Hierarchical Translation Equivalence over Word Alignments

Khalil Sima'an*
University of Amsterdam

Gideon Maillette de Buy Wenniger**
University of Amsterdam

We present a theory of word alignments in machine translation (MT) that equips every word alignment with a hierarchical representation with exact semantics defined over the translation equivalence relations known as hierarchical phrase pairs. The hierarchical representation consists of a set of synchronous trees (called Hierarchical Alignment Trees – HATs), each specifying a bilingual compositional build up *for a given word aligned, translation equivalent sentence pair. Every HAT consists of a single tree with nodes decorated with local transducers that conservatively generalize the asymmetric bilingual trees of Inversion Transduction Grammar (ITG). The HAT representation is proven semantically equivalent to the word alignment it represents, and minimal (among the semantically equivalent alternatives) because it densely represents the subsumption order between pairs of (hierarchical) phrase pairs. We present an algorithm that interprets every word alignment as a semantically equivalent set of HATs, and contribute an empirical study concerning the* exact coverage *of subclasses of HATs that are semantically equivalent to subclasses of manual and automatic word alignments.*

1. Introduction

A major challenge for machine translation (MT) research is to systematically define for every source-target sentence pair in a parallel corpus a bilingual recursive structure that shows how the target-language translation of the source sentence is built up from the translations of its parts. The core of this challenge is to align, recursively, the parts that are translation equivalents in every sentence pair in a parallel corpus (Wu 1996, 1997; Wu and Wong 1998). This kind of recursive alignment at the sub-sentential level (in contrast with the word level) is often represented as a pair of source-target trees with alignment links between their nodes. Nodes that are linked together dominate fringes that are considered translation equivalents. Inducing hierarchical alignments in parallel texts (Wu 1997) turns out a far more difficult task than inducing conventional word alignments, i.e., alignments at the lexical level.

Perhaps learning hierarchical alignment is so difficult because it hinges on fundamental knowledge of how translation equivalent units *compose together recursively* into larger units. The (hierarchical) phrase-based SMT models, e.g., (Zens, Och, and Ney 2002; Koehn, Och, and Marcu 2003; Galley et al. 2004; Chiang 2007; Zollmann and Venugopal 2006; Mylonakis and Sima'an 2011), avoid this difficulty by directly extracting rules of translation equivalence (also known as phrase pairs or synchronous productions) from a *word aligned parallel corpus*. The extraction heuristics treat word alignments as constraints that define the set of admissible translation equivalents. For example, the phrase pairs admissible by a given word alignment are non-empty pairs of contiguous sub-strings that are aligned together but not with other positions

* Institute for Logic, Language and Computation, University of Amsterdam. k.simaan@uva.nl
** Institute for Logic, Language and Computation, University of Amsterdam. gemdb@gmail.com

outside, e.g., (Koehn, Och, and Marcu 2003). The hierarchical phrase pairs of Chiang (Chiang 2005, 2007) are defined by a recursive extension of these admissibility constraints. And the GHKM approach (Galley et al. 2004) is directly aimed at reconciling the admissibility constraints over word alignments with the constituency constraints expressed by syntactic structure. In all these cases, word alignment is assumed the starting point for extracting basic translation equivalents, used as the *lexical(ized)* part of a synchronous grammar.

The current state-of-the-art SMT systems employ automatically induced word alignments that are known to be far from perfect. The phrase pair extraction heuristics used by state-of-the-art models seem to compensate for the inaccuracy of word alignments by extracting a *grossly redundant* set of translation equivalents. This redundancy leads to overgeneration but puts the burden of selecting the better translations squarely on the statistical model.

In this paper we concentrate on the question how to represent word alignments in a parallel corpus as *(sets of) synchronous tree pairs* (STPs) that *exactly* capture the (unpruned) set of lexical translation equivalents that are commonly extracted from word alignments. We are motivated primarily by the idea that when such a hierarchical representation is available, future hierarchical translation models need not start out by hypothesizing a synchronous grammar before seeing the word aligned parallel data. Instead, a variety of synchronous grammars can be extracted directly from the hierarchical representation, in analogy to the way monolingual grammars are currently extracted from monolingual treebanks.[1] On the one hand, such a representation provides a formal tool for rigorous analysis of the kinds of synchronous grammars that best fit with the word alignments, and on the other, it replaces the phrase extraction heuristics with a sentence-level hierarchical representation, that facilitates the statistical modeling of *how translation equivalents compose together into larger translation equivalents*.

We present a hierarchical theory of word alignments that equips them with:

- An asymmetric representation of word alignments that extends permutations into a representation (called permutation sets) that accommodates many-to-one, one-to-many and many-to-many alignments.
- A hierarchical representation (called HATs) as a rather limited form of STPs and an algorithm that computes a set of HATs for every permutation set. The semantics[2] of the HATs produced by our algorithm is proven equivalent to the set of lexical translation equivalence relations, known from phrase-based models and Chiang synchronous grammars.

We exploit this theory for an empirical study on manually and automatically word aligned parallel corpora providing statistics over sub-classes of word alignments. We report *coverage figures* for limited forms of HATs and exemplify a possible application contributing novel insights to an ongoing debate on *(how to compute) the alignment coverage of (normal form) ITG*, e.g., (Zens and Ney 2003; Wu 1997; Wellington, Waxmonsky, and Melamed 2006; Huang et al. 2009; Wu, Carpuat, and Shen 2006; Søgaard and Wu 2009).

We will first provide an intuitive outline of the present work and a road map that explains the structure of this paper.

1 Treebank grammars in parsing are extracted from unambiguously manually annotated sentences, whereas here a set of STPs is computed for every word aligned sentence pair. It will be necessary to induce a probability distribution over the different STPs that represent every word alignment in a parallel corpus. The present work is not concerned with inducing such distributions but merely with defining the exact set of STPs.

2 Our use of the word semantics is in the formal sense of the set-theoretic interpretation of a representation.

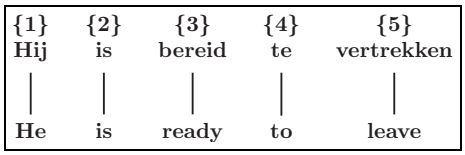
2. An Intuitive Outline: How to Represent Translation Equivalence Recursively?

What is the semantics of word aligned sentence pairs in parallel corpora?. In machine translation, source-target sentence pairs in a parallel corpus are considered *translation equivalents*. Word alignments are interpreted as the lexical relations that delimit the space of *sub-sentential translation equivalence units* (also called translation units) that underly MT models. To define the semantics of word alignments in parallel corpora, we need to define:

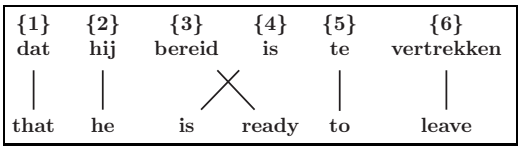
- The minimal translation equivalence relations intended by individual alignment links between words, and
- The translation equivalence relations defined by the different forms of *co-occurrence* of individual alignment links in word alignments of sentence pairs.

Crucially, in this paper we are also interested in defining (for *every* word alignment) a hierarchical representation that details explicitly the *recursive, compositional build up* of translation equivalents from the individual word alignment links up to the sentence-pair level.

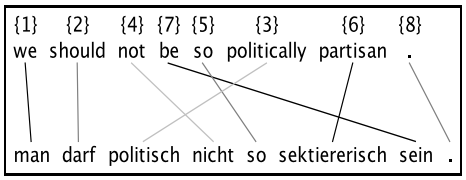
A word alignment defines the *minimal translation equivalence relations* in a sentence pair. Individual words linked together are translation equivalents (TEs), and we interpret multiple words linked with a single word conjunctively, i.e., the linked words together are equivalent to that word. Figures 1a to 1d show example word alignments[3]. In Figure 1d, `worthwhile` is equivalent to `moeite waard` (and we will write this ⟨`worthwhile`, `moeite waard`⟩); neither `moeite` nor `waard` separately is equivalent to `worthwhile`. Unaligned words must group with other aligned words in their surroundings to form possible units of translation equivalence. In Figure 1d, the Dutch word `de` groups with `moeite waard` leading to the TE ⟨`worthwhile`, `de moeite waard`⟩.



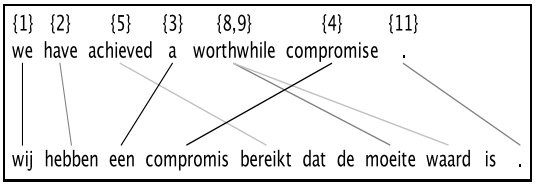
(a) Example monotone alignment (Dutch-English)



(b) Example alignment (Dutch-English)



(c) Alignment 4131 from Europarl EN-DE

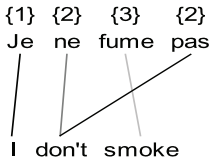


(d) Alignment 6213 from Europarl EN-NE.

Figure 1: Example word alignments of varying complexity

How minimal TE relations combine together into larger units is perhaps a theoretical matter related to the assumption of *compositional translation* (Janssen 1998; Landsbergen 1982). In contrast, data-driven approaches define which target units in parallel data are *likely* to be good translations of which source units, given a word alignment, e.g., (Zens, Och, and Ney 2002;

3 The sets of integers in these figures are of later relevance in this discussion.



A = (I XP == Je XP) *B = (don't XP == ne XP pas)*
C = (smoke == fume) *D = (don't smoke == ne fume pas)*
E = (I don't XP == Je ne XP pas)
F = (I don't smoke == Je ne fume pas)

Figure 2: A word aligned sentence pair and a selection of translation equivalents. XP is a nonterminal variable.

Koehn, Och, and Marcu 2003; Chiang 2007). Statistics are gathered over TE units of varying lengths, including TEs and their sub-units down to the minimal units. In phrase-based models, the non-minimal TE relations (called phrase pairs) are formed by taking contiguous sequences of minimal units equivalent to each other on both sides (under the conjunctive interpretation) and extracting them as larger phrase pair equivalents. In Figure 1b, for example, one may extract ⟨`bereid is`, `is ready`⟩ and ⟨`hij bereid is`, `he is ready`⟩ but cannot extract ⟨`hij is`, `he is`⟩ for the lack of adjacency of ⟨`hij`, `he`⟩ and ⟨`is`, `is`⟩ on the Dutch side (note that the latter does constitute a phrase pair of the alignment in Figure 1a). In Figure 1d, we find ⟨`achieved a worthwhile compromise`, `een compromis bereikt dat de moeite waard is`⟩, where the unaligned words (`dat de`) are included because they are the only material intervening between two TEs.

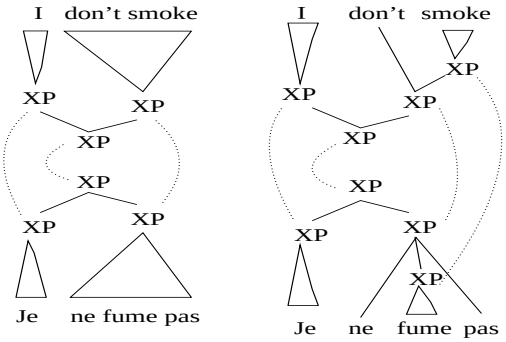
The *extraction method* used in phrase-based models can be seen to compose two or more TE units into a larger TE unit if and only if their components are *adjacent on both source and target sides*[4] (Koehn, Och, and Marcu 2003). Adjacency on both sides can be seen as simple *synchronized concatenation of sequences*, albeit with the possibility of *permuting the order* of the sequences on the one side relative to the other.

Chiang's extraction method (2007) extends the set of phrase pairs with higher order TE relations (synchronous productions) containing pairs of *variables* linked together to stand for TE sub-units that has been abstracted away from a phrase pair. The phrase pair ⟨`hij bereid is`, `he is ready`⟩ in Figure 1b can produce a Chiang-style synchronous rule ⟨`hij X is`, `he is X`⟩, where `X` on both sides stands for two nonterminal variables linked together. Note that the two `X` instances stand in positions where another TE unit ⟨`bereid`, `ready`⟩ used to reside. Figure 2 exhibits one more alignment for a shorter (and well known) sentence pair and a selection of example TEs.

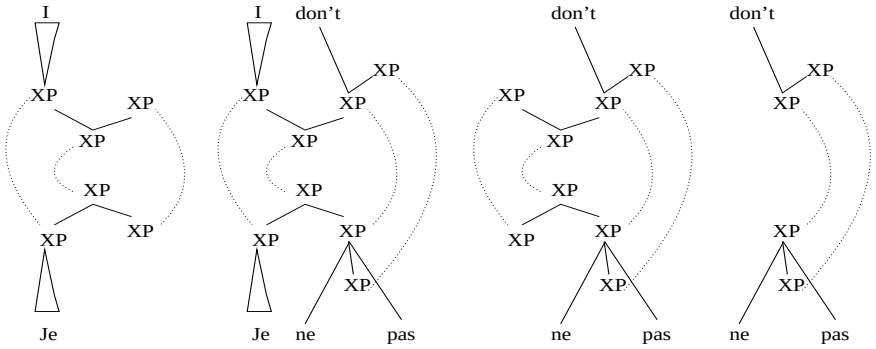
In Section 4 we define the semantics of a sentence-level word alignment to be equivalent to the set of translation equivalence relations that are extracted from it by the Chiang method[5]. With this semantics in place, we are interested in the question how to represent a word alignment in a hierarchical formalism that harbors all and only the translation equivalents that the semantics of word alignments defines, and makes explicit the compositional structure of TEs?

What is the recursive structure of translation equivalence?. By extracting arbitrary length (hierarchical) phrase pairs directly, current extraction methods do not concern themselves with the question how sub-units of TEs compose together to form larger TEs in a word aligned sentence pair in the training data. We prefer a representation that shows as much as possible how a constellation of multiple TE units compose together to form larger *composit* TE units.

4 In other words they must form contiguous spans on both sides.
5 In practical systems like Hiero, the extractions are pruned using various heuristics. These heuristics are not relevant for this study.



(a) Spurious Ambiguity



(b) Synchronous fragments extracted from the right STP in Figure 3a

Figure 3: Example STPs and synchronous fragments

Initially we make the assumption that TEs compose together by *synchronized concatenation with reordering* (thereby enforcing the aforementioned assumption of *bilingual adjacency*). Later on we provide a conservative generalization of this assumption on composition for representing translation equivalence relations that are *discontinuous*, thereby covering the Chiang-style TEs.

Figure 3a exhibits two STPs, pairs of trees with linked pairs of nodes.[6] Edges linking pairs of nodes in the two trees stand for pairs of TEs formed by the sequences of words at the fringes of the subtrees dominated by the two nodes.[7] In both STPs we find the TE ⟨`don't smoke`, `ne fume pas`⟩. In the one STP we find this TE as an atomic unit (the left STP), whereas it is a composit one in the other (the right STP). The two STPs constitute legitimate alternative outcomes of a synchronous grammar like that used in (Chiang 2007; Mylonakis and Sima'an 2011): the atomic version employs a TE (phrase pair production) ⟨`XP`→`don't smoke`, `XP`→`ne fume pas`⟩, whereas the other one employs a derivation consisting of two productions: ⟨`XP`→`smoke`, `XP`→`fume`⟩ substituted in the `XP` linked slots in ⟨`XP`→`don't XP`, `XP`→`ne XP pas`⟩.

