that $s \leq t'$ or $t \leq s'$, or they are nested so that $s \leq s' \leq t' \leq t$ or $s' \leq s \leq t \leq t'$.

Two cases of monolingual segment constraints are very common. A *disjoint segmentation* lets algorithms for disjoint (flat) alignment focus on a coarser granularity than the raw token level; for example, for sentence alignment it is usual to first break the input texts into sentence segments. On the other hand, a *compositional segmentation* of one or both of the monolingual texts can be obtained by monolingually parsing the text(s), imposing the resulting nested segments as constraints.

Bisegment constraints (anchor and slack constraints)

An anchor is a known bisegment, i.e., a pair of segments (or boundaries, in the case of zero-length segments) that is *a priori* known to be coupled, and is a hard constraint. As shown in Figure 16.2, *anchor constraints* are bisegment constraints that can be thought of as confirmed positions within the matrix that represents the alignment candidate space. The special boundary cases of the bisegments $(0, 0, 0, 0)$ and (T, T, V, V) are the *origin* and *terminus*, respectively.

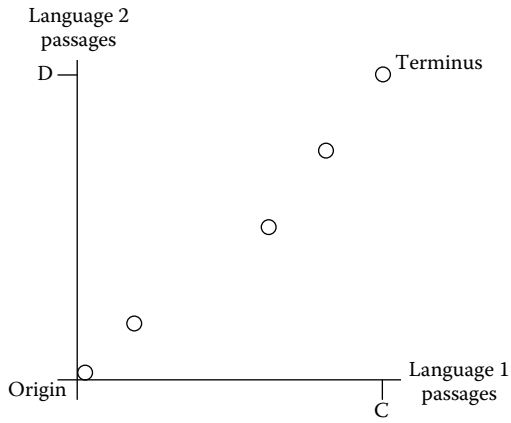


FIGURE 16.2 Anchors.

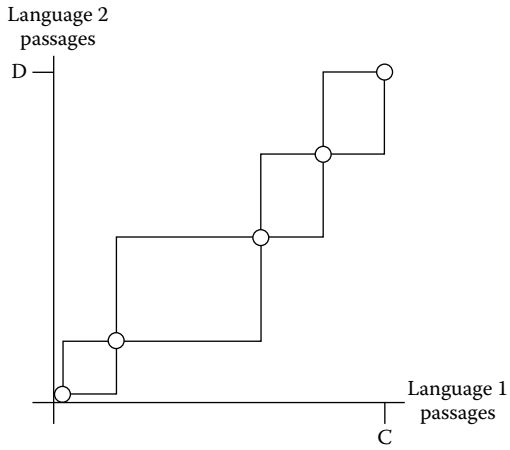


FIGURE 16.3 Slack bisegments between anchors.

When taken together with the monotonicity constraint, a set of anchors divides the problem into a set of smaller, independent alignment problems. Between any two anchors are passages whose alignment is still undetermined, but whose segment couplings must remain inside the region bounded by the anchors. As shown in Figure 16.3, the correct alignment could take any (monotonic) path through the rectangular region delimited by the anchors. We call the candidate subspace between adjacent anchors a *slack bisegment*. A set of slack bisegments can be used either as bisegment constraints (*slack constraints*), as for example described below, or to define features.

There are two common cases of anchor/slack bisegment constraints:

1. *End constraints*. Most techniques make the assumption that the origin and terminus are anchor boundaries. Some techniques also assume a coupling between the first and last *passages*.
2. *Incremental constraints*. A previous processing stage may produce an alignment of a larger passage size. If we are willing to commit to the coarser alignment, we obtain a set of anchor boundaries and/or slack bisegments.

The latter kind of anchor occurs in compositional alignment methods based on *iterative refinement* schemes, which progressively lay down anchors in a series of passes that gradually restrict

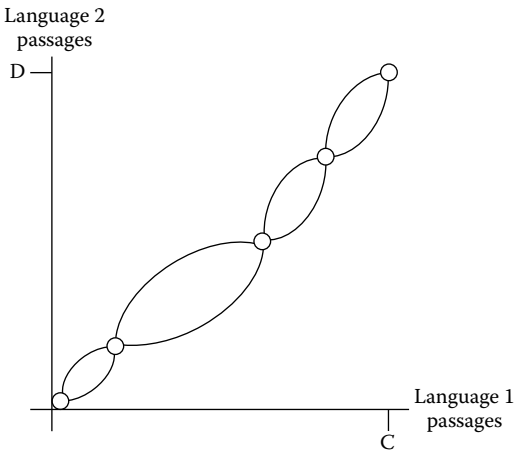


FIGURE 16.4 Banding the slack bisegments using variance.

the alignment candidate space. At the outset, candidate couplings may lie anywhere in the global (root) slack bisegment comprising the entire rectangular matrix; eventually, they are restricted to fall within many small local (leaf) slack bisegments. Each slack bisegment between adjacent anchors is a smaller alignment subproblem that can be processed independently. A variation on this scheme is *hierarchical iterative refinement*, which produces a compositional alignment by performing one pass at each level in a predefined hierarchy of token granularities. The first pass aligns segments at the coarsest token level (commonly, sections or paragraphs); this level is chosen so that the total numbers (T and V) of tokens (commonly, paragraphs or sentences) are small. Committing to the output of this stage yields a bisegmentation that is taken as the slack bisegment constraints for the next pass, which aligns segments at the next finer token level (commonly, paragraphs or sentences). This approach is taken so that the alignment subproblem corresponding to any slack bisegment has small T and V values. The alignment is refined on each pass until the sentence-level granularity is reached; at each pass, the quadratic cost is kept in check by the small number of tokens within each slack bisegment (between adjacent anchors).

Bands

A common heuristic constraint is to narrow the shape of the rectangular slack bisegments instead into *slack bands* that more closely resemble bands, as shown in Figure 16.4. We call this *banding*. Banding relies on the assumption that the correct couplings will not be displaced too far from the average, which is the diagonal between adjacent anchors. Different narrowing heuristics are possible.

One method is to model the variance assuming the displacement for each passage is independently and identically distributed. This means the standard deviation at the midpoint of a $T : V$ bitext is $O\sqrt{(T)}$ for the language-0 axis and $O\sqrt{(V)}$ for the language-1 axis. Kay and Röscheisen (1993) approximate this by using a banding function such that the maximum width of a band at its midpoint is $O\sqrt{(T)}$.*

Another method, due to Simard and Plamondon (1996), is to prune areas of the slack bisegment that are further from the rectangle's diagonal than some threshold, where the threshold is proportional to the distance between the anchors. This results in slack bands of the shape in Figure 16.5.

* They do not give the precise function.

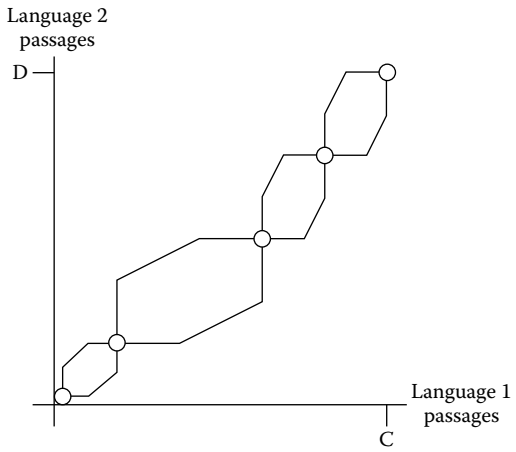


FIGURE 16.5 Banding the slack bisegments using width thresholds.

Banding has the danger of overlooking correct couplings if there are large differences in the translations. Kay and Röscheisen (1993) report a maximum displacement of 10 sentences from the diagonal, in a bitext of 255:300 sentences. However, most bitext, especially in larger collections, contains significantly more noise. The width of the band can be increased, but at significant computational expense.

Guides

A heuristic constraint we call *guiding*, due to Dagan et al. (1993), is applicable when a rough *guide* alignment already exists as the result of some earlier heuristic estimate. The preexisting alignment can be used as a guide to seek a more accurate alignment. Assuming the preexisting alignment is described by the mapping function $\mathbf{a}^0(v)$, a useful alignment range constraint is to define an allowable deviation from $\mathbf{a}^0(v)$ in terms of some distance d: a language-0 position t is a possible candidate to couple with a language-1 position v iff

$$\mathbf{a}^0(v) - d \leq t \leq \mathbf{a}^0(v) + d \tag{16.1}$$

This is depicted in Figure 16.6. We denote the set of $(s, s+1, t, t+1)$ couplings that meet all alignment range constraints as $\mathcal{B}$.

Alignment range constraints

Monotonicity, anchors, banding, and guiding are all special cases of alignment range constraints. There are many other possible ways to formulate restrictions on the space of allowable alignments. In general, alignment range constraints play an important role. Some alignment techniques are actually computationally infeasible without strong *a priori* alignment range constraints. Other techniques employ an iterative modification of alignment range constraints, similar to iterative refinement with anchors.

Bilexicon constraints

A bilexicon holds known translation pairs or *bilexemes*. In general, machine translation models work with phrasal bilexicons, that may contain characters, tokens, words, phrases, compounds, multi-word expressions, or collocations—particularly for non-alphabetic languages where the difference is not so clear.

The strongest lexeme-coupling constraints come from lexemes (or tokens, including punctuation) that have 1-to-1 deterministic translations. These in effect provide anchors if monotonicity is assumed. However, the rarity of such cases limits their usefulness.

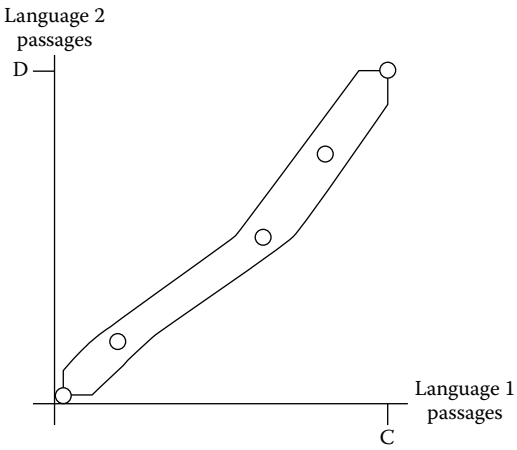


FIGURE 16.6 Guiding based on a previous rough alignment.

A bilexicon usually supplies lexical translations that are 1-to-many or many-to-many. The set of known translations can be used to cut down on the space of candidate couplings. Clearly, whatever pruning method is used must still allow for unknown word translations, since translation is not always word-by-word, and since bilexicons have Imperfect coverage (especially in methods that learn a bilexicon as they perform the alignment).

A *weighted bilexicon* stores additional information about the *degree* of correlation or association in word pairs. This can be particularly clean in probabilistic alignment techniques. Various other statistical or *ad hoc* scores can also be employed.

Cognates

Cognates are word pairs with common etymological roots, for example, the bilexemes *financed:financier* or *government:gouvernement* in English and French. For alphabetic languages that share the same (or directly mappable) alphabets, it is possible to construct heuristic functions that compare the spelling of two words or passages. In the simplest case, the function returns true or false (a decision function), acting like a low-accuracy, easily constructed bilexicon with low memory requirements. Alternatively, a function that returns a score acts like a weighted bilexicon. In either case, a cognate function can be used either in place of or in addition to an alignment bilexicon.

Segment lengths

The lengths of passages at or above the sentence granularity can be strong features for determining couplings. Most reported experiments indicate that the correlation is relatively strong even for unrelated languages, if the translation is tight. This means that the utility of this feature is probably more dependent on the genre than the language pair. Segment lengths are typically measured in bytes, characters, or simple word tokenizations. Length-based methods are discussed in Section 16.3.1.

Syntactic parses

Using syntactic information from automatic parsers to improve word alignment accuracy is increasingly common, for example, as discussed in Sections 16.5 and 16.6.

Language universals

Outside of the relatively superficial features just discussed, relatively little attempt has been made to bring to bear constraints from theories about language-universal grammatical properties. One exception is discussed in Section 16.6. Language-universal constraints apply to all (or a large class) of language pairs, without making language-specific assumptions, and apply at the sentence and word granularities.

16.2.3 Classes of Algorithms

Broadly speaking, alignment techniques search for an optimal bisegmentation using (1) dynamic programming, (2) greedy best-first, (3) discriminative, or (4) heuristic algorithms for constrained optimization.

Although authors vary greatly on notations and descriptions, the majority of alignment algorithms can in fact be formulated as search algorithms that attempt to minimize the total cost of $\mathcal{A}$, the entire set of couplings, subject to some set of constraints:

$$Cost(\mathcal{A}) = \sum_{(p,r)\in\mathcal{A}} Cost(p, r) \tag{16.2}$$

Search techniques can be heuristic or exhaustive, and can employ greedy, backtracking, beam, or exhaustive strategies.

16.3 Sentence Alignment

Sentence level techniques generally adopt a restriction to monotonic alignment, and are applicable to larger units as well, such as paragraphs and sections. Taking advantage of this, hierarchical iterative refinement usually works rather well, by first aligning paragraphs, yielding biparagraphs whose internal sentences can subsequently be aligned.

Broadly speaking, sentence alignment techniques rely on sentence lengths, on lexical constraints and correlations, and/or on cognates. Other features could no doubt be used, but these approaches appear to perform well enough.

16.3.1 Length-Based Sentence Alignment

The length-based approach examines the lengths of the sentences. It is the most easily implemented technique, and performs nearly as well as lexical techniques for tightly translated corpora such as government transcripts. The overall idea is to use dynamic programming to find a minimum cost (maximum probability) alignment, assuming a simple hidden generative model that emits sentences of varying lengths. Purely length-based techniques do not examine word identities at all, and regard the bitext as nothing more than a sequence of sentences whose lengths are the only observable feature; Figure 16.7 depicts how the alignment algorithm sees the example of Figure 16.1a. The length-based approach to sentence alignment was first introduced by Gale and Church (1991a) and Brown et al. (1991), who describe essentially similar techniques; a more thorough evaluation is found in Gale and Church (1993).

This is a first example of the most common successful modeling paradigm for all sorts of alignment tasks: search for an *optimal cost alignment* that explains the bitext, assuming it was emitted by some underlying *generative model* of transduction. The generative model is usually stochastic, so the costs being minimized

Lengths of sentences in the bitext

Language 1:	10	30	15	12		
Language 2:	12	14	15	12	11	2

FIGURE 16.7 Example sentence lengths in an input bitext.

are simply weights representing negative log probabilities. In most cases, the generative models are most clearly described as *transducers* (procedural automata) or equivalent *transduction grammars** (declarative rule sets), which can be viewed in three ways:

Generation. A transducer or transduction grammar generates a *transduction*, which is a set of string translation pairs or *bistrings*, just as an ordinary (monolingual) language grammar generates a language, which is a set of strings. In the bilingual case, a transduction grammar simultaneously generates two strings (**e**, **f**) at once, which are translation pairs. The set of all bistrings that can be generated defines a relation between the input and output languages.