6 For the moment being we do not discuss constraints on this general representation and leave that for the formal sections in the sequel.

7 Observe particularly how the French words `ne` and `pas` that are equivalent to `don't` (a discontinuous French side) are not dominated by a pair of linked nodes on their own, and that `ne` and `pas` stand directly under the same mother node `XP` which is linked with the `XP` under which `don't` is also found. This is important for representing the discontinuous TE.

Our proposed representation (called Hierarchical Alignment Trees – HATs) will avoid this kind of *derivational redundancy*. If one TE is a sub-unit of another subsuming TE (e.g., ⟨`smoke`, `fume`⟩ is subsumed by ⟨`don't smoke`, `ne fume pas`)⟩ then the representation developed here will explicitly represent this subsumption order (just like the STP to the right in Figure 3a). We will prove that in the set of HATs that we define for a word alignment, every linked pair of nodes dominates a pair of fringes that constitutes a phrase pair, and that all phrase pairs are represented by linked nodes. *Crucially, the HATs will explicitly represent the subsumption relation between phrase pairs: every phrase pair that can be decomposed into smaller phrase pairs will be represented as such, and the HATs are the STPs that contain the maximal number of nodes that the word alignment permits.*

For building MT models, one could extract *synchronized fragments* (possibly conditioned on context). *Synchronized fragments* can be extracted under the constraint that we "cut" only at linked pairs of nodes (a DOT heuristic due to (Poutsma 2000)). This leads to synchronous fragments that can be used in a Synchronous Tree Substitution Grammar (Eisner 2003) and akin to Chiang's synchronous context-free productions (discarding the heuristic constraints on length etc). Besides the phrase pairs (fully lexicalized fragments) we could also obtain (among others) the synchronous fragments shown in Figure 3b from the right-side STP in Figure 3a. By discarding the internal `XP` nodes in these synchronous fragments we obtain Chiang synchronous context-free productions.

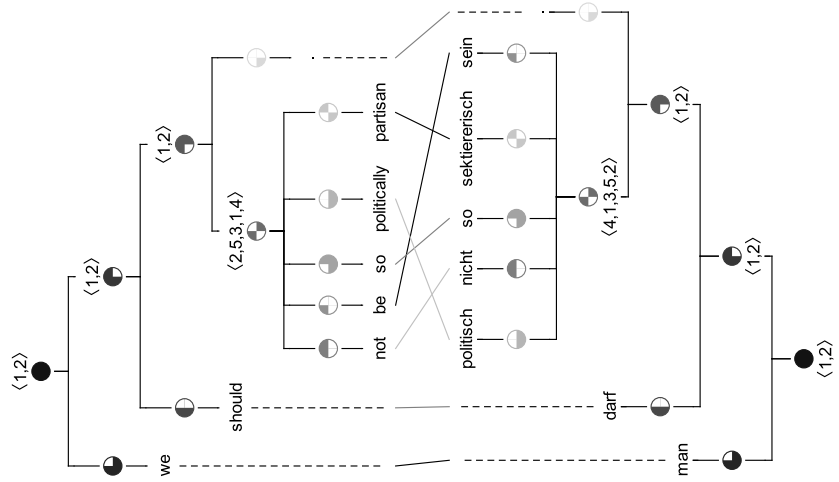
How to build hierarchical structures for translation equivalence?. Consider first the two simple alignments in Figures 1a and 1b. Intuitively speaking, for Figure 1a (monotone), any pair of identical binary-branching trees with linking of all identical (isomorphic) nodes constitutes a suitable STP for this example. The same strategy would work for the second example (Figures 1b) provided that we first group `bereid is` and `is ready` under a pair of linked nodes. Both examples can be dealt with using binary STPs, and in fact these binary STPs are of the kind that can be generated by normal-form ITG. These two alignment are simple examples of what is known as *binarizable permutations* (Huang et al. 2009). In Figures 1a and 1b, the integer permutations are shown as a sequence of singleton sets of integers above the Dutch words; the dual permutation can be formed on the English side by writing down for every English word the position of the Dutch word linked with it.

For developing the hierarchical representation for general word alignments we must address the technical challenges of how to represent complex word order differences and alignments that are not one-to-one. The permutation notation will not work for one-to-many, many-to-one or many-to-many alignments and a special extension is needed. Figure 1d shows how our proposed extended representation looks like: the position of the word `worthwhile` is linked with two Dutch positions and hence is represented by a set of both $\{8, 9\}$. This new extension of permutations to represent general word alignments (and its meaning) is called a *permutation set*. In section 5 we present permutation sets and formalize the counterparts of TEs in this new asymmetric representation.

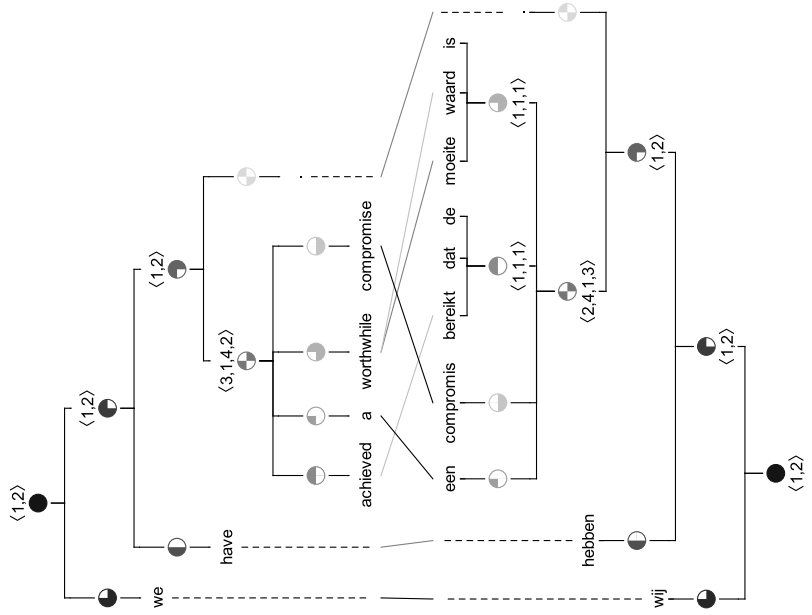
Binarizable permutations constitute a proper subset of the word alignments found in actual data. Figure 1c and 1d show two examples of non-binarizable permutations sets, where the first one is a permutation and the second is a proper permutation set. The word order differences in these word alignments is such that they cannot be represented by an STP generated by an (NF-)ITG. In Figure 1c, the crossing alignments constitute the non-binarizable permutation $\langle 2, 5, 3, 1, 4\rangle$.[8] A similar, but slightly more complex situation holds for Figure 1d because of the

8 To see that it is non-binarizable check that none of the adjacent pairs of integers constitutes a pair of successive integers, i.e., the foreign side positions of the adjacent TEs are not adjacent to one another.

one-to-many alignment: with the unaligned words `dat de` at positions 6 and 7, the permutation set $\langle\{5,6,7\},3,\{8,9\},4\rangle$ is as non-binarizable as $\langle 3,1,4,2\rangle$.



(a) HAT for the word alignment in Figure 1c



(b) HAT for the word alignment in Figure 1d

Figure 4: Two HAT representations for examples of word alignments

For representing word alignments (permutation sets) with STPs, Section 6 develops the HAT representation and algorithms for interpreting word alignments as sets of HATs. We

prove that the HATs have equivalent semantics to word alignments (permutations sets) and are compact/minimal in the sense discussed above. Figure 4 shows two of the HAT representations for the examples in Figures 1c and 1d. To avoid a jungle of node links in such largish examples, we resort to an implicit representation of node linking and a squared off tree representation. Pairs of nodes linked together are represented with the same filled circles (word alignments are left in tact as a visual aid). Figure 4 shows that a minimal branching STP is chosen given the constraints of the word alignment. An idiomatic treatment of a phrase pair that cannot be decomposed under the defined semantics is shown in Figure 4a. Figure 4b shows an interesting case: it represents $\langle$`achieved`, `bereikt dat de`$\rangle$ and $\langle$`worthwhile`, `moeite waard is`$\rangle$ as two pairs of linked nodes. The complex permutation set in this example ($\langle 5, 3, \{8, 9\}, 6, 7, 4\rangle$), discussed earlier, is represented as a four-branching pair of linked nodes (◓). This permutation set is equivalent[9] to $\langle 3, 1, 4, 2\rangle$, an inversion of the famous non-binarizable $\langle 2, 4, 1, 3\rangle$ of (Wu 1997). The nodes on both sides of the STP are decorated with such permutation sets standing for transduction operators that generalize the inversion operator of ITG.[10]

An important property of the HAT representation is that the branching factor at every internal node is *minimal*. In Section 6 we prove that the choice for minimal branching nodes (minimal segmentation of a sub-permutation) leads to the compact representation which avoids the derivational redundancy exemplified above. Because the branching factor of every node is the minimal given the word alignment, then for binarizable permutations we will automatically obtain fully binary HATs (that can be generated by a normal-form ITG).

What kinds of word alignments are being used for building current state-of-the-art SMT systems?. Are the cases of non-binarizable permutations frequent in parallel data or are they marginal cases? What hierarchical properties do these word alignments have? Section 7 provides an empirical study that addresses these and other empirical questions pertaining to subsets of word alignments and the HAT representation. But first, in the next section, we review the related work.

3. Related Work

Given the relative importance of word alignments, the question how to represent them hierarchically has received limited attention. Earlier work makes different modeling assumptions regarding the nature of word alignments, the STP formalism or the role of syntactic trees. Many of these modeling assumptions emanate from the choice for a specific probabilistic synchronous grammar when inducing word alignments. In this work we assume that word alignments are given in parallel data and, therefore, can afford to avoid this assumption.

Wu (Wu 1997) presents a framework for learning hierarchical alignments under Inversion-Transduction Grammar (ITG). The framework starts out by postulating a synchronous grammar for bilingual parsing of sentence pairs in a parallel corpus. By postulating a grammar (ITG) first, the goal is to represent the whole corpus of bilingual sentences as members in the language of this grammar. This contrasts with the goal of the present work: we aim at representing every *individual* word alignment in a parallel corpus with hierarchical bilingual representations (sets of STPs) that are provably equivalent in terms of a *predefined semantic notion of translation equivalence*. Given the definition of translation equivalence over word alignments, one can say

9 After reducing $\langle\{8, 9\}, 6, 7\rangle$ into a single position 6.

10 The complexity of parsing k-branching synchronous grammar with $k > 3$ (rank k syntax-directed transduction grammars (Lewis and Stearns 1968)) is well documented (Satta and Peserico 2005) but irrelevant to this work. Parsing a parallel string depends on the kind of synchronous grammar at hand. In this paper we are neither concerned with extraction nor with parsing under a given grammar but merely with specifying the *exact* representation.

that here we aim at making hierarchical representations explicit that reside in the word alignment itself, *not in an external grammar* specified prior to seeing the alignments. In this sense our goals are different from those of Wu (1997).

Other assumptions regarding representing word alignments are made as well. Typically, when a certain section of a given word alignment cannot be decomposed under the working assumptions regarding the formal grammar, it is extracted as a single atomic phrase pair.[11] This means that translation equivalents that are related to one another in the data (e.g., one embedded in the other) are represented as alternative, unrelated rules. We think that this complicates bilingual parsing as well as the statistical estimation of these grammars (see e.g., (Marcu and Wong 2002; DeNero et al. 2006; Mylonakis and Sima'an 2010, 2011)).

The direct correspondence assumption (DCA) of syntactic parse trees underlies the efforts at building syntactic parallel treebanks, and word alignments are considered merely as heuristic constraints on node-linking, e.g., (Tinsley, Hearne, and Way 2009; Zhechev and Way 2008; Tinsley and Way 2009). This line of research is not concerned with representing the translation equivalence that the input word alignments define but with a node-linking relation between two *monolingual parse trees*.

The ITG (or Wu's) hypothesis states that all (or at least the vast majority of the correct) word alignments (in any parallel corpus) can be represented as pairs of binary trees (STPs) with a bijective node linking relation where "vertically crossing" node alignment are not allowed. Furthermore, for every sequence of sister nodes, the links has only two possible orientations: fully monotone or fully inverted. The stability of the ITG assumption is an empirical matter that depends on the relative frequency of complex permutations and alignment constructions (like Discontinuous Translation Units (Søgaard and Kuhn 2009)) in actual data. Here, we are not concerned with the validity/stability of the ITG assumption in practice. Instead, we are interested in representing word alignments hierarchically to represent our choice for a predefined notion of translation equivalence. Crucially, the resulting hierarchical structure reflects our choice of translation equivalence semantics. Whether a word alignment can be covered by (NF-)ITG or any another formalism is a secondary matter pertaining to various coverage metrics of the translation equivalence relations represented in our hierarchical representation (see also (Søgaard and Kuhn 2009) for a similar observation). We think that the coverage itself can be effectively measured as the intersection between the set of HATs equivalent to the word alignment and the set of synchronous trees generated by the given formalism for the sentence pair. It is crucial to point out the subtle point that the measured coverage is always with regards to a predefined notion of translation equivalence over word alignments and the kind of trees projected from them. We will elaborate more on this observation in Section 7.

The empirical part of this paper (Section 7) explores the hierarchical nature of word alignments within our hierarchical theory. However, as a first example application of our formal and algorithmic findings, we make a modest, yet distinct, contribution to the NF-ITG coverage debate. By defining a shared semantics for word alignments and HATs, our algorithm for computing a set of HATs for every word alignment affords us to report coverage figures based on formal inspection of the set of HATs to determine whether there exists at all an NF-ITG that can generate them. This approach is distinct from earlier approaches to the study of empirical ITG coverage in that it formally builds the HATs for *every* word alignment before doing any measurements. In Section 7, we will contrast our approach with earlier work on NF-ITG coverage.

11 The redundancy of such synchronous grammar rules is reminiscent of Data-Oriented Parsing (DOP) (Bod, Scha, and Sima'an 2003), albeit a major difference is that latter extracts fragments from a treebank implying explicit internal structure shared between different fragments.

4. Defining Translation Equivalence over Alignments

Throughout this paper we will be concerned with *sentences*, finite sequences of tokens (atomic or terminal symbols). The *alignments* will consist of sequences of individual *links* between these tokens. For the purposes of intuitive and simple exposition we will often talk about words but the treatment will apply to atomic tokens at other granularity levels.

Unaligned words (NULLs) lead to an extensive notational burden and, in principle, we will not provide a formal treatment of unaligned words in the more advanced sections, but our intuitive treatment of this special case will aim at showing that an extension of the present techniques to unaligned words is inexpensive.