Recognition. A transducer or transduction grammar accepts or *biparses* all bistrings of a transduction, just as a language grammar parses or accepts all strings of a language.

Transduction. A transducer or transduction grammar translates or *transduces* foreign input strings **f** to emitted output strings **e**.

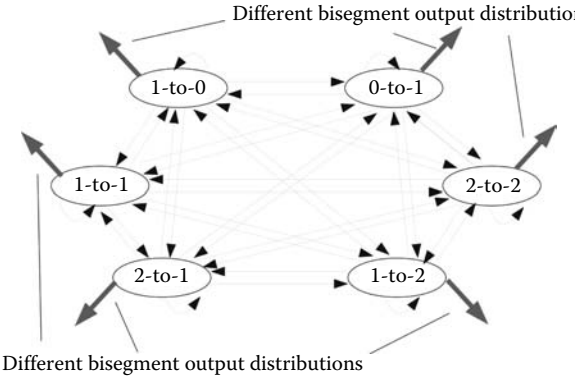
As we shall see, there is a hierarchy of equivalence classes for transductions—just as there is Chomsky's hierarchy of equivalence classes for languages. Just as in the monolingual case, there is a tradeoff between generative capacity and computational complexity: the more expressive classes of transductions are orders of magnitude more expensive to biparse and train:

MONOLINGUAL (Chomsky hierarchy)		BILINGUAL	
regular or finite-state languages		regular or finite-state transductions	
FSA	$O(n^2)$	**FST**	$O(n^4)$
or		*or*	
CFG		*SDTG (or synchronous CFG)*	
that is		*that is*	
right regular or left regular		*right regular or left regular*	
context-free languages		inversion transductions	
CFG	$O(n^3)$	**ITG**	$O(n^6)$
		or	
		SDTG (or synchronous CFG)	
		that is	
		binary or ternary or inverting	
		syntax-directed transductions	
		SDTG	$O(n^{2n+2})$
		or	
		(or synchronous CFG)	

Fortunately, for length-based sentence alignment, it suffices to model one of the simplest bilingual classes—*finite-state transductions*—which can be generated by a *finite-state transducer* or *FST* as depicted in Figure 16.8, or by an equivalent *finite-state transduction grammar* or *FSTG*, written as a transduction grammar that is right regular.† For sentence alignment, the natural tokens that are being aligned are sentences, but for length-based models, there is a also useful subtokenization at the byte, character, or simple word level. The model emits bitext as a monotonic series of bisegments, each bisegment being a short sequence of E emitted sentence tokens in language 0 coupled with a short sequence of F foreign

* Also recently called "synchronous grammars."

† FSTGs may also be written in left regular form.



(a) FST model using transition network notation.

$$
\begin{array}{llll}
 & \vdots & & \\
\text{1-to-1} & \rightarrow & [\ e_i/\epsilon\ \text{1-to-0}\ ] & \text{for all segments of subtoken length } i \text{ in language-0} \\
\text{1-to-1} & \rightarrow & [\ \epsilon/f_j\ \text{0-to-1}\ ] & \text{for all segments of subtoken length } j \text{ in language-1} \\
\text{1-to-1} & \rightarrow & [\ e_i/f_j\ \text{1-to-1}\ ] & \text{for all 1-to-1 bisegments of subtoken lengths } i,j \\
\text{1-to-1} & \rightarrow & [\ e_i/f_j\ \text{1-to-2}\ ] & \text{for all 1-to-2 bisegments of subtoken lengths } i,j \\
\text{1-to-1} & \rightarrow & [\ e_i/f_j\ \text{2-to-1}\ ] & \text{for all 2-to-1 bisegments of subtoken lengths } i,j \\
\text{1-to-1} & \rightarrow & [\ e_i/f_j\ \text{2-to-2}\ ] & \text{for all 2-to-2 bisegments of subtoken lengths } i,j \\
 & \vdots & &
\end{array}
$$

(b) Alternative notation for the same model using FSTG transduction rule notation; same pattern for 1-to-0, 0-to-1, 2-to-1, 1-to-2, and 2-to-2 rules.

FIGURE 16.8 Equivalent stochastic or weighted (a) FST and (b) FSTG notations for a finite-state bisegment generation process. Note that the node transition probability distributions are often tied to be the same for all node/nonterminal types.

sentence tokens in language 1.* For example, Figure 16.1a shows one sequence of 1-to-1, 1-to-2, and 0-to-1 bisegments that could have generated the bitext length sequences of Figure 16.7.

Formally, we denote a transducer or transduction grammar by $G = (\mathcal{N}, \mathcal{W}_0, \mathcal{W}_1, \mathcal{R}, S)$, where $\mathcal{N}$ is a finite set of nodes (states) or nonterminals, $\mathcal{W}_0$ is a finite set of words (terminals) of language 0, $\mathcal{W}_1$ is a finite set of words (terminals) of language 1, $\mathcal{R}$ is a finite set of transduction rules (which can be written as either transition rules or rewrite rules), and $S \in \mathcal{N}$ is the start node (state) or nonterminal. The space of bisegments (terminal-pairs) $\mathcal{X} = (\mathcal{W}_0 \cup \{\epsilon\}) \times (\mathcal{W}_1 \cup \{\epsilon\})$ contains lexical translations denoted x/y and singletons denoted x/ϵ or ϵ/y, where $x \in \mathcal{W}_0$ and $y \in \mathcal{W}_1$.

Each node (state) or nonterminal represents one type of bisegment. The usual practice is to allow bisegment types representing at least the following E-to-F configurations: 0-to-1, 1-to-0, 1-to-1, 1-to-2, and 2-to-1. Allowing 2-to-2 bisegments in addition is reasonable. It can be convenient to think of these bisegment types as operations for translating language-0 passages into language-1 passages, respectively: insertion, deletion, substitution, expansion, contraction, and merger. Bisegments with three or more sentences in one language are not generally used, despite the fact that 1-to-3, 2-to-3, 3-to-3, 3-to-2, and 3-to-1 sentence translations are found in bitext once in a while. This is because alignment of such passages using only the sentence length feature is too inaccurate to make the extra parameters and computation worthwhile. Similarly, it generally suffices to ignore the state history, by tying all states so they share the same outgoing transition probability distribution, a simple multinomial distribution over the bisegment

* Brown et al. (1991) describe this FST as a single-state hidden Markov model (HMM) that emits bisegments (which they call "beads"). However, the type of bisegment must be stochastically selected each time the state is reached, and then its output probabilities must be conditioned on the bisegment type. This effectively splits the single state into separate states for each bisegment type, as described here.

types that can be estimated from a small hand-aligned corpus. Alternatively, EM can be used to estimate this distribution simultaneously with the segment length distributions.

The output distribution for each state (i.e., bisegment type) is modeled as a function of its monolingual segments' lengths as measured in subtokens (typically bytes, characters, or word subtokens). Given the bisegment type, the length $l_0 = t - s$ of its emitted segment $\mathbf{e}_{s..t}$ is determined. The length of each sentence token in the emitted segment is assumed to be independent, and to follow some distribution. This distribution is implicitly Poisson in the case of Gale and Church (1991b). In Brown et al. (1991), relative frequencies are used to estimate probabilities for short sentence lengths (up to approximately 80 simple English and French words), and the distribution for longer sentences is fit to the tail of a Poisson. Since the empirical distribution for short sentences is fairly Poisson-like,* these variations do not appear to have a significant impact.

Finally, the length $l_1 = v - u$ of the foreign segment $\mathbf{f}_{u..v}$ is determined, by assuming that its difference from the length of the emitted segment follows some distribution, usually a normal distribution. The following difference functions are used by Gale and Church (1991b) and Brown et al. (1991), respectively:

$$\delta(l_0, l_1) = \frac{(l_1 - l_0 c)}{\sqrt{l_0 s^2}} \tag{16.3}$$

$$\delta(l_0, l_1) = \log \frac{l_1}{l_0} \tag{16.4}$$

Aside from normalizing the mean and variance to one's arbitrary preference, the only decision is whether to take the logarithm of the sentence length difference, which sometimes produces a better fit to the empirical distribution.

These assumptions are sufficient to compute the probability of any bisegment. It is convenient to write a candidate bisegment as (*bisegment-type*, *s*, *u*) where its emitted segment begins with the *s*th passage in **e** and its foreign segment begins with the *u*th passage in **f** (since we assume total disjoint segmentations in both languages, *t* and *v* can be inferred). For instance, a candidate 1-to-2 bisegment that hypothesizes coupling $\mathbf{e}_{31..32}$ with $\mathbf{f}_{36..38}$ has the estimated probability $\hat{P}(\text{1-to-2}, 31, 36)$.

Given this generative model, the actual alignment algorithm relies on dynamic programming (Bellman 1957) to find the maximum probability alignment. For each bisegment, we take $Cost(p, r)$ to be its negative log probability, and we minimize Equation 16.2 subject to the constraint that the bisegmentation is bijective, and is a total cover of all sentence tokens in both languages. The recurrence has the structure of dynamic time-warping (DTW) models and is based on the fact that the probability of any sequence of bisegments can be computed by multiplying the probability of last bisegment with the total probability of all the bisegments that precede it. Let the minimum cost (maximum log probability) up to passages (t, v) be $\delta(t, v) = logP(\mathbf{e}_{0..t}, \mathbf{f}_{0..v})$ where $0 \leq t \leq T$ and $0 \leq v \leq V$. The recurrence chooses the best configuration over the possible types of the last bisegment.

1. **Initialization.**

$$\delta(0, 0) = 0 \tag{16.5}$$

2. **Recursion.**

$$\delta(t, v) = \min \begin{cases} \delta(t,v-1) - \log \hat{P}(\text{0-to-1}, t, v-1) \\ \delta(t-1,v) - \log \hat{P}(\text{1-to-0}, t-1, v) \\ \delta(t-1,v-1) - \log \hat{P}(\text{1-to-1}, t-1, v-1) \\ \delta(t-1,v-2) - \log \hat{P}(\text{1-to-2}, t-1, v-2) \\ \delta(t-2,v-1) - \log \hat{P}(\text{2-to-1}, t-2, v-1) \\ \delta(t-2,v-2) - \log \hat{P}(\text{2-to-2}, t-2, v-2) \end{cases} \tag{16.6}$$

Note that the form of the recurrence imposes a set of slope constraints on the time warping.

* For example, see Figure 16.4 in Brown et al. (1991).

The most significant difference between the methods is that the Gale and Church (1991b) method measures sentence token lengths in terms of number of character subtokens, whereas the Brown et al. (1991) method uses number of simple English/French word subtokens (as determined by European language specific heuristics relying primarily on whitespace and punctuation separators). Gale and Church (1993) report that using characters instead of words, holding all other factors constant, yields higher accuracy (in their experiment, an error rate of 4.2% for characters as compared to 6.5% for words). The reasons are not immediately obvious, though Gale and Church (1993) use variance measures to argue that there is less uncertainty since the number of characters is larger (117 characters per sentence, as opposed to 17 words). However, these experiments were conducted only on English, French, and German, whose large number of cognates improve the character length correlation.

Wu (1994) showed that for a large English and Chinese government transcription bitext, where the cognate effect does not exist, the Gale and Church (1991b) method is somewhat less accurate than for English, French, and German, although it is still effective. Church et al. (1993) show that sentence lengths are well correlated for the English and Japanese AWK manuals, but do not actually align the sentences. We know of no experimental results comparing character and word length methods on non-cognate languages; the lack of such experiments is in part because of the well-known difficulties in deciding word boundaries in languages such as Chinese (Chiang et al. 1992; Lin et al. 1992; Chang and Chen 1993; Lin et al. 1993; Wu and Tseng 1993; Sproat et al. 1994; Wu and Fung 1994).

Bitexts in some language pairs may exhibit highly dissimilar sentence and clause structures, leading to very different groupings than the simple bisegments we have been considering. For example, although the sentence byte length correlations are strong in the English–Chinese government transcriptions used by Wu (1994), Xu and Tan (1996) report that the CNS English–Chinese news articles they use have very different clause and sentence groupings, and therefore suggest generalizing the bisegments to allow many-clause-to-many-clause couplings.

In general, length-based techniques perform well for tightly translated bitexts. However, they are susceptible to misalignment in the case where the bitext contains long stretches of sentences with roughly equal length, as for example in dialogues consisting of very short utterances, or in itemized lists. One possible solution is discussed in the section on lexical techniques.

The basic algorithm is $O(TV)$ in both space and time, that is, approximately quadratic in the number of segments in the bitext. This is prohibitive for reasonably large bitexts. Three basic methods for circumventing this problem are banding, the hierarchical variant of iterative refinement, and thresholding.

Banding is often a feasible approach since the true alignment paths in many kinds of bitexts lie close enough to the diagonal as to fall within reasonable bands. However, banding is not favored when the other two approaches can be used, since they make fewer assumptions about the alignment path.

Hierarchical iterative refinement appears to work well for tightly translated bitexts such as government transcripts, which are typically organized as one document or section per session. Generally, only a few passes are needed: document/section (optional), paragraph, speaker (optional), and sentence.

Thresholding techniques prune the δ matrix so that some alignment prefixes are abandoned during the dynamic programming loop. Many variants are possible. Relative thresholding is more appropriate than absolute thresholding; the effect is to prune any alignment prefix whose probability is excessively lower than the most probable alignment prefix of the same length. Beam search approaches are similar, but limit the number of "live" alignment prefixes for any given prefix length. It is also possible to define the beam in terms of a upper/lower bounded range (v^-, v^+), so that the loop iteration that computes the tth column of $\delta(t, v)$ only considers $v^- \leq v \leq v^+$. This approach is similar to banding, except that the center of the band is dynamically adjusted during the dynamic programming loop.

Thresholding can cause large warps (deletions, insertions) to be missed. Chen (1993) suggests a resynchronization method to improve the robustness of relative thresholding schemes against large warps, based on monitoring the size of the "live" prefix set during the dynamic programming loop. If this set reaches a predetermined size, indicating uncertainty as to the correct alignment prefix, the presence of a large warp is hypothesized and the alignment program switches to a resynchronization mode. In

this mode, both sides of the bitext are linearly scanned forward from the current point, seeking rare words. When corresponding rare words are found in both sides, a resynchronization point is hypothesized. After collecting a set of possible resynchronization points, the best one is selected by attempting sentence alignment for some significant number of sentences following the candidate resynchronization point, taking the resulting probability as an indication of the goodness of this resynchronization point. Resynchronization can improve error rates significantly when the bitext contains large warps.

16.3.2 Lexical Sentence Alignment

The prototypical lexically based sentence alignment technique was proposed by Kay and Röscheisen (1988) in the first paper to introduce a heuristic iterative refinement solution to the bitext alignment problem.* The method employs banding, and has the structure shown in Algorithm 1.†

Algorithm 1 Lexical Sentence Align

1: Initialize the set of anchor (sentence) alignments $\mathcal{A}$
2: **repeat**
3: Compute the (sentence alignment) candidate space by banding between each adjacent pair of anchors
4: Collect word-similarity statistics from the candidate space
5: Build a bilexicon containing sufficiently similar words
6: Collect sentence-similarity statistics from the candidate space, with respect to the bilexicon
7: Add sufficiently confident candidates to the set of alignments $\mathcal{A}$
8: **until** no new candidates were added to $\mathcal{A}$
9: **return** $\mathcal{A}$

Although this algorithm can easily be implemented in a straightforward form, its cost relative to lexicon size and bitext length is prohibitive unless the data structures and loops are optimized. In general, efficient implementations of the length-based methods are easier to build and yield comparable accuracy, so this lexical algorithm is not as commonly used.

A notable characteristic of this algorithm is that it constructs a bilexicon as a by-product. Various criteria functions for accepting a candidate bilexeme are possible, rating either the word pair's degree of correlation or its statistical significance. For the correlation between a candidate bilexeme (w, x), Kay and Röscheisen (1988) employ Dice's coefficient (van Rijsbergen 1979),

$$\frac{2N(w,x)}{N(w)+N(x)} \tag{16.7}$$

whereas Haruno and Yamazaki (1996) employ mutual information (Cover and Thomas 1991),

$$\log \frac{NN(w,x)}{N(w)N(x)} \tag{16.8}$$

For the statistical significance, Kay and Röscheisen (1988) simply use frequency, and Haruno and Yamazaki (1996) use t-score:

$$\frac{P(w,x)-P(w)P(x)}{\sqrt{P(w,x)/N}} \tag{16.9}$$

* A simplified version of the method was subsequently applied to a significantly larger corpus by Catizone et al. (1989).
† For the sake of clarifying the common conceptual underpinnings of different alignment techniques, our description uses different terms from Kay and Röscheisen (1988). Roughly, our $\mathcal{A}$ is the Sentence Alignment Table, our candidate space is the Alignable Sentence Table, and our bilexicon is the Word Alignment Table.

In either case, a candidate bilexeme must exceed thresholds on both correlation and significance scores to be accepted in the bilexicon.

The bilexicon can be initialized with as many pre-existing entries as desired; this may improve alignment performance significantly, depending on how accurate the initial anchors are. Even when a good pre-existing bilexicon is available, accuracy is improved by continuing to add entries statistically, rather than "freezing" the bilexicon. Haruno and Yamazaki (1996) found that combining an initial seed bilexicon (40,000 entries from a commercial machine-readable translation dictionary) with statistical augmentation significantly outperformed versions of the method that either did not employ the seed bilexicon or froze the bilexicon to the seed bilexicon entries only. Results vary greatly depending on the bitext, but show consistent significant improvement across the board. Precision and recall are consistently in the mid-90% range, up from figures in the range of 60.

With respect to applicability to non-Indo-European languages, the Haruno and Yamazaki (1996) experiments show that the algorithm is usable for Japanese–English bitexts, as long as the Japanese side is presegmented and tagged. Aside from the variations already discussed, two other modified strategies are employed. To improve the discriminativeness of the lexical features, an English–Japanese word coupling is only allowed to contribute to a sentence-similarity score when the English word occurs in only one of the candidate sentences to align to a Japanese sentence. In addition, two sets of thresholds are used for mutual information and *t*-score, to divide the candidate word couplings into high-confidence and low-confidence classes. The high-confidence couplings are weighted three times more heavily. This strategy attempts to limit damage from the many false translations that may be statistically acquired, but still allow the low-confidence bilexicon entries to influence the later iterations when no more discriminative leverage is available from the high-confidence entries.

It is possible to construct lexical sentence alignment techniques based on underlying generative models that probabilistically emit bisegments, similar to those used in length-based techniques. Chen (1993) describes a formulation still using bisegment types with 0-to-1, 1-to-0, 1-to-1, 1-to-2, and 2-to-1 sentence tokens, but at the same time, a subtokenization level where each bisegment is also viewed as a multiset of bilexemes. Instead of distributions that govern bisegment lengths, we have distributions governing the generation of bilexemes. Word order is ignored,* since this would be unlikely to improve accuracy significantly despite greatly worsening the computational complexity. Even with this concession, the cost of alignment would be prohibitive without a number of additional heuristics. The basic method employs the same dynamic programming recurrence as Equation 16.6, with suitable modifications to the probability terms; the quadratic cost would be unacceptable, especially since the number of bilexeme candidates per bisegment candidate is exponential in the bisegment length. Chen employs the relative thresholding heuristic, with resynchronization. The probability parameters, which are essentially the same as for the length-based methods except for the additional bilexeme probabilities, are estimated using the Viterbi approximation to EM.

Like the Kay and Röscheisen (1988) method, this method induces a bilexicon as a side effect. For the method to be able to bootstrap this bilexicon, it is important to provide a seed bilexicon initially. This can be accomplished manually, or by a rough estimation of bilexeme counts over a small manually aligned bitext. The quality of bilexicons induced in this manner has not been investigated, although related EM-based methods have been used for this purpose (see below).

The alignment precision of this method on Canadian Hansards bitext may be marginally higher than the length-based methods. Chen (1993) reports 0.4% error as compared with 0.6% for the Brown et al. (1991) method (but this includes the effect of resynchronization that was not employed by Brown et al. (1991)). Alignment recall improves by roughly 0.3%. Since the accuracy and coverage of the induced bilexicon has not been investigated, it is unclear how many distinct bilexemes are actually needed to obtain this level of performance improvement. The running time of this method is "tens of times" slower than length-based methods, despite the many heuristic search approximations.

* Similarly, in spirit, to IBM word alignment Model 1, discussed elsewhere.

To help reduce the running time requirements, it is possible to use a cheaper method to impose an initial set of constraints on the coupling matrix. Simard and Plamondon (1996) employ a cognate-based method within a framework similar to SIMR (discussed later) to generate a set of anchors, and then apply banding.

The *word_align* method (Dagan et al. 1993) may also be used for lexical sentence alignment, as discussed in Section 16.4.2.

16.3.3 Cognate-Based Sentence Alignment

For some language pairs like English and French, the relatively high proportion of cognates makes it possible to use cognates as the key feature for sentence alignment. Simard et al. (1992), who first proposed using cognates, manually analyzed approximately 100 English-French bisentences from the Canadian Hansards, and found that roughly 21% of the words in bisentences were cognates. In contrast, only 6% of the words in randomly chosen non-translation sentence pairs were cognates.

Cognate-based alignment can in fact be seen as a coarse approximation to lexical alignment, where the bilexemes are based on a heuristic operational definition of cognates, rather than an explicitly listed lexicon. The Simard et al. (1992) method's cognate identification heuristic considers two words w, x to be cognates if

1. w, x are identical punctuation characters;
2. w, x are identical sequences of letters and digits, with at least one digit; or
3. w, x are sequences of letters with the same four-character prefix.

This heuristic is clearly inaccurate, but still discriminates true from false sentence pairs fairly well: it considers 30% of the words in bisentences to be cognates, versus only 9% in non-translation sentence pairs.

Given such a definition, any of the lexical alignment methods could be used, simply by substituting the heuristic for lexical lookup. The techniques that require probabilities on bilexemes would require that a distribution of suitable form be imposed. Simard et al. (1992) use the dynamic programming method; however, for the cost term, they use a scoring function based on a log-likelihood ratio rather than a generative model. The score of a bisegment containing c cognates and average sentence length n is

$$-\log\left(\frac{P(c|n,t)}{P(c|n,\neg t)} \cdot P(\textit{bisegment-type})\right) \tag{16.10}$$

The pure cognate-based method does not perform as well as the length-based methods for the Canadian Hansards. In tests on the same bitext sample, Simard et al. (1992) obtain a 2.4% error rate, where the Gale and Church (1991b) method yields a 1.8% error rate. The reason appears to be that even though the mean number of cognates differs greatly between coupled and non-coupled sentence pairs, the variance in the number of cognates is too large to separate these categories cleanly. Many true bisentences share no cognates, while many non-translation sentence pairs share several cognates by accident.

Cognate-based methods are inapplicable to language pairs such as English and Chinese, where no alphabetic matching is possible.

16.3.4 Multifeature Sentence Alignment

Higher accuracy can be obtained by combining sentence length, lexical, and/or cognate features in a single model. Since a length-based alignment algorithm requires fewer comparison tests than a lexical alignment algorithm, all other things being equal, it is preferable to employ length-based techniques when comparable accuracy can be obtained. However, as mentioned earlier, length-based techniques can misalign when the bitext contains long stretches of sentences with roughly equal length. In this

case, it is possible to augment the length features with others. One method of combining features is by incorporating them into a single generative model. Alternatively, the less expensive length-based feature can be used in a first pass; afterward, the uncertain regions can be realigned using the more expensive lexical or cognate features. Uncertain regions can be identified either using low log probabilities in the dynamic programming table as indicators of uncertain regions, or using the size of the "live" prefix set during the dynamic programming loop.

The method of Wu (1994) uses a single generative model that combines sentence length with lexical (bi)subtoken features, which in their case were English words and Chinese characters. A very small translation lexicon of subtokens is chosen before alignment; to be effective, each subtoken should occur often in the bitext and should be highly discriminative, i.e., the translation of each chosen subtoken should be as close to deterministic as possible. The total set of chosen subtokens should be small; otherwise, the model degenerates into full lexical alignment with all the computational cost. The model assumes that all subtokens in a bisegment, except for those that belong to the set of chosen subtokens, are generated according to the same length distribution as in the basic length-based model. The dynamic programming algorithm is generalized to find the maximum probability alignment under this hybrid model. Significant performance improvement can typically be obtained with this method, though the degree is highly dependent upon the bitext and the bitoken vocabulary that is chosen.

Simard et al. (1992) combine sentence length with cognate features using a two-pass approach. The first pass is the Gale and Church (1991b) method which yields a 1.8% error rate. The second pass employs the cognate-based method, improving the error rate to 1.6%.

16.3.5 Comments on Sentence Alignment

Sentence alignment can be performed fairly accurately regardless of the corpus length and method used. Perhaps surprisingly, this holds even for lexicon-learning alignment methods, apparently because the decreased lexicon accuracy is offset by the increased constraints on alignable sentences.

The advantages of the length-based method are its ease of implementation and speed. Although it is sometimes argued that the asymptotic time complexity of efficiently implemented lexical methods is the same as length-based methods, even then the constant factor for would be significantly costlier than the length-based methods.

The advantages of the lexical methods are greater robustness, and the side effect of automatically extracting a lexicon.

Multifeature methods appear to offer the best combination of running time, space, accuracy, and ease of implementation for aligning sentences in tightly translated bitexts.

16.4 Character, Word, and Phrase Alignment

We now consider the case where the vectors **e** and **f** denote sentences that are sequences of lexical units or *lexemes*—character or word/phrase tokens—to be aligned (rather than documents, sections, or paragraphs that are sequences of sentence tokens, as in the preceding section).

The first new difficulty is how to define the input "word" tokens. Compared to paragraph or sentence tokens, it is significantly more slippery to define what a *word* token is. The frequently used, but simplistic, tokenization assumption that whitespace and punctuation separates "words" does not really hold even for Western European languages—as for example, in the multi-"word" words *put off*, *by the way*, *inside out*, *thank you*, *roller coaster*, *break a leg*, and so on. For many other major languages, the simplifying assumption is even less workable.

In Chinese, for example, which has tens of thousands of unique characters and is written without any spaces, any attempt to heuristically guess where "word" boundaries might lie is inherently highly error-prone. Premature commitment to artificial "word" boundaries significantly impacts alignment and

translation accuracy, since the tokens may have been wrongly segmented to begin with. For this reason, a conservative approach is to avoid *a priori* word tokenization altogether, and assume character tokens instead.

Many simpler word alignment models in fact perform only *token alignment*, where all bisegments are one token or zero tokens long, in both languages. Other models tackle the more realistic general case of *multitoken alignment* where the lexemes being aligned may be words/phrases spanning multiple tokens.

For convenience, we will often use "word" and "lexeme" to refer generally to multitoken lexical units that may be characters, phrases, compounds, multi-word expressions, and/or collocations.

16.4.1 Monotonic Alignment for Words

For languages that share very similar constituent ordering properties, or simply to obtain a very rough word alignment as a bootstrapping step toward more sophisticated word alignments, it is possible to apply many of the techniques discussed for sentence alignment and noisy bitext alignment. For such applications, we may make the simplifying assumption that word order is more or less monotonically preserved across translation. The same dynamic programming algorithms for monotonic alignment of Section 16.3.1 can be used, for example, to perform multitoken alignment yielding bisegments that couple 0-to-1, 1-to-0, 1-to-1, 1-to-2, or 2-to-1 *lexeme* tokens.

Unless the languages are extremely similar, however, length-based features are of low accuracy. In general, useful features will be the lexical features, and for some language pairs also the cognate features.

16.4.2 Non-Monotonic Alignment for Single-Token Words

For true coupling of words (as opposed to simple linear interpolation up to the word granularity), generally speaking, we must drop the monotonicity constraint. Realistically, translating a sentence requires *permuting* (or *reordering*) its component lexemes. Motivations to move to a more accurate lexical coupling model include

Alignment accuracy.
: In principle, lexical coupling without false monotonicity assumptions leads to higher accuracy.

Sparse data.
: Smaller bitexts should be alignable, with bilexicons automatically extracted in the process. The lexical alignment models discussed above require large bitexts to ensure that the counts for good bilexemes are large enough to stand out in the noisy word-coupling hypothesis space.

Translation modeling.
: Accurate lexical coupling is necessary to bootstrap learning of structural translation patterns.

Complexity of coupling at the word level rises as a result of relaxing the monotonicity constraint. Partly for this reason, the assumption is usually made that the word coupling cannot be many-to-many. On the other hand, models that permit one-to-many couplings are feasible. The TM-Align system (Macklovitch and Hannan 1996) employs such a model, namely IBM Model 3 (Brown et al. 1988, 1990, 1993). This model incorporates a *fertility* distribution that governs the probability on the number of language-1 words generated by a language-0 word. Macklovitch and Hannan (1996) report that 68% of the words were correctly aligned. Broken down more precisely, 78% of the content words are correctly aligned, compared to only 57% of the function words.*

The *word_align* method (Dagan et al. 1993) employs a dynamic programming formulation similar to that discussed in Section 16.3.1, but has slope constraints that allow the coupling order to move slightly

* Although the IBM statistical translation models are all based on word alignment models, they themselves have not reported accuracy rates for word alignment.

backward as well as forward. The method does not involve any coarser (section, paragraph, sentence) segmentation of the bitexts. The output of the basic algorithm is a partial word alignment.