Definition 1 (Alignment and sub-alignment)
Given a source and target sentence pair, $\mathbf{s} = s_1, \ldots, s_n$ and $\mathbf{t} = t_1, \ldots, t_m$, we define an alignment $\mathbf{a}$ as a relation of pairs consisting of a position in $\mathbf{s}$ and another in $\mathbf{t}$ or NULL, i.e., $\mathbf{a} \subseteq \{0, 1, \ldots, n\} \times \{0, 1, \ldots, m\}$ where position 0 stands for NULL. Each individual pair is a *link*. We will call $\mathbf{b}$ a sub-alignment of an alignment $\mathbf{a}$ when $\mathbf{b} \subseteq \mathbf{a}$.

Alignments in machine translation play a major role in defining the atomic elements of translation equivalence, words linked together or even phrase pairs. We view alignments as postulating basic word-level relations of translation equivalence, that when (somehow) combined together would lead to larger units of translation equivalence, up to the sentence level. The crucial question usually is which links to combine together and which operators to use for the combination. Before we make any choices, we first provide a general definition of relations of translation equivalence defined over an alignment.

Definition 2 (Translation-admissable sub-alignments)
Given an alignment $\mathbf{a}$ between $\mathbf{s}$ and $\mathbf{t}$, a non-empty sub-alignment $\mathbf{b} \subseteq \mathbf{a}$ is *translation-admissable (t-admissable)* iff for every $\langle x, y\rangle \in \mathbf{b}$ holds $\{\langle x1, y1\rangle \in \mathbf{a} \mid (x1 = x) \vee (y1 = y)\} \subseteq \mathbf{b}$. In other words, all alignments involving source position x in $\mathbf{a}$ must either all be in $\mathbf{b}$ or else none, and similarly for all alignments involving target position y.

In Figure 2, since `ne` and `pas` are both linked with `don't`, it is reasonable to think that `don't` translates as `ne + pas`. Hence the definition of t-admissable sub-alignments. The set of all t-admissable sub-alignments of an alignment $\mathbf{a}$, denoted $TA(\mathbf{a})$, is attractive because it defines an important range of translation equivalents (that subsumes phrase pairs). In Figure 2, the sub-alignment representing the word linking $\{\langle$`Je`, `I`$\rangle, \langle$`fume`, `smoke`$\rangle\}$ is t-admissable for this alignment, whilst it is not a phrase pair.

For computational and representational reasons, we will be interested in a subspace of the t-admissable sub-alignments for a given alignment $\mathbf{a}$, particularly the phrase pairs known from phrase-based translation, and phrase-like synchronous productions (containing "holes") as introduced by Chiang (Chiang 2005). Intuitively, for standard phrase pairs, links are grouped together into larger units of translation equivalence if they are adjacent both at the source and target sides.

Definition 3 (Phrase-pair sub-alignment)
A t-admissable sub-alignment $\mathbf{b} \subseteq \mathbf{a}$ is called a *phrase pair sub-alignment* iff both the sets of source and target positions in $\mathbf{b}$ minus the NULLs (position zero), constitute contiguous sequences of source and target positions.

Definition 4 (Minimal phrase-pair sub-alignment)
A phrase pair sub-alignment is minimal if and only if none of its proper sub-aligments is a phrase pair.

Definition 5 (Chiang-admissable sub-alignments)
A t-admissable sub-alignment $\mathbf{b} \subseteq \mathbf{a}$ is called *Chiang-admissable* iff there exists a phrase pair sub-alignment $\mathbf{b}_p \subseteq \mathbf{a}$ such that $\mathbf{b} \subseteq \mathbf{b}_p$ and the complement $(\mathbf{b}_p \setminus \mathbf{b})$ is either empty or constitutes a set of phrase pair sub-alignments.

Clearly, every phrase pair sub-alignment is also Chiang-admissable. However, Chiang-admissable sub-alignments may consist of non-contiguous (on one side or boths sides) sequences of t-admissable sub-alignments that correspond to a phrase pair with "gaps" that stand for phrase pairs. Figure 2 shows a few examples of phrase pairs (translation equivalents *C, D, F*). The same figure shows Chiang-admissable sub-alignments (*A, B, E*) represented with the "holes" between the segments marked with the symbol XP, in following of standard practice.

Having defined the semantics of word alignments as phrase-pair and Chiang-admissable sub-alignments, we will now fix this semantics for all kinds of future representations of word alignments. Our choice for this semantics has attractive properties but this does not come without a price.

Limitations. This semantics conflates the differences between certain word alignments and avoids difficult questions about the semantics of *multi-word, minimal phrase pair sub-alignments* (see Figure 5). Under the *conjunctive interpretation* of word alignments, the contiguous sequences on both sides of a minimal phrase pair sub-alignment belong together. It is then important to recognize that this choice cannot discriminate between different alignments that lead to the same minimal phrase pair. Figure 5 exhibits three word alignments that constitute minimal phrase pairs of the same string pair and all three share the same semantics (set of Chiang-admissable sub-alignments). The topic of extending the semantics such that it discriminates between some of these cases is not treated in this paper.

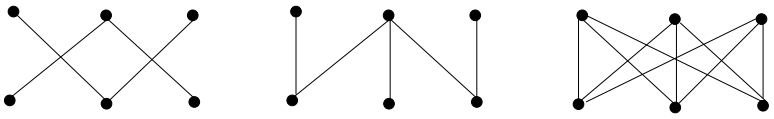


Figure 5: Three word alignments that constitute minimal phrase pairs

5. Alignments as Permutation Sets: A Representation of Relative Word Order

Alignments make explicit various phenomena at the lexical level, in particular word-order differences. Crossing links between positions as well as alignments that relate groups of source words to groups of target words express constraints on how a source and target sentences relate to one another as sentential tranlsation equivalents. The challenge of modeling word-order differences is a major reason for studying syntactic and hierarchical models, e.g., (Wu 1997; Chiang 2007).

In the preceding section we represented t-admissable sub-alignments, phrase pairs or Chiang-admissable, as sets of sub-alignments. The adjacency of links on both sides symultanuosly turned out a crucial constraint on grouping links into phrase pair sub-alignments. Following (Wu 1997; Huang et al. 2009), we choose a simple mechanism to represent how the target-side of a given sub-alignment is obtained from the source side: permutation of positions.

Permutations are useful representations for a subclass of alignments that naturally capture the adjacency requirement. In this section, we propose a new representation of alignments, called *permutation sets*, that extends permutations. We also discuss how translation equivalence follows from permutations and how it relates to translation equivalence definitions from the preceding section, particularly the phrase pairs and Chiang sub-alignments.

Bijective alignments and Permutations. A special interesting case of alignments is the class of *bijective* (i.e. 1:1 and onto) alignments. If $\mathbf{a}$ is bijective, then the positions on the one side can be described as a *permutation* of the positions on the other side. For example, if $\mathbf{a} = \{1\text{-}2, 2\text{-}1, 3\text{-}3, 4\text{-}4\}$ (with source positions coming first in the pairs), then the target positions constitute the permutation $\langle 2, 1, 3, 4\rangle$ relative to source positions.

Definition 6 (Permutations and shifted-permutations)
A permutation π over the range of integers $[1..n]$ is a sequence of integers such that each integer in $[1..n]$ occurs *exactly once* in π. We will also consider permutations over integers in $[i..j]$ (for $i \neq 1$) and refer to them as *shifted-permutations*.

Definition 7 (Sub-permutation)
A sub-permutation π_x of a permutation π is a contiguous subsequence of π which constitutes a shifted-permutation.

Sub-permutations clearly comply with the adjacency requirement of linked positions on both sides, which are required for phrase pair sub-alignments. The following rather straightforward lemma highlights the fact that bijective alignments and permutations can be used to define exactly the same sets of phrase pair sub-alignments (translation equivalence relations):

Lemma 1
The set of phrase pair sub-alignments for a bijective alignment (permutation) $\mathbf{a}$ is equivalent to the set of sub-permutations of the permutation corresponding to $\mathbf{a}$.

The proof of this lemma follows from the definitions of phrase pair sub-alignment for bijective permutations and the definition of sub-permutation.

Notation. For traversing a permutation π from left to right we will employ indeces to mark the current state of traversal: at start the *index* is zero and after moving one *position* to the right the index increments by one (i.e., index $j > 0$ stands between positions j and $j + 1$). The notation $\pi_{<j}$ and $\pi_{>j}$ refers to sub-sequences of π that are repectively the prefix ending with the integer at position j and the suffix starting with the integer at position $j + 1$. Note that these subsequences are not necessarily sub-permutations of π since they might consist of integers that do not define a range of successive integers. For example, in $\pi = \langle 2, 1, 3, 4\rangle$, we find sub-permutations $\pi_{<3} = \langle 2, 1, 3\rangle$ and $\pi_{>3} = \langle 4\rangle$, but $\pi_{>1} = \langle 1, 3, 4\rangle$ is not a sub-permutation.

We can also represent a permutation, e.g., $\langle 2, 1, 3, 4\rangle$, as ranging over singleton sets, i.e., $\langle\{2\}, \{1\}, \{3\}, \{4\}\rangle$. This allows us to encode non-bijective alignments, containing sub-alignments that are not 1:1, as extensions of permutations over sets of target positions relative to source positions. The sets of target positions imply grouping constraints defined by alignments that are not 1:1.

Back to our running example (Figure 2). If we take French as the source language then the representation (called permutation set) of this alignment is $\langle\{1\}, \{2\}, \{3\}, \{2\}\rangle$, whereas if we take English as the source side the *permutation set* should be $\langle\{1\}, \{2, 4\}, \{3\}\rangle$. To arrive at these permutations sets, simply scan the source (respectively target) side left to right word-by-

word and for every word position note down in a set notation the target positions linked with that word. In the representation $\langle\{1\},\{2\},\{3\},\{2\}\rangle$ (the French-side view) the set $\{2\}$ appears in the second and fourth positions signifying that the 2^{nd} English word is linked with both the second and fourth French positions. In other words, the two appearances of $\{2\}$ signify a grouping constraint for the second and fourth French positions with surrounding positions. This example should also highlight the *asymmetric* nature of this representation (just like permutations).

Now we define permutation sets by providing a recipe for obtaining them from alignments. To avoid notational complications, our definition in the single case of unlinked words (i.e., linked with NULL) remains somewhat informal.

Definition 8 (Permutation sets for non-bijective alignments)
A *permutation set* is a finite sequence of finite sets of integers such that the union of these sets constitutes a range of integers $[0..n]$. The three cases of non-bijective alignments are represented in permutation sets as follows:

- For every source position $i_s > 0$ (i.e., not NULL), we represent this position with a set $\mathbf{a}(i_s)$ that fulfills $j_t \in \mathbf{a}(i_s)$ iff i_s and j_t are linked together and none is NULL, i.e., $j_t \in \mathbf{a}(i_s)$ iff $\langle i_s, j_t\rangle \in \mathbf{a}$ and $j_t \neq 0$.
- For contiguous spans of source positions linked with NULL we will group them with the directly adjacent source positions that are linked with target words. For every contiguous span of (one or more) unlinked source words, any prefix of this span may group with aligned source words directly to its left, and the remaining suffix of the span will group with the aligned source words directly to its right. Either the prefix or the suffix can be the empty string but not both. This leads to multiple alternative permutation sets that together represent the same alignment. This conforms with current practice in phrase pair extraction, e.g., (Koehn, Och, and Marcu 2003).
- For contiguous spans of target position linked with NULL, we will first group the positions with the non-NULL linked adjacent positions and then represent the target sets of positions that correspond to every source position. The grouping of the prefix and the suffix of such contiguous spans proceeds analoguously to the NULL-aligned source positions in the preceding item.

If we put NULL-links aside and view alignments asymmetrically (say from the source side), we find that every permutation set represents a single alignment and that every alignment (viewed from the source side) can be represented by a single permutation set. The NULL cases lead to multiple permutation sets that correspond to one and the same alignment. This only leads to more notation and in the sequel we will not deal with NULL links, knowing that they can be treated with a relatively straightforward extension of the present techniques.

Another example of a permutation set is $\langle\{1,2\},\{3\},\{2,4\}\rangle$ which implies the alignment $\{1-1, 1-2, 2-3, 3-2, 3-4\}$ (where here we informally represent an alignment as a set of linked source-target pairs of positions $x-y$).

Definition 9 (Sub-permutation of a permutation set)
A sub-permutation of a permutation set $\pi = \langle s_1, \ldots, s_m\rangle$ is a *contiguous* subsequence $\langle s_i, \ldots s_j\rangle$ ($i \leq j$) that fulfills the requirement that the union $(s_i \cup \ldots \cup s_j)$ of the sets $s_i, \ldots s_j$ constitutes a contiguous range of integers and for every integer $x \in (s_i \cup \ldots \cup s_j)$ holds $x \notin (s_1, \cup \ldots \cup s_{i-1} \cup s_{j+1} \cup \ldots \cup s_m)$.

This definition demands contiguous chunks on both sides and that links from the same position (including discontinuous cases) must remain together. For example, for permutation set $\langle\{1\},\{2\},\{3\},\{2\}\rangle$ (Figure 2, French as source), the atomic subsequences $\langle\{1\}\rangle$ and $\langle\{3\}\rangle$ are sub-permutations. In contrast, the atomic subsequence $\langle\{2\}\rangle$ is not a sub-permutation because one copy of the 2 remains in the complement (the two copies together stand for a discontinuous alignment with English position 2). The subsequence $\langle\{2\},\{3\},\{2\}\rangle$ constitutes a sub-permutation, whereas $\langle\{1\},\{2\}\rangle$ is not a sub-permutation because again one copy of the 2 remains in the complement.

Partial sets. In a permutation set, we refer to those sets that do not constitute an atomic sub-permutation (like $\{2\}$ in $\langle\{1\},\{2\},\{3\},\{2\}\rangle$ or $\{2,5\}$ in $\langle 3,\{2,5\},1,4\rangle$) with the term *partial sets*. A partial set either shares positions with other partial sets or it consists of a non-singleton set that does not constitute a sub-permutation.[12]

In permutation set $\langle\{1,2\},\{3\},\{2,4\}\rangle$ for example, we find that each of $\{1,2\}$ and $\{2,4\}$ separately are partial sets, whereas $\{3\}$ is an (atomic) sub-permutation.

Also under the latter definition we find that the set of phrase pair sub-alignments is equivalent to the set of sub-permutations as stated in the following lemma, which applies to alignments not containing NULL links but can be extended to the general case also.

Lemma 2
The set of phrase pair sub-alignments for an alignment (permutation set) $\mathbf{a}$ is equivalent to the set of sub-permutations of the permutation sets corresponding to $\mathbf{a}$.

We will also define the following intuitively simple partial order relation over sub-permutations of the same permutation π:

Definition 10 (Partial order $<$ over permutation sets)
Given a permutation set π_1 over $[i..j]$ and another permutation set π_2 over $[k..l]$. We will write $\pi_1 < \pi_2$ iff $i \leq \mathbf{j} < \mathbf{k} \leq l$. This relation extends naturally to sub-permutations also.

In summary, permutation sets are asymmetric representations of alignments. In terms of translation equivalence relations, it is important to highlight the equivalence of the set of phrase pair sub-alignments defined by a given alignment to the set of sub-permutations of the corresponding permutation set. This equivalence implies that we can represent alignments hierarchically if we succeed to represent permutation sets hierarchically. As we shall see, because permutation sets are asymmetric they constitute a nice intermediate representation on the way from alignments to hierarchical representations.

6. The Hierarchical Structure of Sub-permutations: Recursive Translation Equivalence

The various kinds of t-admissable sub-alignments from Section 4 stand for sets of translation equivalence relations that can be extracted given an alignment. A permutation set provides an asymmetric representation of the alignment in terms of order differences of the target sentence

12 Partial sets are contiguous source-side sub-units in what is known as Discontinuous Translation Units (DTUs). These correpond to the two cases of a source side position aligned with a discontinuous set of positions on the target side or a target side position aligned with a discontinuous set of positions on the source side. (Søgaard and Kuhn 2009).

relative to the source sentence. By concentrating on sub-permutations of a given permutation set, we concentrate attention on sub-alignments that link consecutive positions on both sides, also known as phrase pair sub-alignments (Section 4).

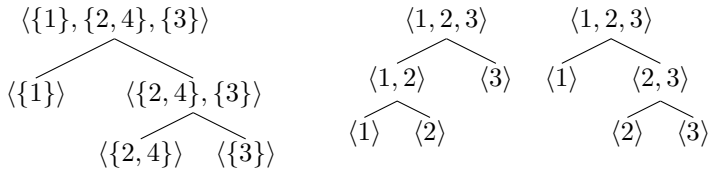


Figure 6: The left tree representation makes explicit the hierarchical structure of the sub-permutations of permutation set $\langle\{1\},\{2,4\},\{3\}\rangle$, and the two to the right for permutation $\langle 1,2,3\rangle$ (since all sets are singletons we simplified the permutation set into a standard permutation). Note that there are two trees for the latter permutation, each showing the recursive grouping using different sub-permutations.

Consider the permutation set $\langle\{1\},\{2,4\},\{3\}\rangle$ for the English side as source in Figure 2. Its sub-permutations are: $\langle\{1\}\rangle$, $\langle\{3\}\rangle$, $\langle\{2,4\},\{3\}\rangle$ and $\langle\{1\},\{2,4\},\{3\}\rangle$. The sequences $\langle\{2,4\}\rangle$ and $\langle\{1\},\{2,4\}\rangle$, for example, are not sub-permutations because $\{2,4\}$ does not constitute a range of consecutive integers.

Let us now concentrate on structuring the sub-permutations of a given permutation set. Figure 6 (left side) shows a graphical representation of how the sub-permutation $\langle\{1\},\{2,4\},\{3\}\rangle$ can be seen as the *concatenation* of the two sub-permutations $\langle\{1\}\rangle$ and $\langle\{2,4\},\{3\}\rangle$, and that the latter sub-permutation is the concatenation of $\langle\{2,4\}\rangle$ and $\langle\{3\}\rangle$, in this order. The same figure also shows two trees for the permutation (set) $\langle 1,2,3\rangle$. Note how these two trees exhibit the grouping of different sub-permutations, that correspond to different sub-alignments, and hence also different translation equivalence relations.

In this section we are interested, on the one hand, in this kind of grouping of sub-permutations into hierarchical structures (trees), and on the other, in a suitable represention that makes explicit the mapping between the local order of source-side groups and the order of their target-side counterparts. The challenge is how to make explicit the hierarchical structure of how sub-permutations compose together, recursively, into larger sub-permutations. Concatenation is the main composition operation that we are going to assume here.

Definition 11 (Concatenation of sub-permutations and/or sequence of partial sets)
The concatenation of sub-permutations is a special case of concatenation of sequences (ordered sets) because sub-permutations are sequences of sets of integers. The same applies to sequences of partial sets. The result of the concatenation of an ordered pair of sequences $\langle\pi_1,\pi_2\rangle$, written $concat(\pi_1,\pi_2)$ is the sequence of sets of integers obtained by concatenating the sequence π_2 after the sequence π_1. We define the concatenation operator to be left-associative.

Note that concatenation itself is not guaranteed to lead to a sub-permutation even if both components are sub-permutations. We will also define the segmentation of a (sub-)permutation (almost but not exactly the inverse of concatenation because concatenation is not guaranteed to result in a sub-permutation).

Definition 12 (Segmentation of a sub-permutation)
A segmentation of a sub-permutation $\pi=\langle k_1,\ldots,k_n\rangle$ is a set of indices $B=\{j_0=0,j_1,\ldots,j_m=n\}$ that segments π into m adjacent, non-overlapping and contiguous subsequences (called segments) such that for all $0\leq i<m$ holds: the sub-sequence of π given by

$k_{j_i}^{j_{i+1}} = k_{j_i} \ldots k_{j_{i+1}}$ is either a *sub-permutation* of π or a sequence consisting of a single *partial set* from π.

For example, the sub-permutation (showing the indices explicitly using extra subscript notation) $\pi_A = \langle_0\{1\}_{,1}\{2\}_{,2}\{3\}_{,3}\{2\}_4\rangle$ has a possible segmentation $B_1 = \{0, 1, 4\}$ leading to sub-permutations $\langle\{1\}\rangle$ and $\langle\{2\}, \{3\}, \{2\}\rangle$, another segmentation $B_2 = \{0, 1, 2, 3, 4\}$ leading to a sub-permutations $\langle\{1\}\rangle$, $\langle\{3\}\rangle$ and twice the sequence with a single partial set $\langle\{2\}\rangle$. The set of indices $B^* = \{0, 2, 4\}$, for example, does not constitute a segmentation of π_A.

A second example might make the segmentations even clearer: for $\pi_B = \langle_0\{1\}_{,1}\{2\}_{,2}\{2\}_{,3}\{3\}_4\rangle$ there are segmentations $B_1 = \{0, 1, 2, 3, 4\}$, $B_2 = \{0, 1, 3, 4\}$, $B_3 = \{0, 1, 4\}$ and $B_4 = \{0, 3, 4\}$ (note that $\langle\{2\}, \{2\}\rangle$ is a sub-permutation of π_B).

Segmentations and the hierarchical structure of sub-permutations. The following lemma (with two sub-statements) is central for devising an algorithm for the hierarchical representation of sub-permutations in permutation sets. Intuitively, the two sub-statements in this lemma together imply that we can build recursive tree hierarchies that work with *minimal segmentations* of π and still cover *all sub-permutations*. This is the intuition behind the algorithms presented in the next Section.

Lemma 3 (Sub-permutations and minimal segmentations)
The lemma has two subsections:

Seg1: For every sub-permutation π_x of another sub-permutation π there exists a segmentation of π into a sequence of segments $\langle A_1, \ldots, A_m\rangle$ in which π_x is a member, i.e., there exists $1 \leq i \leq m$ such that $A_i = \pi_x$.

Seg2: Let $k > 1$ be the minimal cardinality of a segmentation of a sub-permutation π. We will refer to B with $|B| = k$ as a minimal segmentation. For every segmentation B of π, there exists a segmentation B_{min} of π such that $B_{min} \subseteq B$ and $|B_{min}| = k$. In other words, every segmentation B can be regrouped into a minimal segmentation.

Proof

Seg1. By contradiction. Let us assume there is no such segmentation of π. If $\pi_x \neq \pi$ is a sub-permutation of π found between positions indexed with i and j, then there exists X_l and X_r, at least one of which is non-empty, such that $\pi = \langle X_l,\ _i\ \pi_x,\ _j\ X_r\rangle$ (the subscripts $_i$ and $_j$ are used to mark the indeces). If π_x is not a member in any segmentation of π, then (by Definition 12) for every segmentation B of π holds $\{i, j\} \not\subset B$. Because the sub-sequence between i and j is a sub-permutation π_x, this implies that either one or both of the sub-sequences X_l and X_r cannot be segmented into sub-permutations and single partial sets. But because $Concat(X_l, \pi_x, X_r) = \pi$ is a sub-permutation, it is a sequence of sets over a range of consecutive integers $[n_l..n_i..n_j..n_r]$, where π_x is defined over the *proper sub-range* $[n_i..n_j]$. Hence, the integer sets in X_l and X_r must be defined as subsets of $[n_l..n_{i-1}n_{j+1}..n_r]$. But this implies that each such integer set in itself is either a partial set or can form on its own a sub-permutation of π. Contradiction, because this does constitute a segmentation B of π such that $\{i, j\} \in B$.

Seg2. Let π be a sub-permutation with a minimal segmentation of cardinality $|B_{min}| = k$. For every segmentation B of π we want to prove that there exists a segmentation of π called B' such that $B' \subseteq B$ and $|B'| = k$. The proof is by induction on $m = (|B| - k)$. For the case $m = 1$: By contradiction. Suppose there is a segmentation B of π with $|B| = k + 1$ for which

there exists no minimal segmentation $B^{'} \subset B$ and $|B^{'}| = k$. Because B segments π into a sequence $X_1, \ldots, X_{k+1}$ of $k+1$ sub-permutations or partial sets, the assumption holds iff *for all consecutive pairs* X_i and X_{i+1} ($1 \leq i \leq k$) holds: $\langle X_i, X_{i+1} \rangle$ does not constitute a sub-permutation. This situation can occur iff for all $1 \leq i \leq k$ holds the intersection of X_i and/or X_{i+1} with $\cup_{j \notin \{i,i+1\}} X_j$ is non-empty (multiple discontiguous source-side positions aligned with the same target position) and/or, the converse, the set $X_i \cup X_{i+1}$ does not constitute a consecutive range of integers. But if this holds for all consecutive pairs of segments in B, then by the definition of segmentation in every consecutive pair of segments at least one segment is a partial set. In that case, B constitutes the only possible segmentation of π of length $k+1$ because at least every other segment in B is a partial set. Because segmentation works with consecutive segments and all consecutive pairs in B cannot form a sub-permutation, there is no way to segment π into k segments. This contradicts our assumption that π has a minimal segmentation of length $k > 1$. This concludes the case $m = 1$ in the induction.

For segmentations B with $m = (|B| - k) > 1$ we will exploit the standard induction assumption: for every segmentation B_x of π which fulfills $1 \leq |B_x| - k < m$ there exists a segmentation of π called $B^{'}$ such that $B^{'} \subset B_x$ and $|B^{'}| = k$. We want to prove that the same holds for every segmentation $|B| = m + k$. By contradiction, let us assume there exists $|B| = m + k$ for which there does not exist a $B^{'} \subset B$ and $|B^{'}| = k$. This can be true iff we cannot reduce B in any way to a segmentation $B_1 \subset B$ such that $k \leq |B_1| < (m + k)$. For if we could find such a segmentation B_1, the induction assumption will apply; in other words, B can be reduced into B_1 first, and by the induction assumption there exists $B^{'} \subset B_1 \subset B$ and $|B^{'}| = k$. But if there exists no segmentation $B_1 \subset B$ such that $k \leq |B_1| < (m + k)$ then *none* of the sequences x of adjacent segments in B of length $2 \leq |x| \leq (m + 1)$ can be reduced into a sub-permutation. Moreover, none of the adjacent segments of length $|x| > (m + 1)$ can be reduced into a sub-permutation because that would contradict the main assumption and the minimality of k for π. But that implies that B segments π in such a way that none of the sequences of segments in B of length $2 \leq |x| < |\pi|$ can be reduced into a sub-permutation. The latter implies that every sequence of length $2 \leq |x| < |\pi|$ must consist of at least one partial set that has its complement outside the sequence in π. But that implies that B and thus also π is a sequence of partial sets each having its complement at a distance $|\pi|$, which can be true iff π cannot be segmented at all into a segmentation of length larger than one. This contradicts the existence of a minimal segmentation of π of length $k > 1$. ■

These two lemma's together prove that every sub-permutation π_x of a sub-permutation π can be a segment in a segmentation B of π which is a superset of a minimal segmentation of π.

Next we take a little detour to define node-linked pairs of trees, called Synchronized Tree Pairs (STPs), and a constrained node linking relation akin to that known from ITG (Wu 1997). After defining STPs we return to our main task of representing the structure of permutation sets as hierarchical structures of their sub-permutations and the reordering they imply.

6.1 Synchronized Tree Pairs with Layered Linking

Because the labels in the trees assumed here are irrelvant to the discussion, we may assume that internal nodes in these trees are unlabeled (or labeled with a single symbol called *bracket*) and that leaf nodes are labeled with integers representing positions in sentence pairs (the terminals).

Definition 13 (Synchronized Tree Pairs (STPs))
A Synchronized Tree Pair (STP) $\langle \tau_s, \tau_t, \sim \rangle$ consists of a pair of trees τ_s and τ_t and a node-linking relation $\sim$ which at least links the roots of both trees. Apart from the roots, any other node in τ_s could possibly (but not necessarily) be linked together with nodes in τ_t.

Wu (Wu 1997) defines BITGs that derive binary STPs $\{\langle \tau_s, \tau_t \rangle\}$ each fulfilling strict criteria on the node-linking relation. Graphically speaking, links do not cross "vertically" but may cross "horizonally". We state this as the *Layered linking property*, also defined by Wu (Wu 2010) as the *crossing constraint*.

Definition 14 (Layered linking)
The linking relation $\sim$ between the nodes of τ_s and τ_t is a *correspondence* (left-total and right-total[13]), and for every pair of linked nodes $\langle \mu_s, \mu_t \rangle$, where ($\mu_s$ in τ_s and μ_t in τ_t), the children of μ_s are linked only with the children of μ_t and vice versa.

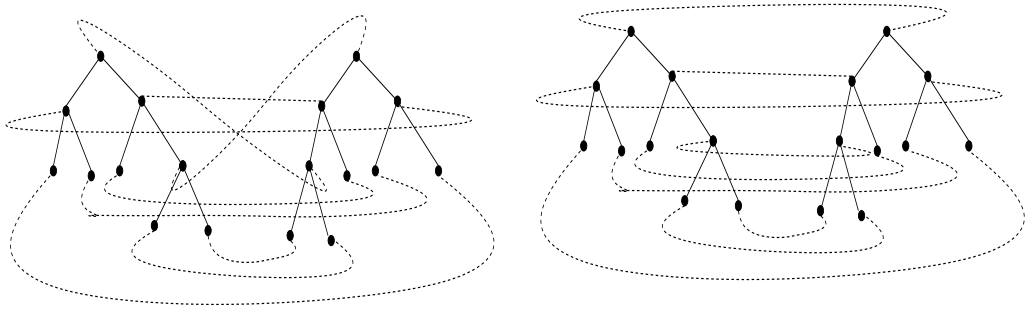


Figure 7: In the left-hand STP the roots are involved in vertically crossing links leading tot non-Layered links, while in the right-hand STP all nodes are involved in layered linking.

Figure 7 shows a schematic example of one layered linking and another non-layered linking (in this abstract figure the links are visualized as edges).

Like many current research that starts out from ITG, e.g., (Wu 1997; Chiang 2007; Zollmann and Venugopal 2006), we also limit the node linking relations to those that abide by *Layered linking*. We do not impose any further constraints. Particularly we do not impose extra constraints on the linking relations of sister nodes.