The method requires as input a set of alignment range constraints, to restrict the search window to a feasible size inside the dynamic programming loop. Dagan et al. (1993) employ *char_align* as a preprocessing stage to produce a rough alignment $\mathbf{a}^0(\cdot)$, and then use guiding to restrict the alignments considered by *word_align*.

An underlying stochastic channel model is assumed, similar to that of the IBM translation model (Brown et al. 1990, 1993). We assume each language-1 word is generated by a language-0 word. Insertions and deletions are permitted, as in the sentence alignment models. However we can no longer simplistically assume that bisegments describe both sides of the bitext in linear order,* since the monotonicity assumption has been dropped: the language-1 words do not necessary map to language-0 words in order. To model this, a new set of *offset probabilities* $o(k)$ are introduced. The offset k of a pair of coupled word positions t, v is defined as the deviation of the position t of the language-0 word from where it "should have been" given where v is

$$k(t, v) = t - \mathbf{a}'(v) \tag{16.11}$$

where $\mathbf{a}'(v)$ is where the language-0 word "should have been." We determine this by linear extrapolation along the slope of the diagonal, from the language-0 position $\mathbf{a}(v^-)$ corresponding to the previous language-1 word:

$$\mathbf{a}'(v) = \mathbf{a}(v^-) + (v - v^-)\frac{T}{V} \tag{16.12}$$

where v^- is the position of the most recent language-1 word to have been coupled to a language-0 word (as opposed to being an insertion). Given a language-0 string $\mathbf{w}$, the channel model generates the language-1 translation $\mathbf{x}$ with the probability

$$P(\mathbf{x}|\mathbf{w}) = \sum_{\mathbf{a}} K \prod_{v=0}^{V-1} P(x_v|w_{a(v)}) \cdot o(\mathbf{a}(v) - \mathbf{a}'(v)) \tag{16.13}$$

In theory this is computed over all possible alignments $\mathbf{a}$, but we approximate by considering only those within the alignment range constraints:†

$$P(\mathbf{x}|\mathbf{w}) = \sum_{\mathcal{A}\subset\mathcal{B}} K \prod_{(t,v)\in\mathcal{A}} P(x_v|w_t) \cdot o(t - \mathbf{a}'(v)) \tag{16.14}$$

Assuming we have estimated distributions $\hat{P}(x|w)$ and $\hat{o}(k)$, the alignment algorithm searches for the most probable alignment $\mathcal{A}*$:

$$\mathcal{A}* = \arg\max_{\mathcal{A}\subset\mathcal{B}} \prod_{(t,v)\in\mathcal{A}} P(x_v|w_t) \cdot o(t - \mathbf{a}'(v)) \tag{16.15}$$

The dynamic programming search is similar to the sentence alignment cases, except on the word granularity and with different slope constraints. It considers all values of v proceeding left-to-right through the words of the language-1 side, maintaining a hypothesis for each possible corresponding value of t. The

* There are no strings of "beads" as in Brown et al. (1991).
† Remember that $\mathcal{A}$ and $\mathbf{a}$ refer to the same alignment.

recurrence is as follows. Let $\delta(t, v)$ denote the minimum cost partial alignment up to the vth word of the language-1 side, such that the vth word is coupled to the tth position of the language-0 side.

$$\delta(t,v) = \min \begin{cases} \min_{t^-:(t^-,v^-)\subset\mathcal{B}\wedge t^- - d\le t\le t^- + d} \delta(t^-,v-1) - \log\hat{P}(x_v|w_t) - \log\hat{o}(t-\mathbf{a}'(v)) \\ \delta(t,v-1) - \log\hat{P}(x_v|\epsilon) \\ \delta(t-1,v) - \log\hat{P}(\epsilon|w_t) \end{cases} \tag{16.16}$$

$$= \min \begin{cases} \min_{t^-:(t^-,v^-)\subset\mathcal{B}\wedge t^- - d\le t\le t^- + d} \delta(t^-,v-1) - \log\hat{P}(x_v|w_t) - \log\hat{o}\left(t-\left(\mathbf{t}^- + (v-v^-)\frac{C}{D}\right)\right) \\ \delta(t,v-1) - \log\hat{P}(x_v|\epsilon) \\ \delta(t-1,v) - \log\hat{P}(\epsilon|w_t) \end{cases} \tag{16.17}$$

$$\tag{16.18}$$

The t^- values that need to be checked are restricted to those that meet the alignment range constraints, and lie within d of t. In practice, the probabilities of word insertion and deletion, $\hat{P}(x_v|\epsilon)$ and $\hat{P}(\epsilon|w_i)$, can be approximated with small flooring constants.

The offset and bilexeme probability estimates, $\hat{o}(k)$ and $\hat{P}(x|w)$, are estimated using EM on the bitext before aligning it. The expectations can be accumulated with a dynamic programming loop of the same structure as that for alignment. An approximation that Dagan et al. (1993) make during the EM training is to use the initial rough alignment $\mathbf{a}^0(v)$ instead of $\mathbf{a}'(v)$.

An experiment performed by Dagan et al. (1993) on a small 160,000-word excerpt of the Canadian Hansards produced an alignment in which about 55% of the words were correctly aligned, 73% were within one word of the correct position, and 84% were within three words of the correct position.

Quantitative performance of *word_align* on non-alphabetic languages has not been extensively evaluated, but Church et al. (1993) align the English and Japanese AWK manuals using the technique. The Japanese text must first be segmented into words, which can lead to an incorrectly segmented text.

It is possible to use *word_align* as a lexical sentence alignment scheme, simply by post-processing its output down to the coarser sentence granularity. The output set of word couplings can be interpreted as a set of candidate anchors, of higher credibility than the usual set of possible word couplings. Using dynamic programming again, the post-processor may choose the set of sentence bisegments that maximizes the (probabilistically weighted) coverage of the candidate anchor set. However, it is not clear that the additional word-order model would significantly alter performance over a model such as that of Chen (1993).

Many single-token word alignment methods, particularly the IBM models [See Section 17.7.1] are inherently asymmetric, leading to artifacts in the alignments that can degrade translation accuracy. To compensate, *symmetrization* methods are commonly used to improve the alignments (Och and Ney 2003). IBM models are first trained in both directions, giving two different token alignments $\mathcal{A}_{ef}$ and $\mathcal{A}_{fe}$ for any sentence pair. A family of various types of symmetrization heuristics can then be applied, for example,

- Intersection: $\mathcal{A} = \mathcal{A}_{ef} \cap \mathcal{A}_{fe}$
- Union: $\mathcal{A} = \mathcal{A}_{ef} \cup \mathcal{A}_{fe}$
- Grow: Start with intersection. Iteratively extend $\mathcal{A}$ by adding token couplings (t, t, v, v) that are found in only one of $\mathcal{A}_{ef}$ or $\mathcal{A}_{fe}$, providing that neither $\mathbf{e}_t$ nor $\mathbf{f}_v$ has a coupling in $\mathcal{A}$, or the alignment (t, t, v, v) has a horizontal neighbor $(t-1, t-1, v, v)$ or $(t+1, t+1, v, v)$ or a vertical neighbor $(t, t, v-1, v-1)$ or $(t, t, v+1, v+1)$.
- Grow-diag: Same as grow, but allow diagonal neighbors as well (Koehn et al. 2003).
- Grow-diag-final-and: Same as Grow-diag but as a final step, also add non-neighbor token couplings of words that otherwise meet the requirements (Koehn et al. 2003).

A simple variant of the ITG model of Wu (1997) restricted to 1-to-1 token alignment can also be trained via EM Wu (1995d) to produce highly accurate single-token bisegmentations (Section 16.6).

```
[.... graphical examples including a complete m-to-n coupling
plus various sparse and medium-density couplings, all attempting
to describe exactly the same coupling of an m-token lexeme to an
n-token lexeme ....]
```

FIGURE 16.9 Multitoken lexemes of length m and n must be coupled in awkward ways if constrained to using only 1-to-1 single-token bisegments.

16.4.3 Non-Monotonic Alignment for Multitoken Words and Phrases

The techniques of the previous section make the simplifying assumption that a word/lexeme is a single token, i.e., that bisegments couple only single tokens. When confronted with the more realistic case where an m-token lexeme needs to be coupled to an n-token lexeme, such models must resort to using some subset of the $m \times n$ possible 1-to-1 single-token couplings, as for example in Figure 16.9. Ultimately, this becomes problematic, because it forces models to make artificial, meaningless distinctions between the many arbitary variations like those exemplified in Figure 16.9. By failing to capture straightforward sequence patterns, the 1-to-1 single-token simplification wastes probability mass on unnecessary alignment ambiguities, and creates needless computational complexity.

These problems do not arise in word alignment with m-to-n bisegments that couple multiple tokens (such as we also used for aligning sentences). Because this impacts translation quality significantly, increasingly, alignment is seen in terms of finding segmentations of multitoken words/lexemes to align. This is commonly emphasized by terms like "multi-word expression," "phrase," and "phrase alignment."

The key to multitoken alignment is to integrate the segmentation decisions into the alignment cost optimization, since the optimal coupling of segments depends on how **e** and **f** are segmented, and vice versa.

IBM Models 4 and 5 support a limited asymmetric form of 1-to-n bisegments (Brown et al. 1993), but still do not cope well with multitoken words/lexemes since they cannot occur on the **e** side.

In contrast, the general ITG model introduced by Wu (1997) fully integrates alignment and segmentation, producing multitoken m-to-n word/phrase alignments subject to compositionally structured constraints on lexeme and segment permutations, using an underlying generative model based on a stochastic transduction grammar (Section 16.6).

Generally speaking, multitoken alignment is best performed with respect to some bilexicon that enumerates possible translations for multitoken lexemes—i.e., a list of all legal bisegment types.* Thus, there are two steps to multitoken alignment: (1) induce a bilexicon, and (2) compute a minimum-cost bisegmentation of the bitext using the induced bilexicon.

In practice, very simple methods for inducing a (phrasal) bilexicon perform surprisingly well, if a good 1-to-1 token alignment is available to bootstrap from. A commonly used method is simply to take as a bilexeme every possible bisegment that is consistent with the 1-to-1 token alignment seen in training sentence pairs, up to some maximum token length. Bilexeme probabilities are estimated simply using relative frequency, and (optionally) very low-frequency bilexemes may be discarded. Although the accuracy of multitoken alignment was not directly evaluated, experimental results from Koehn et al. (2003) indicate that when evaluated on translation accuracy, this approach outperforms more complicated methods that filter out bisegments violating syntactic constituent boundaries.

On the other hand, the quality of the induced bilexicon has proven to be fairly sensitive to the quality of the initial 1-to-1 token alignment. The method of Koehn et al. (2003) employs the intersection of IBM alignments trained via EM in both directions, supplemented with the grow-diag-final-and family of heuristics discussed in Section 16.4.2. However, comparative experiments by Saers and Wu (2009) indicate that higher accuracy is obtained by using ITG alignments trained via EM and restricted to 1-to-1 token alignment.

* Often referred to as a *phrase table* in work on phrase-based SMT.

Given an induced bilexicon, multitoken alignment of any sentence pair can then be obtained by computing a minimum-cost bisegmentation, for example by using the dynamic programming method of the ITG biparsing model (Section 16.6) or greedy or heuristic variants.

16.5 Structure and Tree Alignment

A *structure alignment* algorithm produces an alignment between constituents (or sentence substructures) within sentence pairs of a bitext. The segments to be aligned are labeled by the nodes in the constituent analyses of the sentences; the output set $\mathcal{A}$ contains pairs of labeled segments (p, r) corresponding to the coupled nodes. All existing techniques process the bitext one bisentence at a time, so sentence alignment always precedes structure alignment. Coupled constituents may be useful in translators' concordances, but they are usually sought for use as examples in EBMT systems (Nagao 1984), phrase-based SMT systems (Koehn et al. 2003), and tree-based SMT systems like those of Wu (1997) and Chiang (2005). Alternatively, the examples constitute training data for machine learning of transfer patterns.

Tree alignment is the special case of structure alignment where the output $\mathcal{A}$ must be a strictly compositional, hierarchical alignment. (The same constraint is applicable to dependency tree models.) This means that tree alignment obeys the following:

Crossing constraint

Suppose two nodes in language-0 p_0 and p_1 correspond to two nodes in language-0 r_0 and r_1 respectively, and p_0 dominates p_1. Then r_0 must dominate r_0.

In other words, couplings between subtrees cannot cross each another, unless the subtrees' immediate parent nodes are also coupled to each other. Most of the time this simplifying assumption is accurate, and it greatly reduces the space of legal alignments and thereby the search complexity. An example is shown in Figure 16.10, where both *Security Bureau* and *police station* are potential lexical couplings to 公安局,

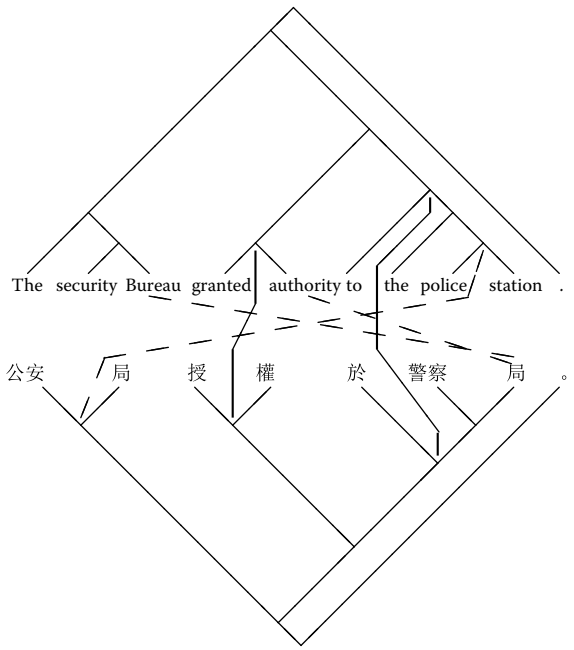


FIGURE 16.10 The crossing constraint.

but the crossing constraint rules out the dashed-line couplings because of the solid-line couplings. The crossing constraint reflects an underlying cross-linguistic hypothesis that the core arguments of frames tend to stay together over different languages. A special case of the crossing constraint is that a constituent will not be coupled to two disjoint constituents in the other language, although it may be coupled to multiple levels within a single constituent subtree.

Structure alignment is usually performed using a *parse-parse-match* strategy. Tree alignment methods require as input the constituent analysis of each side of the bitext (with the exception of the biparsing methods described later). Unfortunately, it is rarely possible to obtain bitext in which the constituent structures of both sides have been marked. However, if suitable monolingual grammars for each of the languages are available, each side can (independently) be parsed automatically, yielding a low-accuracy analysis of each sides, before the tree alignment begins. A variant on this is to supply alternative parses for each sentence, either explicitly in a list (Grishman 1994) or implicitly in a well-formed substring table (Kaji et al. 1992). Note that the parsed bitext is not parallel in the sense that corresponding sentences do not necessarily share a parallel constituent structure. It is difficult, if not impossible, to give an interpretation based on some underlying generative model.