For permutation sets, the attractive aspects of STP's with layered node linking is that they can be represented as a single source tree decorated at every internal node μ_s with a local transducer that explains how the child-order of the linked target node μ_t is obtained as an order permutation of the child-order of μ_s, plus or minus a handfull extra operations for dealing with many-to-one and one-to-many alignments.

6.2 Representing permutation sets with STPs: Requirements

The algorithms for structuring permutations and permutation sets into layered-linking STPs, presented respectively in Subsections 6.3 and 6.4, take as input a permutation (respectively permutation set) π and output a finite set of STPs. These algorithms abide by three requirements

13 In simple words, every node in τ_s is linked with some node in τ_t and vice versa.

stated in terms of the correspondence between sub-permutations and linked nodes in the STPs built by these algorithms. On the one hand, these requirements provide the intuitive justification for the algorithms, and on the other, they constitute explicit requirements bridging between the world of sub-permutations and the world of linked nodes in STPs.

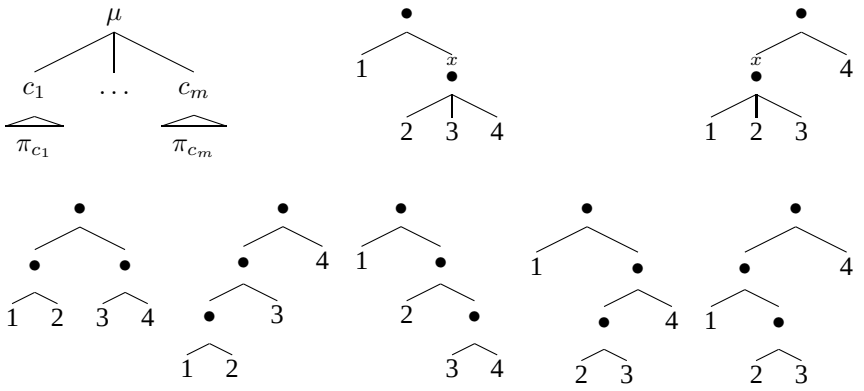


Figure 8: Upper-left corner: a schematic view of decomposing sub-permutations. All other trees show possible hierarchical structuring of permutation $\langle 1, 2, 3, 4\rangle$ into sub-permutations. Nodes labeled with x will not be allowed by our algorithm. These trees constitute one side of the STPs we build for permutations.

I1 (Soundness): Every linked node dominates a subtree with a fringe that corresponds to a sub-permutation. We will say that the linked node dominates that sub-permutation and that the latter is dominated by the node.

I2 (Completeness): For every sub-permutation π_x of input permutation π there will be a linked node that dominates π_x at least in one of the STPs that the algorithm generates for π.

I3 (Maximal hierarchy): Let linked node μ have a sequence of $m > 1$ child nodes $c_1, \ldots c_m$, and denote with $\pi_{c_1}, \ldots \pi_{c_m}$ the sequence of fringes (sub-permutations or partial sets) dominated by $c_1, \ldots c_m$ (See Figure 8 (upper-left corner) for a schematic sketch):

- The sub-permutation π_μ dominated by node μ is equivalent to $concat(\pi_{c_1}, \ldots \pi_{c_m})$.
- m is a minimal segmentation of π_μ.

Requirements (I1) and (I2) together guarantee that there is a one-to-one mapping between the linked nodes in the STPs built by the algorithm and the sub-permutations of the input permutation set. Requirement (I3) guarantees that the tree structure is meaningful and maximal as a hierarchical organization of sub-permutations. The tree structure is compact in that it organizes sub-permutations that concatenate together into larger sub-permutations resulting from the concatenation, and it is maximal because when $m > 1$ is the minimal segmentation of a sub-permutation into sub-permutations, the tree structure will be as deep as possible and contain as

ALGORITHM 1 (CONSTRUCTING PETS FOR PERMUTATIONS: $PeTs(\pi)$)
Input: A (sub-)permutation $\pi = k_1, \ldots k_n$ over $[(l+1)..(l+n)]$ for some $l \geq 0$.
Output: The set of Permutation Trees $PeTs(\pi)$.

$\mathbf{n = 1}$: Set $PeTs(\pi)$ contains a single PeT consisting of a root node directly dominating a leaf node labeled with the only position in π, and the operator on the root node is the identity operator. Return $PeTs(\pi)$

$\mathbf{n > 1}$: **Segment:** Let the set B be a *minimal* set of indices between positions in π such that B segments π into $\underline{m > 1}$ *sub-permutations* $\pi_1, \ldots \pi_m$.

Build:

If there exist such minimal segmentations with $m > 1$ then
For every minimal set of indices B:

1. For each $1 \leq i \leq m$, we build the set $PeTs(\pi_i)$ which is built recursively for sub-permutation π_i.
2. For every combination of PeTs $t_1, \ldots t_m \in PeTs(\pi_1) \times \ldots \times PeTs(\pi_m)$, we build and add to $PeTs(\pi)$ a PeT consisting of a root-node dominating $m = (|B| - 1)$ nodes; each of the resulting child nodes $1 \leq i \leq m$ dominates the PeT t_i for π_i.
3. The root node of the PeT built for B is labeled with a permutation operator $O = \langle o_1, \ldots, o_m \rangle$ over $[1..m]$ such that for every two positions $1 \leq i \neq j \leq m$ holds: $\pi_{o_i} < \pi_{o_j}$ iff $o_i < o_j$.
4. Return $PeTs(\pi)$.

else `#there exist no such segmentations` B `with` $m > 1$`, i.e.` $m = 1$

1. There is no way to segment π and we build a root node of a PeT that dominates n nodes, with the i^{th} node ($1 \leq i \leq n$) dominating the only member of $PeTs(k_i)$.
2. The PeT root node is labeled with the permutation operator $O = \langle o_1, \ldots, o_n \rangle$ over $[1..n]$ such that for every two positions $1 \leq i \neq j \leq m$ holds: $\pi_{o_i} < \pi_{o_j}$ iff $o_i < o_j$.
3. Return $PeTs(\pi)$.

Figure 9: Algorithm $PeTs(\pi)$ outputs the set of PeTs for input permutation π.

many linked nodes as possible (given all other requirements on what the structure is supposed to represent). Figure 8 shows example trees for structuring the simple permutation $\langle 1, 2, 3, 4 \rangle$. The nodes labeled with x do not abide by the minimal segmentation requirement, whereas nodes that are unlabeled do so.

6.3 Permutation Trees: Hierarchies over Sub-permutations of Permutations

We start the effort for representing permutation sets hierarchically by first considering the simpler case of permutations.

A Permutation Tree for a given permutation π over $[1..n]$ is a layered-linking STP consisting of a tree representation over sub-permutations of π and node-links represented as local *permutation operators* decorating every node; a permutation operator on node μ stipulates how to permute the ordered sequence of the children of μ to obtain the sequence of children of the node that μ is linked with. Permutation Trees (PeTs) are inspired by and extend the kind of trees generated by

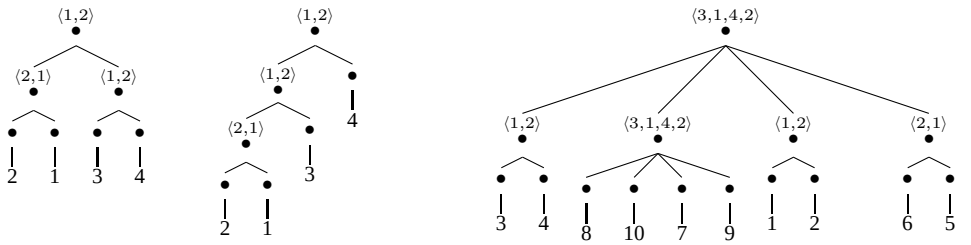


Figure 10: Permutation trees: left two for the same permutation, right the single PeT for the permutation. Note that the identity operators are not marked on pre-terminal nodes but left implicit.

ITGs because they allow arbitrary layered linking relations as opposed to the inverted/monotone binary choice that ITG strictly postulates. For every permutation we will define a unique set of PeTs that fulfills the requirements (I1-I3) from Section 6.2.

Figure 9 shows Algorithm 1 for building the set of PeTs for an input permutation. When π is non-trivial (i.e., $n > 1$), the algorithm builds the PeTs recursively each time by segmenting π into a minimal sequence of sub-permutations and then building sets of PeTs recursively for these sub-permutations. The PeTs for π are obtained by combining PeTs from these sets and putting them under a single node dominating the segmentation.

Figure 10 exhibits the two PeTs for permutation $\langle 1, 2, 3, 4\rangle$, and the single PeT for permutation $\langle 3, 4, 8, 10, 7, 9, 1, 2, 6, 5\rangle$. The latter case is interesting because the minimal segmentation of this permutation consists of four sub-permutations. Note the recursive structure which contains binary branching as well n-ary branching nodes, depending on the minimal segmentations of the sub-permutations. We will return to *binarizable permutations* in the sequel.

Theorem 1 (Soundness, completeness and maximal hierarchy of $PeTs(\pi)$)
The set of PeTs output by Algorithm $PeTs(\pi)$ fulfills the requirements (I1, Soundness), (I2, Completeness) and (I3, Maximal hierarchy) stated in detail in Section 6.2.

Proof

Soundness. By construction: every node created by $PeTs(\pi)$ is built for a sub-permutation. This is easy to check for the case $n = 1$ as well as for the **Build** subsection of $n > 1$.

Completeness. The proof hinges on Lemma 3. We will proceed by induction on the length of π. The cases $n = 1$ and $n = 2$ are trivial and are easy to prove as base for induction. We concentrate on the induction step to length $n > 2$, where we will use the induction assumption for all (sub-)permutations shorter than n. The proof of the induction step is by contradiction. Suppose now there is a sub-permutation π_x of π that is not dominated by any node in a PeT built by $PeTs(\pi)$. Necessarily $\pi_x \neq \pi$ and $n > 1$ because the algorithm must build a root node for every input permutation π, whether within $n = 1$ or whithin **Build** for $n > 1$. There exists a segmentation B of π in which π_x participates (by Lemma 3, part Seg1). Because π_x is assumed not to have a node in any PeT, B cannot be minimal (because if it were minimal then Algorithm $PeTs(\pi)$ would build a node for it at $\mathbf{Build}$). However, if B itself is not minimal, then there is a hierarchical structuring (tree) of B (by concetenation of sub-permutations into new sub-permutations) leading to a minimal segmentation B_{min} of π (by Lemma 3, part Seg2). Consequently, π_x will be covered within one of the sub-permutations called $\pi_{(x)}$ of B_{min}. By the induction assumption

and because $\pi_{(x)}$ must be strictly shorter than π, we find that π_x will have a node built for it within the PeTs for $\pi_{(x)}$. Because B_{min} is a minimal segmentation of π, Algorithm $PeTs(\pi)$ will build a root node dominating a child for every sub-permutation in the sequence defined by B_{min}, i.e. also for $\pi_{(x)}$. Particularly, a copy of the node built for $\pi_{(x)}$ will constitute the root for every PeT in $PeTs(\pi_{(x)})$ (**Build**). Contradiction because now there is a node corresponding for π_x in a PeT built for π.

Maximal hierarchy. This follows from the explicit points in the algorithm (particularly **Segment** and **Build**), Lemma 3 and the above proofs.
■

This theorem states the attractive property of the set $PeTs(\pi)$ that all and only the sub-permutations of π are represented as nodes labeled with permutation operators in the PeTs. Another attractive property is that the minimal branching nodes define maximal trees in number of nodes; becaused nodes dominate sub-permutations, and the latter are translation equivalence relations, we find that PeTs are maximal hierarchies over sub-permutations and can be viewed as hierarchical, extensive explanations of translation equivalence.

Binary Inversion-Transduction Trees (BITTs). Consider for example the permutation $\langle 2,1,3,4\rangle$: both PeTs for this permutation are fully binary branching (see Figure 10). A well-known example of a permutation that does not have binary branching PeTs is $\langle 2,4,1,3\rangle$, due to (Wu 1997). The latter permutation is called a *non-binarizable permutation*. Permutations that have fully binary branching PeTs are called *binarizable* permutations (Huang et al. 2009). Binarizable permutations can be hierarchically structured into PeTs by normal-form ITGs (NF-ITGs), hence we will refer to fully binary branching PeTs with the name *Binary Inversion Transduction Trees*. Interestingly, and due to (Wu 1997), any permutation of length exactly three is binarizable. From this follows that none of the PeTs built by algorithm $PeTs()$ will contain 3-branching nodes. Note that is not necessarily true for the general case of permutation sets (next section).

6.4 Hierarchical Alignment Trees (HATs) for permutation sets

Figure 11 shows Algorithm 2 which builds the set $HATs(\pi)$ for any permutation set π. It generalizes Algorithm 1 and the differences are local. We will refer to the STPs built by Algorithm 2 with the name *Hierarchical Alignment Trees* (HATs), in order to distinguish them from PeTs and BITTs, which are simpler versions of HATs.

Observe in Algorithm 2 an important difference with Algorithm 1 for the case $(n = 1)$: Algorithm 2 discriminates between partial sets (which are represented as leaf nodes) and sub-permutations, which are built as pre-terminal nodes dominating a leaf node. The importance of making this distinction can be understood from the fact that all members of a partial set must remain together (under the same mother node) because none of them separately corresponds to a target-side node. This is in contrast with sub-permutations.

Another difference is that Algorithm 2 defines the node operators as permutation sets. For step **3** under **Build** in Figure 11, the permutation set is initially built with “place holders" (step **a**) that are normalized (step **b**) to get rid of gaps. For example, in the permutation set $\langle 3,\{2,6\},1,4,5\rangle$ (see Figure 14) the algorithm will reduce sub-permutation $\langle 4,5\rangle$ into a single position (step **a**), leading to $\langle 3,\{2,6\},1,4\rangle$ which does not constitute a permutation set (not a contiguous range of integers). In (step **b**), these “place holders" are reduced to a consecutive range of integers: in this example for 6 we find $k_m = 4 \neq (6-1)$ and hence 6 is exchanged with 5, leading to $\langle 3,\{2,5\},1,4\rangle$.

ALGORITHM 2 (CONSTRUCTING HATS FOR PERMUTATION SETS: $HATs(\pi)$)
Input: A sub-permutation / permutation set $\pi = \langle k_1, \ldots k_n \rangle$, where each k_i is an integer set.
Output: The set of HATs $HATs(\pi)$.

n = 1: if (π consists of a single partial set)
then HAT consists of a leaf node labeled π.
else π must contain a sub-permutation
HAT = a node dominating a leaf labeled with π,
and the operator on the node is the identity operator.
Return $HATs(\pi) = \{HAT\}$.

n > 1: Segment: Let $B = \{j_0 = 0, j_1, \ldots, j_m = n\}$ be a *minimal* set of indices between positions in π such that B segments π into $\underline{m > 1}$ *segments*; for all $0 \leq i < m$, π_{i+1} stands for segment $\langle k_{j_i} \ldots k_{j_{i+1}} \rangle$.

Build:
If there exist such minimal segmentations with $m > 1$ then
For every minimal set of indices B:

1 For each $0 \leq i < m$, we build the set $HATs(\pi_{i+1})$ which is built recursively for $\langle \pi_{i+1} \rangle = \langle k_{j_i}^{j_{i+1}} \rangle$.