The kind of structures that are to be coupled clearly depend on the linguistic theory under which the sides are parsed. The simplest approach is to use surface structure, which can be represented by bracketing each side of the bitext (Sadler and Vendelmans 1990; Kaji et al. 1992). Another approach was described by Matsumoto et al. (1993), who use LFG-like (Bresnan 1982) unification grammars to parse an English–Japanese bitext. For each side, this yields a set of candidate feature-structures corresponding to the constituent structures. The feature-structures are simplified to dependency trees. Structural alignment is then performed on the dependency trees, rather than the original constituent trees. (Alignment of dependency trees can be performed in essentially the same way as alignment of constituent trees.)

16.5.1 Cost Functions

Various cost functions may be employed. Some of the qualitative desiderata, along with exemplar cost functions, are as follows.

Couple leaf nodes (words/lexemes) that are lexical translations.

This is simply word alignment, recast here as the special case of structure alignment at the leaf level. A bilexicon is assumed. Filtering heuristics may be employed, for example, to ignore any words that are not open-class or content words. The simplest cost function of this type is

$$Cost(p,r) = \begin{cases} -1 & \text{if } (w,x) \in \text{Bilexicon} \\ & \quad \text{where } w = \mathbf{e}_{\text{span}(p)}, x = \mathbf{f}_{\text{span}(r)} \\ 0 & \text{otherwise} \end{cases} \tag{16.19}$$

where, in other words, the pair of leaf nodes p, r holds the words w and x.

Alternatively, if a probabilistic bilexicon is available, a soft version of the cost function may be used, in which the negative log probability of a word pair is taken as its cost (instead of -1).

Couple leaf nodes (words/lexemes) that are similar.

To overcome the spotty coverage of most bilexicons, thesauri may be employed. One approach (Matsumoto et al. 1993) employs a single language-0 thesaurus to estimate the cost (dissimilarity) between a word pair w, x. All possible translations of the language-1 word x are looked up in the language-0 thesaurus, and the length l of the shortest path from any one of them to w is taken as the cost. Matsumoto et al. (1993) use an additional biasing heuristic that always subtracts a constant d from the path length to get l (if the path length is less than the d, then l is assumed to be zero).

$$Cost(p,r) = \begin{cases} -6 & \text{if } (w,x) \in \text{Bilexicon} \\ & \quad \text{where } w = \mathbf{e}_{\text{span}(p)}, x = \mathbf{f}_{\text{span}(r)} \\ l & \text{otherwise} \end{cases} \tag{16.20}$$

Couple internal nodes that share coupled leaf nodes (words/lexemes).

Let $\mathcal{A}_{words}$ denote the subset of $\mathcal{A}$ that deals with coupling of leaf nodes (usually this subset is precomputed in an earlier stage, according to one of the criteria above and possibly with additional constraints). Let $Keywords(p)$ be the set of leaves of p that are deemed important, for example, all the content words. The simplest cost function of this type is

$$Cost(p,r) = \begin{cases} -1 & \text{if } (w,x) \in \mathcal{A}_{words} \\ & \quad \text{for all } w \in Keywords(p), x \in Keywords(r) \\ 0 & \text{otherwise} \end{cases} \tag{16.21}$$

This cost function is what Kaji et al. (1992) in effect use, permitting nodes to be matched only if they share exactly the same set of coupled content words.

A softer approach is to maximize the number of shared coupled leaf nodes.

$$Cost(p,r) = \sum_{w \in Keywords(p), x \in Keywords(r)} Cost(w,x) \tag{16.22}$$

Again, probabilistic versions of these cost functions are straightforward, if a probabilistic bilexicon is available.

Couple nodes that share as many coupled children/descendants as possible.

Similar *ad hoc* cost functions to those above can be formulated. The idea is to maximize structural isomorphism. Through the recursion, this also attempts to share as many coupled leaf nodes as possible.

16.5.2 Algorithms

Alignment still optimizes Equation 16.2, but the bisegments may now be either compositional or disjoint.

In general, constituent alignment has lower complexity than tree alignment, as there is no need to enforce the crossing constraint. Kaji et al. (1992) describe a simple bottom-up greedy strategy. Suppose that a bracketed sentence pair contains P language-0 nodes and R language-1 nodes. The algorithm simply considers all $P \times R$ possible couplings between node pairs p, r, starting with the smallest spans. Any pair whose cost is less than a preset threshold is accepted. Thus, as many pairs as possible are output, so long as they meet the threshold.

For tree alignment as opposed to constituent alignment, bottom-up greedy search is still possible but performance is likely to suffer due to the interaction of the coupling hypotheses. Crossing constraints deriving from early miscouplings can easily preclude later coupling of large constituents, even when the later coupling would be correct. For this reason, more sophisticated strategies are usually employed.

Matsumoto et al. (1993) employ a branch-and-bound search algorithm. The algorithm proceeds top-down, depth-first. It hypothesizes constituent couplings at the highest level first. For each coupling that is tried, a lower bound on its cost is estimated. The algorithm always backtracks to expand the hypothesis with the lowest expected cost. The hypothesis is expanded by further hypothesizing couplings involving the constituent's immediate children (subconstituents).

Grishman (1994) employs a beam search. Search proceeds bottom-up, hypothesizing couplings of individual words first. Only hypotheses whose cost is less than a preset threshold are retained. Each step in the search loop considers all couplings involving the next larger constituents consistent with the current set of smaller hypotheses. The costs of the larger hypotheses depend on the previously computed costs of the smaller hypotheses.

Dynamic programming search procedures can be constructed. However, the worst case complexity is exponential, since it grows with the number of permutations of constituents at any level. The procedure of Kaji et al. (1992) employs two separate well-formed substring tables as input, one for each sentence of the input sentence pair, that are computed by separate dynamic-programming chart parsing processes. However, the actual coupling procedure is greedy, as described above. A true dynamic programming approach (Wu 1995c) is described later in Section 16.6.4 on biparsing alignment (in which permutation constraints guarantee polynomial time complexity).

16.5.3 Strengths and Weaknesses of Structure and Tree Alignment Techniques

The most appropriate application of constituent alignment approaches appears to be for EBMT models. Given the same bitext, constituent alignment approaches can produce many more examples than strict tree alignment approaches (driving up recall), though many of the examples are incorrect (driving down precision). However, the nearest-neighbor tactics of EBMT models have the effect of ignoring incorrect examples most of the time, so recall is more important than precision.

Strict tree alignment approaches tend to have higher precision, since the effects of local disambiguation decisions are propagated to the larger contexts, and vice versa. For this reason, these methods appear to be more suitable for machine learning of transfer patterns.

Constituent alignment has three main weaknesses stemming from the parse-parse-match approach:

Lack of appropriate, robust, monolingual grammars.
This condition is particularly relevant for many low-resource languages. A grammar for this purpose must be robust since it must still identify constituents for the subsequent coupling process even for unanticipated or ill-formed input sentences.

Mismatch of the grammars across languages.
The best-matching constituent types between the two languages may not include the same core arguments. While grammatical differences can make this problem unavoidable, there is often a degree of arbitrariness in a grammar's chosen set of syntactic categories, particularly if the grammar is designed to be robust. The mismatch can be exacerbated when the monolingual grammars are designed independently, or under different theoretical considerations. For example, the negative results of Och et al. (2004) were widely interpreted to be due to the mismatch between independent parsers for the different languages.

Inaccurate selection between multiple possible constituent couplings.
A constituent in one sentence may have several potential couplings to the other, and the coupling heuristic may be unable to discriminate between the options.

16.6 Biparsing and ITG Tree Alignment

Bilingual parsing approaches use the same grammar to simultaneously parse both sides of a bitext (Wu 1995a). In contrast to the parse-parse-match approaches discussed in Section 16.5.3, biparsing approaches readily admit interpretations based on underlying generative *transduction grammar* models. Recall from Section 16.3.1 that a single transduction grammar governs the production of sentence pairs that are mutual translations. The most useful biparsers are the probabilistic versions, which work with stochastic transduction grammars that in fact constitute structured *bilingual language models*.

Biparsing approaches unify many of the concepts discussed above. The simplest version accepts sentence-aligned, unparsed bitext as input. The result of biparsing includes parses for both sides of the bitext, plus the alignment of the constituents.

There are several major variants of biparsing. It is possible to perform biparsing alignment *without* a linguistic grammar of either language, using the BITG technique discussed below. On the other hand, biparsing can make use of a monolingual grammar if one exists. If any *a priori* brackets on the input bitext are available, biparsing can accept them as constraints.

16.6.1 Syntax-Directed Transduction Grammars (or Synchronous CFGs)

Aside from the finite-state transductions considered in Section 16.3.1, the other major equivalence class of transductions from classic compiler theory is the class of *syntax-directed transductions*, which are transductions that can be generated by a *syntax-directed transduction grammar* or *SDTG* (Lewis and Stearns 1968; Aho and Ullman 1969,b; Aho and Ullman 1972), which have also recently been called "synchronous context-free grammars."*

As with the finite-state models in Section 16.3.1, formally $G = (\mathcal{N}, \mathcal{W}_0, \mathcal{W}_1, \mathcal{R}, S)$, the only difference being that the transduction rules in $\mathcal{R}$ are more expressive. The transduction rules in SDTGs superficially resemble the production rules in CFGs, except for generating terminal symbols on two streams instead of one, and allowing the right-hand-side symbols to be reordered by any permutation for language-1, as for example in $A \rightarrow a_0 a_1 a_2 a_3 :: a_1 a_3 a_0 a_2$ or simply $A \rightarrow a_0 a_1 a_2 a_3 :: 1\ 3\ 0\ 2$.

However, the similarity ends upon closer inspection. Unlike monolingual CFGs, bilingual SDTGs (or synchronous CFGS) do *not* form a single primary equivalence class. As seen in Section 16.3.1, in the Chomsky (1957) hierarchy, all CFGs form a single equivalence class, independent of the *rank*, which is the maximum number k of nonterminals in the rules' right-hand-sides. However, in the bilingual case of SDTGs, the hierarchy shatters into an infinite number of classes—for any $k > 4$, the class of SDTGs of rank k has strictly greater generative capacity than the class of SDTGs of rank $k - 1$. (The cases of $k = 2$ and $k = 3$ are suprisingly different, however, as in the discussion of ITGs below.)

In SDTGs (or synchronous CFGs), segment order variation (for words or phrases) between languages is accommodated by letting the symbols on each rule's right-hand-side appear in different order for language-0 and language-1. Any permutation of the right-hand-side symbols is allowed.

Any SDTG trivially implements the crossing constraint. This is because in the generative model, at any time the rewrite process only substitutes for a single constituent, which necessarily corresponds to contiguous spans in (both of) the sentences.

The main issue for alignment models based on transduction grammars is how much flexibility to allow. It is obvious that finite-state transductions are insufficiently expressive at a subsentential granularity, since they do not allow reordering of words and phrases across languages. SDTGs were widely used in rule-based MT, as well as example-based MT (Nagao 1984). However, computational complexity for SDTGs (or synchronous CFGs) is excessively expensive. No polynomial-time biparsing or parameter training algorithm is known for the general case, since growth in a number of possible alignments follows the number of permutations of right-hand-side symbols, which is exponential—resulting in algorithms that are $O(n^{2n+2})$, as seen in the hierachy in Section 16.3.1.

16.6.2 Inversion Transduction Grammars

In current state-of-the-art machine translation models, an intermediate equivalence class of transductions that offers a balance of generative capacity and computational complexity falling in between that of FSTGs and SDTGs (or synchronous CFGs) has become widely used, namely that of *inversion transduction grammars* or *ITGs*. Theoretical analyses and numerous empirical results have accumulated indicating a better fit of *inversion transductions* to modeling translation and alignment between many human

* The motivation for the standard term "syntax-directed" stems from the roots of SDTGs in compiler theory, where the input–output connotation of the term reflects a transduction view, but the theory of course remains compatible with the generation or recognition (biparsing) views as well. Also, "transduction" and "translation" are often used interchangeably, as are "grammar" and "schema."

language pairs, despite the fact that ITGs do not attempt to fully model so-called free word order languages and "second-order" phenomena such as raising and topicalization (Wu 1995b, 1997). Given that the polynomial-time biparsing and training algorithms for ITGs are much less expensive than for SDTGs—no more than $O(n^6)$, as seen in the hierachy in Section 16.3.1—and translation accuracy is at the same time empirically *higher* using ITGs, it is often worth accepting the restricted expressiveness of ITG constraints upon the space of permitted reordering permutations.

For ITGs the transduction rules in $\mathcal{R}$ are more expressive than in finite-state models, but less expressive than in SDTGs (or synchronous CFGs). For example, inversion transductions do not permit the permutation described by the transduction rule example in Section 16.6.1. A simple example of a permutation allowed in inversion transductions is shown in Figure 16.1c and d. However, inversion transductions are surprisingly flexible; even the extreme reordering shown in Figure 16.1e and f falls within the class.

Inversion transductions can be described in at least three equivalent ways, as follows.

ITGs are transduction grammars with only straight or inverted rules.

This means the order of right-hand-side symbols for language-1 is either the same as language-0 (*straight* orientation) or exactly the reverse (*inverted* orientation). A straight rule is written $A \rightarrow [a_0 a_1 \ldots a_{r-1}]$, and an inverted rule is written $A \rightarrow \langle a_0 a_1 \ldots a_{r-1} \rangle$, where $a_i \in \mathcal{N} \cup \mathcal{X}$ and r is the rank of the rule.*

ITGs are the closest bilingual analogue of monolingual CFGs in that any ITG can be converted to a 2-normal form—which is not true for SDTGs (or synchronous CFGs) in general. In particular, a theorem in Wu (1995c) shows that for any inversion transduction grammar G, there exists an equivalent inversion transduction grammar G' where all rules are either lexical rules or binary rank syntactic rules (analogous to Chomsky normal form for monolingual CFGs), such that every rule takes one of the following forms:

S	$\rightarrow$	ϵ/ϵ
A	$\rightarrow$	x/y
A	$\rightarrow$	x/ϵ
A	$\rightarrow$	ϵ/y
A	$\rightarrow$	$[B\ C]$
A	$\rightarrow$	$\langle B\ C \rangle$

where as before x and y are segments of one or more tokens (so the bisegments are typically *phrasal*), and A, B, and C are any nonterminals. The theorem leads directly to the second characterization of ITGs.

ITGs are transduction grammars with rules of rank ≤ 2.

That is, any SDTG whose rules are all binary-branching is an ITG. The equivalence follows trivially from the fact that the only two possible permutations of a rank-2 right-hand-side are straight and inverted. This means that any SDTG (or synchronous CFG) of binary rank—having at most two nonterminals on the right-hand-side of any rule—is an ITG. (Similarly, any SDTG (or synchronous CFG) that is right regular is a FSTG.)