2 For every combination of HATs $t_1, \ldots t_m \in HATs(\pi_1) \times \ldots \times HATs(\pi_m)$, we build and add to $HATs(\pi)$ a HAT consisting of a root-node dominating $m = (|B| - 1)$ nodes; each of the resulting child nodes $0 \leq i < m$ dominates the HAT t_i for $\langle \pi_{i+1} \rangle = \langle k_{j_i} \ldots k_{j_{i+1}} \rangle$.

3 The root node of the HAT built for B is labeled with a *permutation set* $O = \langle o_1, \ldots, o_m \rangle$ built as follows:

a For $0 \leq i < m$ `# make place holders`
if π_{i+1} is a sub-permutation then $o_{i+1} = \min \cup_{x=j_i}^{j_{i+1}} k_x$
else $o_{i+1} = \pi_{i+1}$ `# for partial sets`

b For $1 \leq i \leq m$ `# reducing the integer gaps`
let $temp = \emptyset$
For all $k \in o_i$ *in increasing order*
let $k_t := \max(\cup_{j=1}^{m} o_j \cap [1..(k-1)])$
$temp$:= $temp \cup \{k_t + 1\}$
$o_i := temp$

4 Return $HATs(\pi)$.

else `#there exist no such segmentations` B `with` $m > 1$, `i.e.` $m = 1$

1 There is no way to segment π and we build a root node of a HAT that dominates n HATs, with the i^{th} HAT being a member of the singleton set $HATs(k_i)$.

2 The HAT root node is labeled with a permutation set $O = \langle o_1, \ldots, o_n \rangle$ equivalent to π.

3 Return $HATs(\pi)$.

Figure 11: Algorithm $HATs(\pi)$ outputs the set of HATs for input permutation set π.

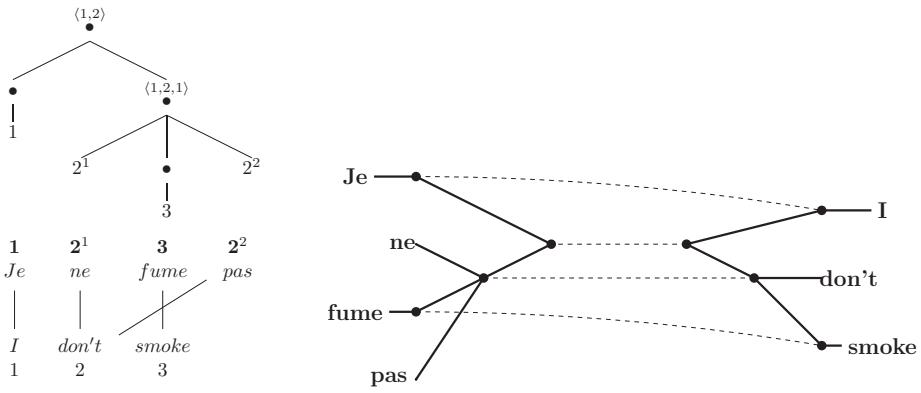


Figure 12: The French construction ne .. pas aligned with don't. The word-aligned sentence pair (lower left corner) is shown with the French side decorated with the permutation set representing English order relative to French order; we use extra notation 2^1 and 2^2 to represent the alignments of both words ne and pas with the second position at the English side. The HAT exhibited for this permutation set is shown in the upper left corner. The notation $\langle 1, 2, 1 \rangle$ on the node dominating this construction stands for (curly brackets ommitted) a permutation set operator linking the first and third children of this node (dominating ne and pas) with the first child under the corresponding node on the English side. The node-aligned trees that the HAT packs together are depicted at the right side of the figure).

Figure 12 contains the running ne ⋆ pas example represented as HAT and unpacked into an STP. Nodes that are represented as • dominate sub-permutation of the permutation set, and they are linked nodes on both sides of the STP. Particularly note how a node dominates fume (inside the ne fume pas) signifying that it constitutes a sub-permutation represented by • and linked with a node dominating smoke. This does not hold for the words ne, pas and don't. They are grouped together only including fume and smoke. Note also that we could extract pairs of linked subtrees from this STP only at the nodes marked with • and thereby extract all sub-permutations, i.e., phrase pairs. If we would allow the subtrees to end at frontier nodes marked •, we would obtain also Chiang-like constructions representing synchronous productions like X→ ⟨ X , X ⟩ where X is a nonterminal variable of the synchronous grammar and the superscript X^1 stands for synchronized nonterminal leaf nodes (linked • nodes in our STPs).

Like Algorithm 1 also Algorithm 2 fulfills the soundness, completeness and maximal hierarchy requirements, i.e., generalizing Theorem 1 from permutations to permutation sets (and the set of phrase pairs to the Chiang's set of synchronous productions). The proofs of this generalization are rather similar (but more detailed) and hinge on Lemma 3 (see Proof of Theorem 1). Importantly, Lemma 3 is stated and proven for *permutation sets in general*, which implies that it applies directly within the proof of the generalization of Theorem 1 to permutation sets (i.e., Algorithm 2). Given the central role of Lemma 3 within the proof, we skip the explicit statement of a corresponding theorem and proof.

Remarks about efficient implementations. An efficient version of Algorithm 2 will avoid recomputation of the same sets of HATs by using a chart or parse-table. A CYK (Younger 1967) bottom-up implementation is relatively easy to devise on the basis of the recursive, top-dwon

Algorithm 2. Given such an efficient chart-based approach, the complexity of Algorithm 2 depends mainly on the complexity of (1) identifying all sub-permutations of a given permutation set and (2) on computing the minimal segmentations for every sub-permutation. Identifying all sub-permutations and partial sets takes $O(n^2)$ time as it demands checking for every span $\langle i, j \rangle$ whether it is a sub-permutation (after an $O(n)$ scan of the input to note down for every integer set whether it is a partial set or not). Having identified all sub-permutations and partial sets, the rest of the algorithm is very similar to standard CYK when a grammar is given, albeit here the grammar is implicit in finding the minimal segmentations of every sub-permutation. To identify the minimal segmentations of a sub-permutation of length m (there are $O(n^2)$ such sub-permutations), Dijkstra's shortest-path algorithm (Dijkstra 1959) can be applied to the graph representing all eligible segmentations of the sub-permutation. The graph consists of m vertices for a sub-permutation of length m and Dijskra's algorithm takes $O(m^2)$ time. Hence, the worst-case complexity of Algorithm 2 is $O(n^4)$.

There is, however, one remaining wrinkle pertaining to the treatment of NULL-aligned words. The phrase pair semantics of translation equivalence dictates that every unaligned word must group with adjacent phrase pairs on either side. When the number of such unaligned words is large, this could lead to a very large number of permutation sets representing the same alignment. Crucially, the structure sharing[14] between the different permutation sets implies that this does not disturb the polynomial-time complexity of the algorithm. Nevertheless, the computation of the full HAT forest can be space-consuming in some extreme cases and for the experiments in Section 7 we will set a cut-off constant on the computation, which rules out only a *negligble* number of alignments in the corpus. With these hints regarding an efficient implementation, we will leave the full formalization of the efficient version of the algorithms for future work dedicated to efficiency issues.

6.5 The role of node operators

The semantics of a node operator that is a permutation set π over the child sequence of the current node is a generalization of the semantics of the ITG operators [] and $\langle\rangle$. The semantics of these operators is the same as the definition of permutation sets for standard alignments but here it applies to the sequence of child nodes of the current node. Algorithmically speaking the semantics of the permutation set π over the children of the current node μ linked with a target side node μ_t is obtained as follows. Scanning π left to right: for the set of integers X in the i^{th} position in π generate child positions under μ_t corresponding to the integers in X and link each of these target positions with the i^{th} child of μ.

Throughout this paper we fixed the semantics of word alignments to be equal to the set of Chiang-admissable sub-alignments (with the phrase pairs being the fully lexicalized bilingually contiguous subset). On the one hand, the HAT representation represents this semantics compactly. On the other, the choice for this semantics circumvents difficult questions about how the semantics of *multi-word, minimal phrase pairs*. A minimal phrase pair is one that cannot be segmented any further because under the *conjunctive interpretation* of word alignments, the contiguous sequences on both sides belong together. Figure 13 exhibits three abstract word alignments that constitute minimal phrase pairs. In the HAT representation each of the three will be represented as a single linked node (pre-terminal level) dominating three terminal nodes. While the tree structure itself is the same, the node operators (permutation sets) on each are different.

14 In analogy to the monolingual case of parsing finite-state automata (lattices or word-graphs) with CFGs (van Noord 1995; Sima'an 1999), the bilingual case here also remains polynomial-time; the time complexity multiplies with a constant factor linear in the number of edges/transitions in the automaton.

The node operators specify the internal alignment structure for, otherwise, hierarchically very difficult to represent word alignments. This way, the internal alignments of phrase pair units remain preserved. As mentioned in Section 4, this difficulty is a theoretical matter regarding the semantics of word alignments.

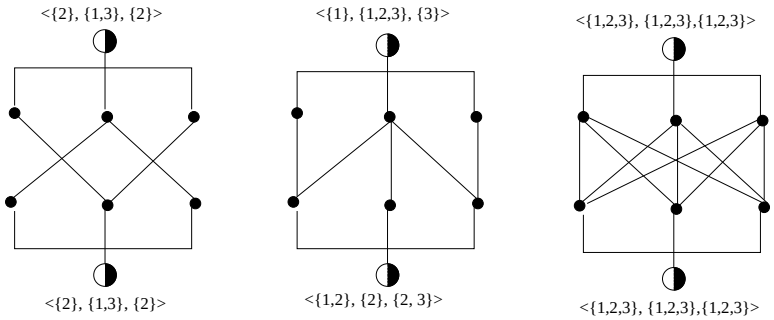


Figure 13: Three word alignments that constitute minimal phrase pairs and their HATs

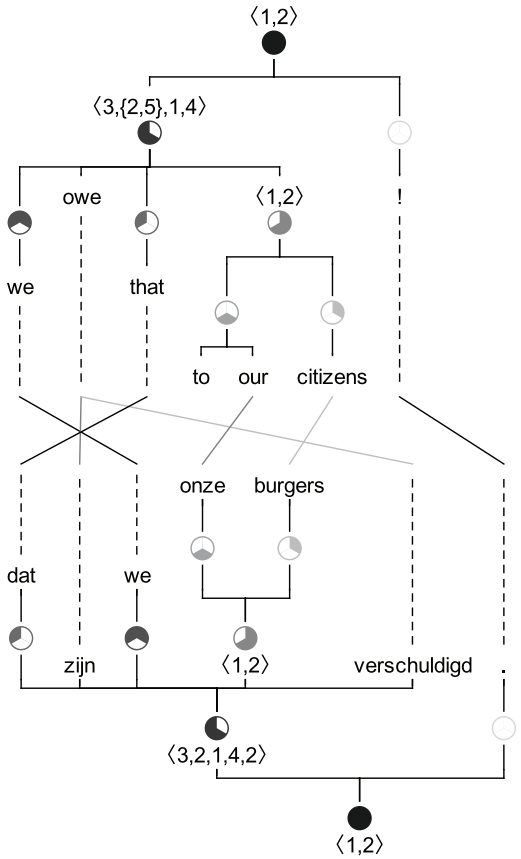


Figure 14: Dual HAT representations (for source and target sides) for a complex word alignment

Figure 14 exhibits an example word alignment (coming straight from our automatically aligned data) and a hierarchical representation consisting of a HAT on one side and its dual HAT on the other side, with links between the pairs of nodes (represented by a choice of circle filling). This word alignment shows an interesting case of a discontinuous alignment of `owe` with `zijn` and `verschuldigd` together with another crossing alignment. The HAT representation *reveals* a pair of linked nodes dominating the synchronous pair $\langle (X_1 \texttt{ owe } X_2\ X_3),\ (X_2 \texttt{ zijn } X_1\ X_3 \texttt{ verschuldigd}) \rangle$, where X_1, X_2 and X_3 stands for three aligned pairs of nodes in the HAT. The latter bilingual construction is a Chiang-admissable sub-alignment. The pair of linked nodes is decorated with a permutation set $\langle 3, \{2, 5\}, 1, 4 \rangle$ on the English side and $\langle 3, 2, 1, 4, 2 \rangle$ on the Dutch side. Observe how these permutation sets actually link the second (`zijn`) and fifth (`verschuldigd`) children of the Dutch node with the second child (`owe`) of the English linked node, thereby maintaining the lexical word aligment for such cases within the HAT structure.

6.6 What are empirical word alignments?

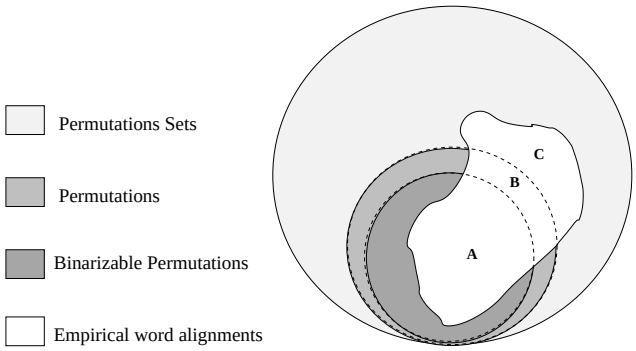


Figure 15: A sketch of a possible characterization of empirical word alignments

Figure 15 shows a sketch of a likely situation: empirical word alignments are not a proper subset of binarizable permutations, nor are they a proper subset of permutations. However, by definition, word alignments are a subset of permutation sets. The figure shows that empirical word alignments overlap with binarizable permutations (area A), with permutations (area A+B) and with permutation sets (area A+B+C). What are the relative sizes of areas A, B and C? Clearly this is an empirical question that depends on the nature of the parallel corpus (language pair and language use) and on the kind of word alignments found in it. In the next section we would like to shed some light on this question. There are various ways for quantifying this question and we would like to cover a few of them using the HAT algorithm.

7. Empirical explorations of the hierarchical characteristics of word alignments

In the preceding sections we showed that every word alignment can be represented by a *semantically equivalent* set of HATs, i.e., given a certain semantic interpretation of word alignments as sets of translation equivalents. With translation equivalence (the semantics) at the center of attention, we are now interested in the statistics of various subsets of HATs that fit well with manual or automatically induced word alignments in parallel corpora. Broadly speaking, the question addressed here is

What percentage of word alignments and translation equivalents can be represented by specifying relevant formal constraints on the HATs (e.g., upperbound on branching factor, upperbound on difference from a permutation)?