Thus, for example, any grammar computed by the binarization algorithm of Zhang et al. (2006) is an ITG. Similarly, any grammar induced following the hierarchical phrase-based translation method, which always yields a binary transduction grammar (Chiang 2005), is an ITG.

ITGs are transduction grammars with rules of rank ≤ 3.

It can be shown that all six possible permutations of a rank-3 right-hand-side can be generated using only straight and inverted rules in combination.

* This notation is a useful shorthand for the straight rule that can also be written in other forms such as $A \rightarrow a_0 a_1 \ldots a_{r-1} :: a_0 a_1 \ldots a_{r-1}$ or $A \rightarrow a_0 a_1 \ldots a_{r-1} :: 0\ 1 \ldots r-1$, and the inverted rule that can also be written in other forms such as $A \rightarrow a_0 a_1 \ldots a_{r-1} :: a_{r-1} \ldots a_1 a_0$ or $A \rightarrow a_0 a_1 \ldots a_{r-1} :: r-1 \ldots 1\ 0$.

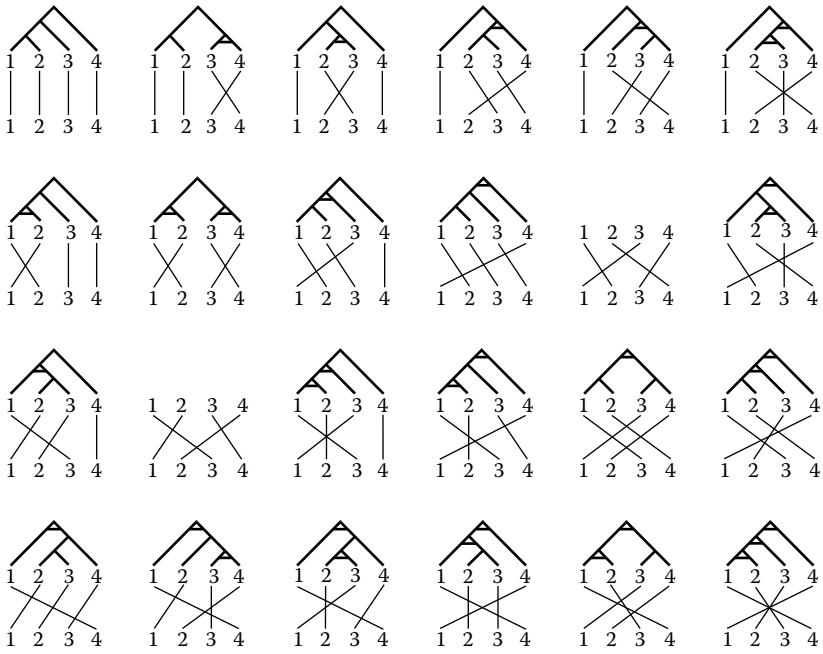


FIGURE 16.11 The 24 complete alignments of length four, with ITG parses for 22. All nonterminal and terminal labels are omitted. A horizontal bar under a parse tree node indicates an inverted rule.

The expressiveness of *ITG constraints* on reordering words, phrases, and constituents in different natural languages is not straightforward to characterize formally. Some light is shed by Figure 16.11, which enumerates how ITGs can deal with the transposition of four adjacent constituents. This case is important because the number of core arguments of a frame is normally less than four, in nearly all linguistic theories. There are $4! = 24$ possible permutations of four adjacent constituents, of which 22 can be produced by combining straight and inverted rules. The remaining two permutations are highly distorted "inside-out" alignments, which are extremely rare in (correctly translated) bitext.* For more than four adjacent constituents, many permutations cannot be generated and do not appear necessary.

The *ITG hypothesis* posits a language universal, namely that the core arguments of frames, which exhibit great ordering variation between languages, are relatively few and surface in syntactic proximity. Of course, this assumption over-simplistically blends syntactic and semantic notions, but the ITG hypothesis has held true empirically to a remarkably large extent. That semantic frames for different languages share common core arguments is more plausible than syntactic frames, but ITGs depend on the tendency of syntactic arguments to correlate closely with semantics. If in particular cases this assumption does not hold, the biparsing algorithm can attempt to contain the damage by dropping some word couplings (as few as possible). For more detailed analyses see Wu (1997) and Saers and Wu (2009).

16.6.3 Cost Functions

A cost function can be derived naturally from the generative model. First a stochastic version of the ITG is created by associating a probability with each rule. Just as for ordinary monolingual parsing, probabilizing the grammar permits ambiguities to be resolved by choosing the maximum likelihood parse. For example, the probability of the rule NN $\overset{0.4}{\rightarrow}$ [A N] is $a_{\text{NN}\rightarrow[\text{A N}]} = 0.4$. The probability of a lexical rule A $\overset{0.001}{\rightarrow}$ x/y

* In fact, we know of no actual examples in any parallel corpus for languages that do not have free word order.

is $b_A(x, y) = 0.001$. Let W_0, W_1 be the vocabulary sizes of the two languages, and $\mathcal{N} = \{A_0, \ldots, A_{N-1}\}$ be the set of nonterminals with indices $0, \ldots, N-1$. (For conciseness, we sometimes abuse the notation by writing an index when we mean the corresponding nonterminal symbol, as long as this introduces no confusion.) As with stochastic CFGs, the probabilities for a given left-hand-side symbol must sum to unity:

$$\sum_{1 \le j,k \le N} (a_{i \to [jk]} + a_{i \to \langle jk \rangle}) + \sum_{\substack{1 \le x \le W_0 \\ 1 \le y \le W_1}} b_i(x, y) = 1$$

The same constituent structure applies to both sides of the sentence pair unlike the earlier structure/tree alignment cases. This means the alignment $\mathcal{A}$ is more naturally thought of in terms of a single shared parse tree, rather than a set of couplings between constituents. We denote the parse tree by a set $\mathcal{Q}$ of nodes $\{q_0, \ldots, q_{Q-1}\}$. Note that $\mathcal{Q}$ can always be transformed back into a set of couplings $\mathcal{A}$.

The natural cost function to be minimized is the entropy (negative log probability) of the parse tree $\mathcal{Q}$:

$$Cost(\mathcal{Q}) = \sum_{q \in \mathcal{Q}} \begin{cases} a_{Rule(q)} & \text{if } q \text{ is an internal node} \\ b_{Rule(q)} & \text{otherwise} \end{cases} \tag{16.23}$$

16.6.4 Algorithms

The biparsing algorithm searches for the minimum cost parse tree on the input sentence pair, and selects the optimal segmentation at the same time. Empirically, this integrated *translation-driven segmentation* is highly effective at taking advantage of the phrasal terminals of ITGs to avoid alignment errors commonly caused by word segmenters that prematurely commit to inappropriate segmentations during preprocessing. The probabilistic cost function optimizes the overlap between the structural analysis of the two sentences. The algorithm resembles the recognition algorithm for HMMs (Viterbi 1967) and CKY parsing (Kasami 1965; Younger 1967).

Let the input English sentence be $\mathbf{e}_0, \ldots, \mathbf{e}_{T-1}$ and the corresponding input Chinese sentence be $\mathbf{f}_0, \ldots, \mathbf{f}_{V-1}$. As an abbreviation we write $\mathbf{e}_{s..t}$ for the sequence of words $\mathbf{e}_s, \mathbf{e}_{s+1}, \ldots, \mathbf{e}_{t-1}$, and similarly for $\mathbf{f}_{u..v}$; also, $\mathbf{e}_{s..s} = \epsilon$ is the empty string.

Assuming an ITG in 2-normal form, it is convenient to use the 4-tuple (s, t, u, v) to uniquely identify each node of the parse tree $q = \mathbf{seg}(s, t, u, v)$, where the substrings $\mathbf{e}_{s..t}$ and $\mathbf{f}_{u..v}$ both derive from the node q. Denote the nonterminal label on q by $\ell(q)$. Then for any node $q = \mathbf{seg}(s, t, u, v)$, define

$$\delta_q(i) = \delta_{stuv}(i) = \max_{\text{subtrees of } q} P[\text{subtree of } q, \ell(q) = i, i \stackrel{*}{\Rightarrow} \mathbf{e}_{s..t}/\mathbf{f}_{u..v}]$$

as the maximum probability of any derivation from i that successfully parses both $\mathbf{e}_{s..t}$ and $\mathbf{f}_{u..v}$. Then the best parse of the sentence pair has probability $\delta_{0,T,0,V}(\mathrm{S})$.

The algorithm computes $\delta_{0,T,0,V}(\mathrm{S})$ using the following recurrences. Note that optimal segmentation in both languages is integrated into the algorithm: $s..t$ and $u..v$ are spans delimiting segments of one or more tokens; if $s = t$ or $u = v$, the span length is zero, meaning that the segment is the empty string ϵ. The argmax notation is generalized to the case where maximization ranges over multiple indices, by making the argument vector-valued. Note that $[\,]$ and $\langle\rangle$ are just constants. The condition $(S - s)(t - S) + (U - u)(v - U) \neq 0$ is a way to specify that the substring in one but not both languages may be split into an empty string ϵ and the substring itself; this ensures that the recursion terminates, but permits words that have no match in the other language to map to an ϵ instead.

1. **Initialization**

$$\delta^0_{stuv}(i) = b_i(\mathbf{e}_{s..t}/\mathbf{c}_{u..v}), \quad \begin{matrix} 0 \le s \le t \le T \\ 0 \le u \le v \le V \end{matrix} \tag{16.24}$$

2. Recursion

For all i, s, t, u, v such that $\begin{cases} 1 \le i \le N \\ 0 \le s < t \le T \\ 0 \le u < v \le V \\ t-s+v-u > 2 \end{cases}$

$$\delta_{stuv}(i) = \max\left[\delta^{[\,]}_{stuv}(i), \delta^{\langle\rangle}_{stuv}(i), \delta^{0}_{stuv}(i)\right] \tag{16.25}$$

$$\theta_{stuv}(i) = \begin{cases} [\,] & \text{if } \delta^{[\,]}_{stuv}(i) > \delta^{\langle\rangle}_{stuv}(i) \text{ and } \delta^{[\,]}_{stuv}(i) > \delta^{0}_{stuv}(i) \\ \langle\rangle & \text{if } \delta^{\langle\rangle}_{stuv}(i) > \delta^{[\,]}_{stuv}(i) \text{ and } \delta^{\langle\rangle}_{stuv}(i) > \delta^{0}_{stuv}(i) \\ 0 & \text{otherwise} \end{cases} \tag{16.26}$$

where

$$\delta^{[\,]}_{stuv}(i) = \max_{\substack{1\le j\le N \\ 1\le k\le N \\ s\le S\le t \\ u\le U\le v \\ (S-s)(t-S)+(U-u)(v-U)\neq 0}} a_{i\to[jk]}\, \delta_{sSuU}(j)\, \delta_{StUv}(k) \tag{16.27}$$

$$\begin{bmatrix} \iota^{[\,]}_{stuv}(i) \\ \kappa^{[\,]}_{stuv}(i) \\ \sigma^{[\,]}_{stuv}(i) \\ \upsilon^{[\,]}_{stuv}(i) \end{bmatrix} = \operatorname*{argmax}_{\substack{1\le j\le N \\ 1\le k\le N \\ s\le S\le t \\ u\le U\le v \\ (S-s)(t-S)+(U-u)(v-U)\neq 0}} a_{i\to[jk]}\, \delta_{sSuU}(j)\, \delta_{StUv}(k) \tag{16.28}$$

$$\delta^{\langle\rangle}_{stuv}(i) = \max_{\substack{1\le j\le N \\ 1\le k\le N \\ s\le S\le t \\ u\le U\le v \\ (S-s)(t-S)+(U-u)(v-U)\neq 0}} a_{i\to\langle jk\rangle}\, \delta_{sSUv}(j)\, \delta_{StuU}(k) \tag{16.29}$$

$$\begin{bmatrix} \iota^{\langle\rangle}_{stuv}(i) \\ \kappa^{\langle\rangle}_{stuv}(i) \\ \sigma^{\langle\rangle}_{stuv}(i) \\ \upsilon^{\langle\rangle}_{stuv}(i) \end{bmatrix} = \operatorname*{argmax}_{\substack{1\le j\le N \\ 1\le k\le N \\ s\le S\le t \\ u\le U\le v \\ (S-s)(t-S)+(U-u)(v-U)\neq 0}} a_{i\to\langle jk\rangle}\, \delta_{sSUv}(j)\, \delta_{StuU}(k) \tag{16.30}$$

3. Reconstruction

Initialize by setting the root of the parse tree to $q_1 = (0, T, 0, V)$ and its nonterminal label to $\ell(q_1) = \mathrm{S}$. The remaining descendants in the optimal parse tree are then given recursively for any $q = (s, t, u, v)$ by

$$\textsc{left}(q) = \begin{cases} \textsc{nil} & \text{if } t-s+v-u \le 2 \\ \left(s, \sigma^{[\,]}_q(\ell(q)), u, \upsilon^{[\,]}_q(\ell(q))\right) & \text{if } \theta_q(\ell(q)) = [\,] \text{ and } t-s+v-u > 2 \\ \left(s, \sigma^{\langle\rangle}_q(\ell(q)), \upsilon^{\langle\rangle}_q(\ell(q)), v\right) & \text{if } \theta_q(\ell(q)) = \langle\rangle \text{ and } t-s+v-u > 2 \\ \textsc{nil} & \text{otherwise} \end{cases} \tag{16.31}$$

$$\textsc{right}(q) = \begin{cases} \textsc{nil} & \text{if } t-s+v-u \le 2 \\ \left(\sigma^{[\,]}_q(\ell(q)), t, \upsilon^{[\,]}_q(\ell(q)), v\right) & \text{if } \theta_q(\ell(q)) = [\,] \text{ and } t-s+v-u > 2 \\ \left(\sigma^{\langle\rangle}_q(\ell(q)), t, u, \upsilon^{\langle\rangle}_q(\ell(q))\right) & \text{if } \theta_q(\ell(q)) = \langle\rangle \text{ and } t-s+v-u > 2 \\ \textsc{nil} & \text{otherwise} \end{cases} \tag{16.32}$$

$$\ell(\text{LEFT}(q)) = \iota_q^{\theta_q(\ell(q))}(\ell(q)) \tag{16.33}$$

$$\ell(\text{RIGHT}(q)) = \kappa_q^{\theta_q(\ell(q))}(\ell(q)) \tag{16.34}$$

The time complexity of this algorithm in the general case is $\Theta(N^3T^3V^3)$ where N is the number of distinct nonterminals and T and V are the lengths of the two sentences. (Compare this to the case monolingual parsing, which is faster by a factor of V^3.) The complexity is acceptable for corpus analysis that does not require real-time parsing.

16.6.5 Grammars for Biparsing

Biparsing techniques may be used without a specific grammar, with a coarse grammar, or with detailed grammars.