This empirical question pertains to statistics over word alignments *in the context of the present choice of semantics (defining the set of translation equivalents)*. The latter observation is important because we think that word alignments do not obtain their semantic interpretation (in terms of translation equivalents) from an external formal grammar. Rather, only after fixing the semantics of word alignments, one can measure the adequacy of a certain (synchronous) grammar/formalism for representing the semantic interpretation of the word alignments.[15]

Next we report various measures over word alignments in corpora. Because HATs are minimally branching STPs for the word alignments, a primary differentiating parameter for the performance measures is the *maximal branching factor* of the nodes in the HATs built for a given word alignment:

β_{max}**:** the *maximal branching factor* of the linked nodes (above the pre-terminal level) in the HATs. The branching factor is measured on both sides and all linked nodes in the HATs for a word alignment must fall within the range $[1..\beta_{max}]$ to be counted in.[16]

For a given β_{max} value we report:

Alignment coverage: This is the percentage of word alignments for which the branching factor of all nodes in all HATs fall within the settings β_{max}. Alignment coverage is not necessarily equivalent to *alignment reachability* (Søgaard and Wu 2009; Søgaard 2010) or the complementary measure of *parsing failure rate (PFR)* (Zens and Ney 2003; Søgaard and Wu 2009), which are both reported under subsets of ITG. Alignment coverage is equivalent to alignment reachability under a given grammar formalism iff both are measured in under the same semantics and such that the setting of β_{max} correspond exactly to the formalism in question.[17] In this sense, alignment coverage for $\beta_{max} = 2$ is an exact (as opposed to upper/lowerbounds) on alignment reachability for NF-ITG.[18]

Translation equivalents coverage (TEC): This is the *percentage of phrase-pairs* that are represented by a linked node in the HATs (if any) that have maximal branching factor β_{max}. The TEC measure is strongly related to Translation Units Error Rate (TUER) (Søgaard and Kuhn 2009; Søgaard and Wu 2009). In fact, if we use the same definition of translation equivalents, the same counting algorithm and the same representation for both (NF-ITG), we find that TUER = (1-TEC).

Binarizability score: The ratio of the number of linked nodes in a HAT (subject to β_{max}) built for a given word alignment relative to the number of such nodes in a hypothetical,

15 Formally, we think that this can be achieved as a measure of the intersection of the set of HATs for a *word alignment* with the set of synchronous trees generated by a synchronous grammar for the *sentence pair*.

16 Null aligned words are not counted in the branching factor, i.e, even unaligned word dominated directly by a node contributes zero to the branching factor of that node. The rationale behind this is that we do not want to discriminate between permutation sets differing only by NULL aligned words. This means that the reported results are rather conservative.

17 Unlike for ITGs, the case $\beta_{max} = 3$ is not necessarily equivalent to $\beta_{max} = 2$ because the permutations sets (as opposed to permutations) over three positions can be non-binarizable, see e.g., Figure 13.

18 Because our algorithm builds HATs over minimal phrase pairs, the case $\beta_{max} = 2$ is equivalent to the NF-ITG over these minimal phrase pairs, i.e., given the defined semantics of word alignments.

fully binary branching HAT (using the length of the shortest among the source and target sentences minus one). The binarizability score, as opposed to TEC, provides a relatively objective measure of how hard it is to represent the word alignments in the corpus by deeply nested HATs. The lower the binarizability score, the less linked nodes exist in the HATs admitted (given β_{max}), if any at all. HATs that contain flat structures indicate complex word alignments consisting of unaligned words and many-to-many alignments that cannot be decomposed into smaller TE units. This can be seen to indicate idiomatic translation, as opposed to compositional translation (given the semantics defined here).

Because for some alignments our algorithm could take a long time to calculate the HAT forest we had to abort the algorithm in a small percentage of cases ($\leq 10^{-5}$), mostly very long, complex alignments with many unaligned words. For all reported experiments we set a cut-off (set at 100k) on the number of inferences per linked node in the CYK chart implementing the HAT algorithm in Figure 11. Once the algorithm exceeds 100k inferences for a node we skip the alignment. In what follows the number of skipped sentence pairs is reported for each experiment separately.

After presenting the empirical results, we will discuss how these results relate to existing work, primarily concentrated on the coverage of (NF-)ITG.

7.1 Manual word alignments

In this part, we use the manually aligned part of the Hansards corpus (English-French), created and first used by Och and Ney (2000, 2003). This tiny corpus consists of 447 manually aligned sentence pairs, were alignment links are labeled with *Sure* or *Possible*. Some basic statistics of this corpus is shown in Table 1. We report the results for the Sure+Possible links, i.e., all alignment links.

Language Pair	**English-French**
Total number of sentence pairs	447
#skipped sentence pairs	0
#sentence pairs containing nulls	231
Mean source length	15.705±6.994
Mean target length	17.362±7.554
Mean and STD of ratio source to target lengths	0.928±0.221
Mean #links per word	2.52 ± 1.73

Table 1: Statistics of Hansards manually aligned corpus. The "mean #links per word" is calculated using the mean $\frac{\text{\# alignment links}}{\min_{x \in \{\mathbf{s},\mathbf{t}\}} length\ of\ x}$ over the corpus word alignments.

Tables 2a, 2b and 2c report respectively the break-down of coverage, TEC and binarizability score to β_{max}. All scores increase rapidly in the range $\beta_{max} \in [2..10]$, but more gradually for $\beta_{max} > 10$. The alignment coverage and the TEC results start low in the seventies for $\beta_{max} = 2$ (NF-ITG), but increase rapidly to the low nineties by $\beta_{max} = 6$. The increase continues at a decaying pace for higher values of β_{max}. The binarizability scores, Table 2c, are a different matter. The results make clear that the HATs built for the word alignments contain only at most 62% of the number of linked nodes in a *hypothetical* fully binary HAT (binarizable permutation). This suggests that these word alignments are represented with HATs that are somewhat flat relative to the hypothetical fully binary HATs. This observation completes the picture drawn by

	English-French
$\beta_{max} = 2$	71.46%
$\beta_{max} = 3$	77.08%
$\beta_{max} = 4$	82.47%
$\beta_{max} = 5$	87.19%
$\beta_{max} = 6$	90.11%
$\beta_{max} = 7$	92.58%
$\beta_{max} = 8$	94.38%
$\beta_{max} = 9$	96.40%
$\beta_{max} = 10$	97.75%
$\beta_{max} = 15$	99.33%
$\beta_{max} = 19$	100.00%

(a) Alignment coverage.

β_{max}	English-French
2	73.54%
3	80.66%
4	87.24%
5	91.74%
6	93.81%
7	95.12%
8	96.96%
9	98.30%
10	98.89%
15	99.81%
19	100.00%

(b) Translation Equivalents Coverage.

β_{max}	English-French
2	42.34%
3	46.97%
4	51.53%
5	55.00%
6	56.82%
7	58.19%
8	59.77%
9	61.02%
10	61.77%
15	62.66%
19	62.91%

(c) Binarizability scores.

Table 2: Scores for Sure+Possible manual alignments in the Hansards corpus as a function of β_{max}.

Kind of HATs (Permutation Sets)	English-French
BITTs (Binarizable permutations)	71.46%
PETs (All permutations)	72.14%
HATs (Permutation Sets)	100.00%

Table 3: The ratio of the different subsets of HATs in the manual Sure+Possible alignments in the Hansards corpus: BITTs, PETs and HATs

the coverage and TEC results for $\beta_{max} > 2$: the word alignments are clearly far more complex than the entry-level cases of binarizable permutations.

At least as interesting, Table 3 reports the coverage of word alignments to the kind of permutation set (HATs) involved: binarizable permutations (BITTs), permutations (PETs) and permutation sets (HATs). Remarkably, merely moving away from binarizable to all permutations does not increase the coverage much, whereas the general case of permutation sets provides full coverage. This result together with the break-down of the statistics to β_{max} values in the other tables suggests actually that the cases of non-binarizable permutations tend to cooccur with other complex forms of alignments, including discontinuous and many-to-many cases. This does not mean that we do not need the full descriptive power of permutations, but that on their own they are *almost as insufficient as their binarizable subset* for capturing word alignments found in actual translation data.

	English-French
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 0$ (pure permutations)	72.04%
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 1$	77.85%
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 4$	85.46%
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 12$	90.16%
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 30$	95.53%
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 50$	98.66%
$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 83$	100.00%

Table 4: Coverage of the Hansard manually aligned corpus as a function of $D(\mathbf{a}, \mathbf{s}, \mathbf{t})$.

To obtain a better picture, we refine Table 3 by ordering *permutation sets* into increasing subsets: *we measure how far a word alignment is from being a permutation*. Formally, a permutation set (word alignment) differs from a permutation in that it is a non-bijective relation, i.e., it may contain many more links than the bijective case (exhaustive one-to-one) and/or the source and target sides may contain a different number of positions. Given our semantics, it makes sense to consider minimal phrase pairs (rather than words) as standing at individual positions. Consequently, we define for every word alignment $\langle \mathbf{s}, \mathbf{t}, \mathbf{a} \rangle$ its "distance" $D(\mathbf{a}, \mathbf{s}, \mathbf{t})$ from being a permutation simply as the absolute value of the difference between the number of individual alignment links it contains ($number\ of\ links(\mathbf{a})$) and the *minimum* number of positions ($length()$) between the source and target sides, where both numbers are measured over the minimal phrase pairs (not the individual words):

$$D(\mathbf{a}, \mathbf{s}, \mathbf{t}) \ = |number\ of\ links(\mathbf{a}) - \min_{x \in \mathbf{s}, \mathbf{t}} length(x)|$$

This measure is a first approximation of the idea, but it provides a meaningful approximate breakdown of the space between word alignments (permutation sets) and permutations. Table 4 shows the breakdown of the coverage of word alignments to any value of $D(\mathbf{a}, \mathbf{s}, \mathbf{t})$ in between the two extreme cases of pure permutations ($D(\mathbf{a}, \mathbf{s}, \mathbf{t}) = 0$) and arbitrary permutation sets that are equivalent to the word alignments themselves. Clearly, approximately one in five of these manual alignments falls somewhat far off from pure permutations ($D(\mathbf{a}, \mathbf{s}, \mathbf{t}) > 1$), whereas almost one in four is beyond binarizable permutations. This highlights the need for future refinements of permutation sets into more tight subsets that characterize manual word alignments.

7.2 Automatic word alignments

We report empirical results on English-Dutch, English-French and English-German corpora obtained from the respective corpora in the Europarl collection (Koehn 2005) by setting an upperbound of 40 words on the sentence length on the source and target sides. Table 5 lists the sizes of the corpora. The corpora we use are about half the size of the original corpora but we think that they are large enough to contain word alignments Representative of the subsuming, full-size corpora.

Language Pairs	**English-Dutch**	**English-French**	**English-German**
Total number of sentence pairs	945167	949408	995909
#skipped sentence pairs	745	257	453
#sentence pairs containing nulls	801948	783367	839335
Mean source length	21.295±8.916	20.556±8.585	21.559±9.138
Mean target length	21.212±9.011	22.552±9.383	20.459±8.872
Mean and STD of ratio of source to target lengths	1.029±0.219	0.934±0.201	1.081±0.244
Mean & STD #links per word	1.12±0.14	1.15±0.15	1.14±0.16

Table 5: The corpora used in our analysis (sentence length ≤ 40 words). The "mean #links per word" is calculated using as the mean over the corpus alignments for the ratio $\frac{\text{\# alignment links}}{\min_{x \in \{\mathbf{s}, \mathbf{t}\}} length\ of\ x}$.

Model	Number of training iterations
IBM model 1	4
HMM model	3
IBM model 3	3
IBM model 4	4

Table 6: GIZA++ training policy

	English-Dutch	English-French	English-German
$\beta_{max} = 2$	45.52%	52.84%	45.60%
$\beta_{max} = 3$	55.77%	67.27%	56.80%
$\beta_{max} = 4$	73.95%	82.60%	74.75%
$\beta_{max} = 5$	83.51%	90.01%	84.63%
$\beta_{max} = 6$	89.92%	94.40%	90.76%
$\beta_{max} = 7$	93.56%	96.72%	94.36%
$\beta_{max} = 8$	95.91%	98.06%	96.55%
$\beta_{max} = 9$	97.37%	98.85%	97.88%
$\beta_{max} = 10$	98.33%	99.31%	98.71%
$\beta_{max} = 15$	99.86%	99.95%	99.90%
$\beta_{max} = 21$	100.00%	100.00%	100.00%

Table 7: Coverage of the corpus as a function of β_{max} for symmetrized (grow-diag-final) word alignments in EuroParl parallel corpora (sentence length ≤ 40) for three language pairs

Following standard practice (e.g., (Koehn et al. 2007)), the sentences were lower-cased and tokenized using the relevant Moses scripts.[19] The sentence pairs were word aligned using GIZA++[20] with training iterations that are shown in Table 6.

The grow-diag-final heuristic was used for symmetrization of the alignments in the two translation directions. The grow-diag-final heuristic is currently considered best practice, and compared to *union* or *intersection* of alignments seems to give a good compromise between precision and recall, i.e. it aligns most words while not becoming overly imprecise in doing so.

Tables 7 and 8 show the alignment and TE Coverage results respectively. Unlike the manual word alignments, for the automatic alignments the coverage and TEC results increase dramatically for β_{max} values in $[2..6]$. For $\beta_{max} = 2$ the results are in the mid/low forties for English-Dutch/German and low fifties for English-French, but by $\beta_{max} = 6$ all results are in the nineties or thereabout.

Automatic alignments obtained by symmetrization heuristics (particularly grow-diag-final) are built in a way that allows the extraction of a large number of phrase pair equivalents. This could explain why binary branching HATs (BITTs) have lower coverage of such word alignments, and why more often than not a larger branching factor than two is needed. Table 9 supports this observation. On the one hand, for at most binary branching HATs ($\beta_{max} = 2$), the binarizability score is very low in the thirties (Dutch/German) or forties (French) suggesting that these word alignments are hard to capture with BITTs. On the other, by $\beta_{max} \geq 10$ the binarizability score is around the 83% for English-French suggesting HATs with many linked nodes, particularly when we contrast this with 62% for the Hansards manual alignment. It

19 `http://www.statmt.org/moses/`
20 `http://www-i6.informatik.rwth-aachen.de/Colleagues/och/software/GIZA++.html`.

	English-Dutch	English-French	English-German
$\beta_{max} = 2$	44.63%	51.99%	43.40%
$\beta_{max} = 3$	56.35%	68.55%	56.13%
$\beta_{max} = 4$	75.79%	84.64%	75.46%
$\beta_{max} = 5$	85.51%	91.93%	85.79%
$\beta_{max} = 6$	91.64%	95.83%	91.83%
$\beta_{max} = 7$	94.91%	97.71%	95.23%
$\beta_{max} = 8$	96.90%	98.73%	97.18%
$\beta_{max} = 9$	98.08%	99.28%	98.33%
$\beta_{max} = 10$	98.82%	99.59%	99.02%
$\beta_{max} = 15$	99.91%	99.97%	99.94%
$\beta_{max} = 20$	100.00%	100.00%	100.00%

Table 8: Translation Equivalents Coverage as a function of β_{max}

	English-Dutch	English-French	English-German
$\beta_{max} = 2$	33.45%	41.43%	32.75%
$\beta_{max} = 3$	42.40%	54.75%	42.45%
$\beta_{max} = 4$	58.38%	68.92%	58.19%
$\beta_{max} = 5$	66.47%	75.47%	66.59%
$\beta_{max} = 6$	71.75%	79.21%	71.62%
$\beta_{max} = 7$	74.63%	81.10%	74.48%
$\beta_{max} = 8$	76.43%	82.15%	76.14%
$\beta_{max} = 9$	77.52%	82.76%	77.14%
$\beta_{max} = 10$	78.22%	83.10%	77.74%
$\beta_{max} = 15$	79.27%	83.54%	78.57%
$\beta_{max} = 21$	79.35%	83.57%	78.62%

Table 9: Binarizability scores as a function of β_{max}

Kind of HATs (Permutation sets)	English-Dutch	English-French	English-German
$BITTs$ (Binarizable permutations)	45.52%	52.84%	45.60%
$PETs$ (Permutations)	52.63%	56.56%	52.55%
$HATs$ (Permutation sets)	100.00%	100.00%	100.00%

Table 10: The ratio of the different subsets of HATs in the corpus: BITTs, PETs and HATs

is unlikely that the latter difference can fully be explained by the difference in language use (Hansards vs EuroParl) and, in fact, the shorter average sentence length in the Hansards manually aligned corpus actually suggests the reverse situation should be true. This supports the hypothesis that the symmetrized automatic alignments are built such that they can facilitate extracting a larger number of phrase pair equivalents, leading to many more nodes in the HATs than manual alignments.