No language-specific grammar (the BITG technique).

In the minimal case, no language-specific grammar is used (this is particularly useful when grammars do not exist for both languages). Instead, a generic *bracketing inversion transduction grammar* or BITG is used:

A	$\overset{a_{[]}}{\rightarrow}$	[A A]	
A	$\overset{a_{\langle\rangle}}{\rightarrow}$	$\langle$A A$\rangle$	
A	$\overset{b_{ij}}{\rightarrow}$	e_i/f_j	for all i,j lexical translation pairs
A	$\overset{b_{i\epsilon}}{\rightarrow}$	e_i/ϵ	for all i language-0 vocabulary
A	$\overset{b_{\epsilon j}}{\rightarrow}$	ϵ/f_j	for all j language-1 vocabulary

This grammar is sufficient to cover the full range of word-order transpositions that can be generated under any ITG (because the 2-normal form implies that rules with rank > 2 are not needed). The unlabeled tree in Figure 16.1f is an example of a BITG biparse tree.

All probability parameters may be estimated by EM (Wu 1995d). However, in practice alignment performance is not very sensitive to the exact probabilities, and rough estimates are adequate. The b_{ij} distribution can be estimated through simpler EM-based bilexicon learning methods, as discussed later. For the two singleton rules, which permit any word in either sentence to be unmatched, a small ϵ-constant can be chosen for the probabilities $b_{i\epsilon}$ and $b_{\epsilon j}$, so that the optimal bracketing resorts to these rules only when it is otherwise impossible to match the singletons. Similarly, the parameters $a_{[]}$ and $a_{\langle\rangle}$ can be chosen to be very small relative to the b_{ij} probabilities of lexical translation pairs. The result is that the maximum-likelihood parser selects the parse tree that best meets the combined lexical translation preferences, as expressed by the b_{ij} probabilities.

BITG biparsing can be seen as being similar in spirit to *word_align*, but without positional offsets. The maximum probability word alignment is chosen, but with little or no penalty for crossed couplings, as long as they are consistent with constituent structure (even if the coupled words have large positional offsets). The assumption is that the language-universal core-arguments hypothesis modeled by ITGs is a good constraint on the space of alignments allowed. The *word_align* bias toward preferring the same word order in both languages can be imitated to a large extent by choosing $a_{\langle\rangle}$ to be slightly smaller than $a_{[]}$ and thereby giving prefering the straight orientation.

Empirically, the BITG technique is fairly sensitive to the bilexicon accuracy and coverage. This is due to the fact that a missed word coupling can adversely affect many couplings of its

dominating constituents. Also, in practice, additional heuristics are useful to compensate for ambiguities that arise from the underconstrained nature of a BITG. (The extreme case is where both sides of a sentence pair have the same word order; in this case there is no evidence for any bracketing.) A number of heuristics are discussed in Wu (1997).

Coarse grammar.

Performance over the BITG technique may often be improved by introducing a small number of rules to capture frequent constituent patterns. This can be done by writing a very small grammar either for one of the languages (Wu 1995d) or for both languages simultaneously. The generic bracketing rules should still be retained, to handle all words that do not fit the constituent patterns. The labeled tree in Figure 16.1f is an example of a biparse tree with a linguistic grammar.

Detailed grammar.

A detailed monolingual grammar for one of the languages can be augmented to convert it into an ITG. A simple but effective heuristic for doing this simply mirrors each rule into straight and inverted versions (Wu and Wong 1998). This biases the constituents that will be aligned to fit the selected language, at the expense of degraded parsing of the other language. Again, the labeled tree in Figure 16.1f is an example of a biparse tree with a linguistic grammar.

16.6.6 Strengths and Weaknesses of Biparsing and ITG Tree Alignment Techniques

Biparsing models have a stronger theoretical basis for selecting alignments, as they are clearly formulated with respect to a generative bilingual language model. This permits probabilistic trading-off between the amount of information in the monolingual parses versus the lexical correspondences.

Biparsing techniques may be used with pre-bracketed bitext, just as with parse-parse-match tree alignment techniques, by including the brackets as *a priori* constraints in the dynamic programming search (Wu 1997). Performance in this case is similar, except that the ordering constraints are slightly stronger with ITGs. Otherwise, the exhaustive dynamic programming search is more reliable than the heuristic search used by tree alignment methods.

The BITG technique can be used to produce a rough bracketing of bitext where no other parsers or grammars are available. It is the only approach under circumstances where no grammar is available for one or both of the languages (thereby precluding the possibility of preparsing the sides of the bitext). Clearly it is the weakest of all the structure/tree/biparsing alignment methods in terms of parsing accuracy. The flip side is that the BITG technique infers new bracketing hypotheses, which can be used for grammar induction. In fact, Zhang and Gildea (2004) show that unsupervised BITG alignments empirically produce significantly better AER scores than a supervised tree-to-string model that depends on the output of a monolingual parser, due to the "impedance mismatch" between the syntactic structure of Chinese and English.

Similarly, if a grammar is available for only one of the languages, a biparsing approach can use just the single grammar whereas parse-parse-match techniques do not apply. The biparsing technique is also useful if two grammars are available but use very different constituent structures (as mentioned earlier, structure/tree alignment methods may be unable to come up with good constituent couplings under these circumstances).

One issue with (non-BITG) biparsing techniques is that it can be difficult and time-consuming to write a single grammar that parses both sides well. It is relatively easy to parse one side well, by using a grammar that fits one of the languages. However, the more language-specific details in the grammar, the more nonterminals it requires to keep the rules in sync with both languages. For this reason, it is important to retain backup generic rules for robustness.

To avoid the manual construction of transduction grammars, a great deal of recent research has focused on automatic methods for inducing transduction grammars—generally ITGs of one form or

another, and occasionally SDTGs.* One popular approach is the *hierarchical phrase-based translation* method of Chiang (2005), which learns a highly lexicalized ITG in a binary-rank normal form with either straight rules like $A \rightarrow a_0 A_1 a_2 A_3 a_4 :: a_0 A_1 a_2 A_3 a_4$ (or simply $A \rightarrow [a_0 A_1 a_2 A_3 a_4]$) or inverted rules like $A \rightarrow a_0 A_1 a_2 A_3 a_4 :: a_4 A_3 a_2 A_1 a_0$ (or simply $A \rightarrow \langle a_0 A_1 a_2 A_3 a_4 \rangle$), where a_i is any sequence of lexical translations x/y or singletons x/ϵ or ϵ/y, and there is only one undifferentiated nonterminal category A (as in BITGs). Various other lexicalized and headed variants of ITGs—including dependency models—have also been shown to improve SMT accuracy (Alshawi et al. 2000; Cherry and Lin 2003; Zhang and Gildea 2005).

Neither biparsing, nor structure/tree alignment techniques that work on surface constituency structures, can accommodate the general case of "second-order transformations" such as raising, topicalization, wh-movement, and gapping. Such transformations can cause the surface constituency structure to lose its isomorphism with the "deep" structure's frames and core arguments. No consistent set of couplings is then possible between the surface tree structures of the two languages. For these phenomena, working on the feature-structure tree instead of the constituency tree may hold more hope.

Nevertheless, a large body of empirical research has shown that inversion transductions appear optimal for the vast majority of statistical MT models (e.g., Lu et al. 2001; Lu et al. 2002; Simard and Langlais 2003; Zhao and Vogel 2003). Zens and Ney (2003) show a significantly higher percentage of word alignments are covered under BITG constraints than under IBM constraints, for German–English (96.5% vs. 91.0%), English–German (96.9% vs. 88.1%), French–English (96.1% vs. 87.1%), and English–French (95.6% vs. 87.6%). Wu et al. (2006) find similar results for Arabic–English (96.2%) and English–Arabic (97.0%).

Furthermore, a comparison of IBM versus (Bracketing) ITG constraints on Japanese–English translation by Zens et al. (2004) found significantly higher MT accuracy measured via BLEU, NIST, WER, and PER scores.

The excellent fit of BITG reordering permutations to naturally occurring sentence translations has led to the development of automated MT evaluation metrics that correlate well with human judgment, such as invWER (Leusch et al. 2003) based on interpreting ITG constraints as a compositional block generalization of Levenshtein string edit distance, and its successor, CDER (Leusch et al. 2006).

The biparsing algorithms for ITG alignment have moreover been successfully applied to numerous other tasks including paraphrasing and textual entailment (Wu 2006) and mining parallel sentences from very nonparallel comparable corpora (Wu and Fung 2005).

For languages without explicit word boundaries, particularly Asian languages, the biparsing techniques are particularly well suited. With segmentation integrated into the biparsing algorithm, the word segmentation of the text can be chosen in tandem with choosing the bracketing and couplings, thereby selecting a segmentation that optimally fits the alignment (Wu 1995c).

16.7 Conclusion

The alignment of matching passages can be performed at various granularities: document-structure, sentence, word, or constituent. A variety of techniques are available for each granularity, with tradeoffs in speed and memory requirements, accuracy, robustness to noisy bitext, ease of implementation, and suitability for unrelated and non-alphabetic languages. Sources of leverage include lexical translations, cognates, end constraints, passage lengths, syntactic parses, and assumptions about monotonicity, approximate monotonicity, and word order.

New approaches to alignment in SMT are continually being developed. Recent perspectives on the tremendous amount of research on syntax and tree-structured models can be seen, for example, in the "Syntax and Structure in Statistical Translation" (SSST) series of workshops (Wu and Chiang 2007; Chiang

* Recall that any SDTG (or synchronous CFG) that is binary rank, ternary rank, or only allows straight/inverted permutations is an ITG; and any SDTG that is right-regular or left-regular is an FSTG.

and Wu 2008; Wu and Chiang 2009). Heuristic association-based models are easier to implement than generative models, and can provide reasonable performance (Melamed 2000; Moore 2005a). Recently, discriminative training methods have been applied to word alignment (Callison-Burch et al. 2004; Moore 2005b; Liu et al. 2005; Klein and Taskar 2005; Fraser and Marcu 2007; Lambert et al. 2007; Ma et al. 2009); one potential advantage is to directly train model parameters to maximize some translation accuracy objective function such as BLEU (Papineni et al. 2002), rather than measures such as *alignment error rate* or *AER* (Och and Ney 2003). Empirically, AER does not necessarily correlate with translation quality (Ayan and Dorr 2006); merely improving AER can reduce translation accuracy (Liang et al. 2006) and vice versa (Vilar et al. 2006).

Alignment models are the core of all modern SMT and EBMT models. They are also tremendously useful in supporting a broad range of functionality in tools such as translators' and lexicographers' workstations, as is well surveyed in the volume edited by Veronis (2000) and the seminal ARCADE project on evaluating alignment (Veronis and Langlais 2000). Increasingly today, alignment algorithms that are faster, more accurate, and more robust are being used for automatic and semi-automatic resource acquisition, especially the extraction of phrasal translation lexicons as well as compositional and hierarchical tree-structured translation examples.

Acknowledgments

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under GALE Contract No. HR0011-06-C-0023, and by the Hong Kong Research Grants Council (RGC) research grants RGC6083/99E, RGC6256/00E, and DAG03/04.EG09. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency.

References

Aho, Alfred V. and Jeffrey D. Ullman (1969a). Properties of syntax-directed translations. *Journal of Computer and System Sciences 3*(3), 319–334.

Aho, Alfred V. and Jeffrey D. Ullman (1969b). Syntex-directed translations and the pushdown assembler. *Journal of Computer and System Sciences 3*(1), 37–56.

Aho, Alfred V. and Jeffrey D. Ullman (1972). *The Theory of Parsing, Translation, and Compiling (Volumes 1 and 2)*. Englewood Cliffs, NJ: Prentice-Hall.

Alshawi, Hiyan, Shona Douglas, and Srinivas Bangalore (2000, Mar). Learning dependency translation models as collections of finite-state head transducers. *Computational Linguistics 26*(1), 45–60.

Ayan, Necip Fazil and Bonnie J. Dorr (2006, Jul). Going beyond AER: An extensive analysis of word alignments and their impact on MT. In *21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL'06)*, Sydney, Australia, pp. 9–16.

Bellman, Richard (1957). *Dynamic Programming*. Princeton, NJ: Princeton University Press.

Bresnan, Joan (Ed.) (1982). *The Mental Representation of Grammatical Relations*. Cambridge, MA: MIT Press.

Brown, Peter F., Jennifer C. Lai, and Robert L. Mercer (1991). Aligning sentences in parallel corpora. In *29th Annual Meeting of the Association for Computational Linguistics (ACL-91)*, Berkeley, CA, pp. 169–176.

Brown, Peter F., John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, Robert L. Mercer, and Paul S. Roossin (1988, Aug). A statistical approach to language translation. In *12th International Conference on Computational Linguistics (COLING-88)*, Budapest, Hungary, pp. 71–76.

Brown, Peter F., John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin (1990, Jun). A statistical approach to machine translation. *Computational Linguistics 16*(2), 79–85.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer (1993, Jun). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics 19*(2), 263–311.

Callison-Burch, Chris, David Talbot, and Miles Osborne (2004, Jul). Statistical machine translation with word- and sentences-aligned parallel corpora. In *42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, Barcelona, Spain, pp. 176–183.

Catizone, Roberta, Graham Russell, and Susan Warwick (1989, Aug). Deriving translation data from bilingual texts. In Uri Zernik (Ed.), *First Lexical Acquisition Workshop*, Detroit, MI.

Chang, Chao-Huang and Cheng-Der Chen (1993, Jun). HMM-based part-of-speech tagging for Chinese corpora. In *Workshop on Very Large Corpora (WVLC)*, Columbus, OH, pp. 40–47.

Chen, Stanley F. (1993). Aligning sentences in bilingual corpora using lexical information. In *31st Annual Meeting of the Association for Computational Linguistics (ACL-93)*, Columbus, OH, pp. 9–16.

Cherry, Colin and Dekang Lin (2003, Aug). A probability model to improve word alignment. In *41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, Sapporo, Japan, pp. 88–95.

Chiang, David (2005, Jun). A hierarchical phrase-based model for statistical machine translation. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, Ann Arbor, MI, pp. 263–270.

Chiang, David and Dekai Wu (Eds.) (2008, Jun). In *Proceedings of SSST-2, Second Workshop on Syntax and Structure in Statistical Translation, at ACL-08:HLT*. Columbus, OH: Association for Computational Linguistics.

Chiang, Tung-Hui, Jing-Shin Chang, Ming-Yu Lin, and Keh-Yih Su (1992). Statistical models for word segmentation and unknown resolution. In *ROCLING-92*, Taipei, Taiwan, pp. 121–146.

Chomsky, Noam (1957). *Syntactic Structures*. The Hague, the Netherlands/Paris, France: Mouton.

Church, Kenneth Ward and Eduard H. Hovy (1993). Good applications for crummy machine translation. *Machine Translation 8*, 239–358.

Church, Kenneth Ward, Ido Dagan, William Gale, Pascale Fung, J. Helfman, and B. Satish (1993). Aligning parallel texts: Do methods developed for English-French generalize to Asian languages? In *Pacific Asia Conference on Formal and Computational Linguistics*, Taipei, Taiwan, pp. 1–12.

Cover, Thomas M. and Joy A. Thomas (1991). *Elements of Information Theory*. New York: Wiley.

Dagan, Ido, Kenneth Ward Church, and William A. Gale (1993, June). Robust bilingual word alignment for machine aided translation. In *Proceedings of the Workshop on Very Large Corpora*, Columbus, OH, pp. 1–8.

Fraser, Alexander and Daniel Marcu (2007, Sep). Measuring word alignment quality for statistical machine translation. *Computational Linguistics 33*(3), 293–303.

Gale, William A. and Kenneth Ward Church (1991a). A program for aligning sentences in bilingual corpora. In *29th Annual Meeting of the Association for Computational Linguistics (ACL-91)*, Berkley, CA, pp. 177–184.

Gale, William A. and Kenneth Ward Church (1991b, Feb). A program for aligning sentences in bilingual corpora. Technical Report 94, AT&T Bell Laboratories, Statistical Research, Murray Hill, NJ.

Gale, William A. and Kenneth Ward Church (1993, Mar). A program for aligning sentences in bilingual corpora. *Computational Linguistics* 19(1), 75–102.

Grishman, Ralph (1994, Aug). Iterative alignment of syntactic structures for a bilingual corpus. In *Second Annual Workshop on Very Large Corpora (WVLC-2)*, Kyoto, Japan, pp. 57–68.

Guo, Jin (1997, Dec). Critical tokenization and its properties. *Computational Linguistics 23*(4), 569–596.

Haruno, Masahiko and Takefumi Yamazaki (1996, Jun). High-performance bilingual text alignment using statistical and dictionary information. In *34th Annual Conference of the Association for Computational Linguistics (ACL-96)*, Santa Cruz, CA, pp. 131–138.

Kaji, Hiroyuki, Yuuko Kida, and Yasutsugu Morimoto (1992, Aug). Learning translation templates from bilingual text. In *14th International Conference on Computational Linguistics (COLING-92)*, Nantes, France, pp. 672–678.

Karlgren, Hans, Jussi Karlgren, Magnus Nordström, Paul Pettersson, and Bengt Wahrolén (1994, Aug). Dilemma–an instant lexicographer. In *15th International Conference on Computational Linguistics (COLING-94)*, Kyoto, Japan, pp. 82–84.

Kasami, T. (1965). An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA.

Kay, Martin and M. Röscheisen (1988). Text-translation alignment. Technical Report P90-00143, Xerox Palo Alto Research Center, Palo Alto, CA.

Kay, Martin and M. Röscheisen (1993). Text-translation alignment. *Computational Linguistics 19*(1), 121–142.

Klavans, Judith and Evelyne Tzoukermann (1990, Aug). The BICORD system. In *13th International Conference on Computational Linguistics (COLING-90)*, Helsinki, Finland, Volume 3, pp. 174–179.

Klein, Dan and Ben Taskar (2005, Jun). Max-margin methods for nlp: Estimation, structure, and applications. In *Tutorial at ACL-05*, Ann Arbor, MI.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu (2003, May). Statistical phrase-based translation. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2003), Companion Volume*, Edmonton, Canada, pp. 48–54.

Lambert, Patrik, Rafael E. Banchs, and Josep M. Crego (2007, Apr). Discriminative alignment training without annotated data for machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007)*, Rochester, NY, pp. 85–88.

Leusch, Gregor, Nicola Ueffing, and Hermann Ney (2003, Sep). A novel string-to-string distance measure with applications to machine translation evaluation. In *Machine Translation Summit IX (MT Summit IX)*, New Orleans, LA.

Leusch, Gregor, Nicola Ueffing, and Hermann Ney (2006, Apr). Cder: Efficient mt evaluation using block movements. In *EACL-2006 (11th Conference of the European Chapter of the Association for Computational Linguistics)*, Trento, Italy, pp. 241–248.

Lewis, Philip M. and Richard E. Stearns (1968). Syntax-directed transduction. *Journal of the Association for Computing Machinery 15*(3), 465–488.

Liang, Percy, Ben Taskar, and Dan Klein (2006). Alignment by agreement. In *Human Language Technology Conference of the North American chapter of the Association for Computational Linguistics (HLT-NAACL 2006)*, New York, pp. 104–111.

Lin, Yi-Chung, Tung-Hui Chiang, and Keh-Yih Su (1992). Discrimination oriented probabilistic tagging. In *Proceedings of the ROCLING-92*, Taipei, Taiwan, pp. 85–96.

Lin, Ming-Yu, Tung-Hui Chiang, and Keh-Yih Su (1993). A preliminary study on unknown word problem in Chinese word segmentation. In *Proceedings of the ROCLING-93*, Taipei, Taiwan, pp. 119–141.

Liu, Yang, Qun Liu, and Shouxun Lin (2005, Jun). Log-linear models for word alignment. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, Ann Arbor, MI, pp. 459–466.

Lu, Yajuan, Sheng Li, Tiejun Zhao, and Muyun Yang (2002, Aug). Learning chinese bracketing knowledge based on a bilingual language model. In *17th International Conference on Computational Linguistics (COLING-02)*, Taipei, Taiwan.

Lu, Yajuan, Ming Zhou, Sheng Li, Changning Huang, and Tiejun Zhao (2001, Feb). Automatic translation template acquisition based on bilingual structure alignment. *Computational Linguistics and Chinese Language Processing 6*(1), 83–108.

Ma, Yanjun, Patrik Lambert, and Andy Way (2009, Jun). Tuning syntactically enhanced word alignment for statistical machine translation. In *13th Annual Conference of the European Association for Machine Translation (EAMT 2009)*, Barcelona, Spain, pp. 250–257.

Macklovitch, Elliott (1994, Oct). Using bi-textual alignment for translation validation:the transcheck system. In *First Conference of the Association for Machine Translation in the Americas (AMTA-94)*, Columbia, MD, pp. 157–168.

Macklovitch, Elliott and Marie-Louise Hannan (1996). Line 'em up: Advances in alignment technology and their impact on translation support tools. In *Second Conference of the Association for Machine Translation in the Americas (AMTA-96)*, Montreal, Canada, pp. 145–156.

Matsumoto, Yuji, Hiroyuki Ishimoto, and Takehito Utsuro (1993, Jun). Structural matching of parallel texts. In *31st Annual Meeting of the Association for Computational Linguistics (ACL-93)*, Columbus, OH, pp. 23–30.

Melamed, I. Dan (1997, Jul). A word-to-word model of translational equivalence. In *35th Annual Meeting of the Association for Computational Linguistics (ACL-97)*, Madrid, Spain, pp. 104–111.

Melamed, I. Dan (2000, Jun). Models of translational equivalence. *Computational Linguistics 26*(2), 221–249.

Moore, Robert C. (2005a). Association-based bilingual word alignment. In *Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, Ann Arbor, MI, pp. 1–8.

Moore, Robert C. (2005b, Oct). A discriminative framework for bilingual word alignment. In *Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, Vancouver, Canada, pp. 81–88.

Nagao, Makoto (1984). A framework of a mechanical translation between Japanese and English by analogy principle. In Alick Elithorn and Ranan Banerji (Eds.), *Artificial and Human Intelligence: Edited Review Papers Presented at the International NATO Symposium on Artificial and Human Intelligence*, pp. 173–180. Amsterdam, the Netherlands: North-Holland.

Och, Franz, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev (2004, May). A smorgasbord of features for statistical machine translation. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2004)*, Boston, MA.

Och, Franz Josef and Hermann Ney (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics 29*(1), 19–52.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002, Jul). BLEU: A method for automatic evaluation of machine translations. In *40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, Philadelphia, PA, pp. 311–318.

Sadler, Victor and Ronald Vendelmans (1990). Pilot implementation of a bilingual knowledge bank. In *13th International Conference on Computational Linguistics (COLING-90)*, Helsinki, Finland, Volume 3, pp. 449–451.

Saers, Markus and Dekai Wu (2009). Improving phrase-based translation via word alignments from stochastic inversion transduction grammars. In *Proceedings of SSST-3, Third Workshop on Syntax and Structure in Statistical Translation, at NAACL-HLT, 2009*, Boulder, CO, pp. 28–36.

Simard, Michel, George F. Foster, and Pierre Isabelle (1992). Using cognates to align sentences in bilingual corpora. In *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-92)*, Montreal, Canada, pp. 67–81.

Simard, Michel and Philippe Langlais (2003, May). Statistical translation alignment with compositionality constraints. In *HLT/NAACL-2003 Workshop on Building and Using Parallel Texts*, Edmonton, Canada.

Simard, Michel and Pierre Plamondon (1996, Oct). Bilingual sentence alignment: Balancing robustness and accuracy. In *Second Conference of the Association for Machine Translation in the Americas (AMTA-96)*, Montreal, Canada, pp. 135–144.

Sproat, Richard, Chilin Shih, William A. Gale, and Nancy Chang (1994, Jun). A stochastic word segmentation algorithm for a Mandarin text-to-speech system. In *32nd Annual Conference of the Association for Computational Linguistics (ACL-94)*, Las Cruces, NM, pp. 66–72.

van Rijsbergen, Cornelis Joost (1979). *Information Retrieval* (2nd ed.). Butterworth-Heinemann, Newton, MA.

Veronis, Jean (Ed.) (2000). *Parallel Text Processing: Alignment and Use of Translation Corpora*, Kluwer, Dordrecht, the Netherlands.

Veronis, Jean and Philippe Langlais (2000, Aug). Evaluation of parallel text alignment systems: The ARCADE project. In Jean Veronis (Ed.), *Parallel Text Processing: Alignment and Use of Translation Corpora*. Dordrecht, the Netherlands: Kluwer. ISBN 0-7923-6546-1.

Vilar, David, Maja Popovic, and Hermann Ney (2006, Nov). AER: Do we need to "improve" our alignments? In *International Workshop on Spoken Language Translation (IWSLT 2006)*, Kyoto, Japan, pp. 205–212.

Viterbi, Andrew J. (1967). Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory 13*, 260–269.

Warwick, Susan and Graham Russell (1990). Bilingual concordancing and bilingual lexicography. In *EURALEX-90*, Malaga, Spain.

Wu, Dekai (1994, Jun). Aligning a parallel english-chinese corpus statistically with lexical criteria. In *32nd Annual Meeting of the Association for Computational Linguistics (ACL-94)*, Las Cruces, NM, pp. 80–87.

Wu, Dekai (1995a, Jun). An algorithm for simultaneously bracketing parallel texts by aligning words. In *33rd Annual Meeting of the Association for Computational Linguistics (ACL-95)*, Cambridge, MA, pp. 244–251.

Wu, Dekai (1995b, Jul). Grammarless extraction of phrasal translation examples from parallel texts. In *Sixth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-95)*, Leuven, Belgium, pp. 354–372.

Wu, Dekai (1995c, Aug). Stochastic Inversion Transduction Grammars, with application to segmentation, bracketing, and alignment of parallel corpora. In *14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, pp. 1328–1334.

Wu, Dekai (1995d, Jun). Trainable coarse bilingual grammars for parallel text bracketing. In *Third Annual Workshop on Very Large Corpora (WVLC-3)*, Cambridge, MA, pp. 69–81.

Wu, Dekai (1997, Sep). Stochastic Inversion Transduction Grammars and bilingual parsing of parallel corpora. *Computational Linguistics 23*(3), 377–404.

Wu, Dekai (2006). Textual entailment recognition based on Inversion Transduction Grammars. In Joaquin Quiñonero Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alché Buc (Eds.), *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005*, Southampton, U.K., April 11–13, 2005, *Revised Selected Papers*, Volume 3944 of *Lecture Notes in Computer Science*, Southampton, U.K., pp. 299–308. Springer, Berlin.

Wu, Dekai, Marine Carpuat, and Yihai Shen (2006, Dec). Inversion Transduction Grammar coverage of Arabic-English word alignment for tree-structured statistical machine translation. In *IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*, Aruba.

Wu, Dekai and David Chiang (Eds.) (2007, Apr). *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*. Rochester, New York: Association for Computational Linguistics.

Wu, Dekai and David Chiang (Eds.) (2009, Jun). *Proceedings of SSST-3, Third Workshop on Syntax and Structure in Statistical Translation, at NAACL-HLT 2009*, Boulder, CO: Association for Computational Linguistics.

Wu, Dekai and Pascale Fung (1994, Oct). Improving Chinese tokenization with linguistic filters on statistical lexical acquisition. In *Fourth Conference on Applied Natural Language Processing (ANLP-94)*, Stuttgart, Germany, pp. 180–181.

Wu, Dekai and Pascale Fung (2005, Oct). Inversion Transduction Grammar constraints for mining parallel sentences from quasi-comparable corpora. In *Second International Joint Conference on Natural Language Processing (IJCNLP 2005)*, Jeju, Korea, pp. 257–268.

Wu, Dekai and Hongsing Wong (1998, Aug). Machine translation with a stochastic grammatical channel. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL'98)*, Montreal, Canada.

Wu, Zimin and Gwyneth Tseng (1993). Chinese text segmentation for text retrieval: Achievements and problems. *Journal of The American Society for Information Science 44*(9), 532–542.

Xu, Donghua and Chew Lim Tan (1996, Jun). Automatic alignment of English-Chinese bilingual texts of CNS news. In *International Conference on Chinese Computing (ICCC-96)*, Singapore, pp. 90–97.

Younger, David H. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control 10*(2), 189–208.

Zens, Richard and Hermann Ney (2003, Aug). A comparative study on reordering constraints in statistical machine translation. In *41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, Sapporo, Japan, pp. 192–202.

Zens, Richard, Hermann Ney, Taro Watanabe, and Eiichiro Sumita (2004, Aug). Reordering constraints for phrase-based statistical machine translation. In *20th International Conference on Computational Linguistics (COLING-04)*, Geneva, Switzerland.

Zhang, Hao and Daniel Gildea (2004, Aug). Syntax-based alignment: Supervised or unsupervised? In *20th International Conference on Computational Linguistics (COLING-04)*, Geneva, Switzerland.

Zhang, Hao and Daniel Gildea (2005, Jun). Stochastic lexicalized inversion transduction grammar for alignment. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, Ann Arbor, MI, pp. 475–482.

Zhang, Hao, Liang Huang, Daniel Gildea, and Kevin Knight (2006, Jun). Synchronous binarization for machine translation. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2006)*, New York, pp. 256–263. Association for Computational Linguistics.

Zhao, Bing and Stephan Vogel (2003, May). Word alignment based on bilingual bracketing. In *HLT/NAACL-2003 Workshop on Building and Using Parallel Texts*, Edmonton, Canada.