Table 10 shows that the coverage of BITTs around 52% for French and 45% for Dutch and German. The coverage of PETs (permutations) increases by 4-7% only, again suggesting that neither BITTs nor PETs (as pure permutation-devices) can provide good coverage of phenomena in word alignments. If only approximately 50% of all such word alignments can be represented fully as a permutation, then the other 50% demands the notion of a permutation set that can capture discontinuous alignments and complex many-to-many cases. Yet, permutation

sets remain simple extensions of permutations and HATs can be seen as conservative extensions of PETs and BITTs.

	English-Dutch	English-French	English-German
$D(\mathbf{a},\mathbf{s},\mathbf{t})=0$ (pure permutations)	52.63%	56.55%	52.55%
$D(\mathbf{a},\mathbf{s},\mathbf{t})=1$	75.86%	80.12%	75.29%
$D(\mathbf{a},\mathbf{s},\mathbf{t})=2$	90.11%	92.27%	89.28%
$D(\mathbf{a},\mathbf{s},\mathbf{t})=3$	96.05%	97.18%	95.60%
$D(\mathbf{a},\mathbf{s},\mathbf{t})=4$	98.44%	98.95%	98.19%
$D(\mathbf{a},\mathbf{s},\mathbf{t})=5$	99.39%	99.60%	99.28%
$D(\mathbf{a},\mathbf{s},\mathbf{t})=10$	100.00%	100.00%	100.00%

Table 11: Coverage of the corpus as a function of $D(\mathbf{a},\mathbf{s},\mathbf{t})$.

Table 11 shows the breakdown of the coverage statistic of automatic word alignments to $D(\mathbf{a},\mathbf{s},\mathbf{t})$, i.e., the measure of "distance" of the word alignment from being a pure permutation.

In symmetrized automatic alignments, it turns out that almost one half is beyond permutations, whereas by distance $D(\mathbf{a},\mathbf{s},\mathbf{t})\leq 5$ there is almost full coverage. Intuitively speaking, only a small number of deletions of alignment links in these alignments should result in alignments that resemble permutations. This does not reveal the full complexity of these alignments but it does suggest that the automatic alignment that are non-binarizable are less complex than their Hansards manual (Sure+Possible) counterpart. We think that this is because the Hansards manual alignments (Sure+Possible) are in many cases dense with links (see the "mean number of links per word" in Tables 1 and 5). Some of these manual alignments simply link almost all source words with almost all target words leading to rather flat HATs and fewer phrase pair translation equivalents than the automatic alignments.

7.3 Discussion and related empirical work on alignments

As stated earlier, the empirical results presented in the preceding sections are not particularly targeted at studying the coverage of a certain grammar formalism (like ITG). Nevertheless, we see the coverage of NF-ITG as an opportunity for a first application of our theory. Our results provide *directly and exactly computed* coverage results for NF-ITG, given the chosen semantics. Before we discuss these results, we will discuss a crucial aspect of how these results are obtained.

The debate concerning the representation power of (NF-)ITG for translation data continues and the reports concentrate mostly on *upper bounds* for the representation power of ITG in terms of manual or automatic word alignments (Zens and Ney 2003; Galley et al. 2004; Wellington, Waxmonsky, and Melamed 2006; Søgaard and Wu 2009; Søgaard 2010). Søgaard and Wu (Søgaard and Wu 2009) observe correctly that the reports in the literature differ considerably in at least four dimensions (i) Data: what data and which alignments are used, (ii) Metrics: the way the coverage is measured (sentence vs. translation unit levels), (iii) Semantics: how to interpret word alignments (disjunctively/conjunctively) and (iv) Algorithmics: the algorithm used for computing the upperbounds. In studying some of the existing literature we found it particularly difficult to pin down the exact choices made, which makes it even more difficult to interpret the reported results. There is, however, a good reason for the difficulty of exact description as we explain next.

In particular, choices (iii) and (iv) pertain to the formal problem of how to parse word alignments with a synchronous grammar. We think that the problem of measuring the exact coverage of word alignments by a synchronous grammar is particularly complicated because there is no

a priori, objective, formal grounds on which word alignments and synchronous grammars can be formally intersected. Before intersecting these two completely different representations, it is necessary to specify a *shared meaning/semantics*. Indeed, it is necessary to define the semantics of word alignments, as sets of translation equivalents, and represent them by synchronous trees with the same semantic interpretation as the trees generated by the synchronous grammar, i.e., linked pairs of nodes dominate pairs of fringes that constitute translation equivalents. Without fixing the latter relation between STPs and translation equivalents, it is difficult to talk about how word alignments exactly intersect with an unknown instance of a synchronous grammar. In essence, here we define for every word alignment a semantically equivalent set of HATs, and then we check that these HATs can be built by an instance of the grammar formalism, i.e., that there exists an instance at all that can generate these HATs. As defined earlier, for measuring sentence-level coverage, we check that there exists an instance of the grammar formalism that can generate all these HATs exactly. And for coverage/recall of translation equivalents/units (TEC), we check the percentage of linked nodes that can be generated by an instance of the grammar formalism. This is exactly the methodology followed by the present work.

Our results with $\beta_{max} = 2$ are in fact exact coverage and TEC results for NF-ITG. Some earlier work has concentrated on measuring *upperbounds* on the coverage/TEC either by discounting the complex alignment cases that cannot be covered by NF-ITG, e.g., (Søgaard and Kuhn 2009; Søgaard and Wu 2009), or by defining a "lighter" semantics of word alignments by employing a disjunctive interpretation (as opposed to the more accepted conjunctive interpretation) (Wellington, Waxmonsky, and Melamed 2006). In contrast with earlier work, our results are exact results, formally based on the intersection idea outlined above and are subject to the chosen semantics.

As far as we can see, the results presented in (Zens and Ney 2003; Wu, Carpuat, and Shen 2006; Søgaard 2010), although based on a different approach and pertain to different data sets and word alignments, are measured in ways that might be (close to) implementing the intersection advocated here. Zens and Ney (2003) employ Viterbi alignments on Hansards data (sentence length up to 30 words) and obtain far higher coverage results for NF-ITG ($\approx$ 81% and 73% depending on direction) than our results for English-French EuroParl data with symmetrized word alignments ($\approx$ 53%). Apart from differences in corpus data, symmetrized alignments, constituting the backbone data for training state-of-the-art systems, are known to be distinctively different from their Viterbi uni-directional ancestors. The coverage result of (Søgaard 2010) on the manual Hansards data (77%) comes very close to our coverage result (72%). Søgaard (2010) is presented densely and somewhat informally that some details escape us. We attribute the difference to various reasons, including sentence-length differences (in (Søgaard 2010) the cutoff is 15 words) and choices of how to define translation equivalents with unaligned words on either side. The work of (Wu, Carpuat, and Shen 2006) concerns Arabic-English data, which is not studied here.

A completely distinct work that reports measures of word alignment coverage under ITG constraints is (Huang et al. 2009). The work is based on the GHKM (Galley et al. 2004) method of extracting synchronous rules, which involves target-language syntax. The authors report percentages of binarizable synchronous rules extracted from the word alignments. The results reported are incomparable to our results for NF-ITG because they are subject to the GHKM extraction method of synchronous rules, which encapsulate very difficult word alignments as internal, lexical parts of a synchronous rule. By doing so, the coverage is measured with regards to a different semantics (the GHKM extraction method) of word alignments than our choice of semantics. Our semantics of word alignments is more exhaustive than the GHKM semantics in that we allow all phrase pairs to be extracted without constraints from monolingual syntax or other performance-driven constraints.

Søgaard and Wu (2009) argue convincingly that ITG has a different coverage than its normal-form variant (akin to Chomsky normal form). We would like to point out that the coverage of permutations (represented by PETs in Tables 3 and 10) actually sets an upperbound on the coverage of the ITG formalism for the word alignments studied here. A lower-bound is set by the binarizable permutations (NF-ITG). Having set a lowerbound and an upperbound, we hope in future work, dedicated to this topic, to calculate exact coverage figures under the chosen semantics for an "all-accepting ITG'.

8. Conclusions and future work

In this paper we presented a theory of word alignments that endows them with semantics from machine translation and with hierarchical representations that capture recursive properties of these semantic units. Viewing word alignments from the perspective of word order differences, initially we extended permutations into permutation sets that constitute asymmetric representations of word alignments. Subsequently we defined the semantics of word alignments in terms of sets of translation equivalents, and advocated the idea that the hierarchical structure of word alignments (as permutations sets) should represent the compositional build up of all translation equivalents found in the word alignment. By adopting phrase pair semantics, this paper exemplifies this general idea, and presents an algorithm for building a set of semantically equivalent Hierarchical Alignments Trees (HATs) for every word alignment. The HAT representation, a conservative extension of the synchronous trees known from ITG, is shown to posses some attractive properties, particularly that every node in every HAT is minimally branching given the choice of the semantics and the kind of hierarchical representation.

On the empirical side we exemplified the use of this theory to analyze the hierarchical properties of word alignments, where we analyzed manual and automatically acquired word alignments. Our analysis concentrates on a break-down of statistics of word alignments by the maximal branching factor needed in the HATs that represent them. Capitalizing on this idea, we report exact coverage results of word alignments for normal-form ITG and advocate the need for a more rigorous approach to measuring the coverage of synchronous grammars using our HAT representation. Particularly, we argued that it is crucial to pin down the semantics of word alignments and their hierarchical representations, as well as the semantics of the trees generated by a grammar, in order to measure the coverage as the intersection between two sets of synchronous trees: the set of HATs defined for a word alignment and the set of synchronous trees generated by an all-accepting grammar for the sentence pair. We implement this idea by imposing the formal constraints of the grammar on the set of HATs, filtering out the HATs (or parts of) that do not abide by these constraints.

In future work we plan to study efficiency aspects of the present algorithms. We will explore different semantics for word alignments and possibly different definitions of node alignments (possibly generalizing layered linking somewhat). Besides studying the exact coverage of ITGs and other synchronous grammars, we expect that our HAT representation can be used to shed light on the stability of the Direct Correspondence Assumption of monolingual syntactic representations projected using word alignments, e.g., (Fox 2002; Hwa et al. 2002). Similarly, we also expect that the HAT representation will allow us to extract a variety of probabilistic synchronous grammars which capture varying degrees of statistical independence between translation units. Finally, we hope that this study prepares the ground for novel and useful methods for the automatic learning of hierarchical alignments in parallel corpora, the original topic that lead to this study.

References

Bod, R., R. Scha, and K. Sima'an, editors. 2003. *Data Oriented Parsing*. CSLI Publications, Stanford University, Stanford, California, USA.

Chiang, D. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, MI, USA.

Chiang, D. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 2(33):201–228.

DeNero, J., D. Gillick, J. Zhang, and D. Klein. 2006. Why generative phrase models underperform surface heuristics. In *Proceedings of the workshop on SMT*, pages 31–38.

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271.

Eisner, Jason. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of ACL 2003 (companion), 41st Annual Meeting of the Association for Computational Linguistics, Companion Volume to the Proceedings, 7-12 July 2003, Sapporo Convention Center, Sapporo, Japan*, pages 205–208. The Association for Computational Linguistics.

Fox, Heidi J. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 304–311, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Huang, Liang, Hao Zhang, Daniel Gildea, and Kevin Knight. 2009. Binarization of synchronous context-free grammars. *Computational Linguistics*, 35(4):559–595.

Hwa, Rebecca, Philip Resnik, Amy Weinberg, and Okan Kolak. 2002. Evaluating translational correspondence using annotation projection. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL 2002)*, pages 392–399.

Janssen, Theo M. V. 1998. An overview of compositional translations. In *Revised Lectures from the International Symposium on Compositionality: The Significant Difference*, COMPOS'97, pages 327–349, London, UK. Springer-Verlag.

Koehn, Ph. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the MT Summit X*, pages 79–86, September.

Koehn, Ph., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: open-source toolkit for statistical machine translation. In *Proceedings of ACL'07*, pages 177–180, Prague, Czech Republic.

Koehn, Ph., F. Och, and D. Marcu. 2003. Statistical phrase-based machine translation. In *In proceedings of Human Language Technologies and the North-American Chapter of the ACL (HLT-NAACL)*, pages 48–54, Edmonton, Canada.

Landsbergen, Jan. 1982. Machine translation based on logically isomorphic montague grammars. In *Proceedings of the 9th conference on Computational linguistics - Volume 1*, COLING '82, pages 175–181, Czechoslovakia. Academia Praha.

Lewis, II, P. M. and R. E. Stearns. 1968. Syntax-directed transduction. *J. ACM*, 15:465–488, July.

Marcu, D. and W. Wong. 2002. A Phrase-based, Joint Probability Model for Statistical Machine Translation. In *Proceedings of EMNLP02*, pages 133–139.

Mylonakis, M. and K. Sima'an. 2011. Learning hierarchical translation structure with linguistic annotations. In *Proceedings of The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL:HLT 2011)*.

Mylonakis, Markos and Khalil Sima'an. 2010. Learning probabilistic synchronous cfgs for phrase-based translation. In *Fourteenth Conference on Computational Natural Language Learning*, Uppsala, Sweden, July.

Och, F. and H. Ney. 2000. Improved statistical alignment models. In *Proceedings of the the 38th Annual Meeting on Association for Computational Linguistics (ACL)*, pages 440–447.

Och, F. and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Poutsma, Arjen. 2000. Data-oriented translation. In *Proceedings of the International Conference on Computational Linguistics (COLING'00)*, pages 635–641.

Satta, Giorgio and Enoch Peserico. 2005. Some computational complexity results for synchronous context-free grammars. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 803–810, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Sima'an, K. 1999. *Learning Efficient Disambiguation*. PhD dissertation (University of Utrecht). ILLC dissertation series 1999-02, University of Amsterdam, Amsterdam.

Søgaard, Anders. 2010. Can inversion transduction grammars generate hand alignments? In *Proccedings of the 14th Annual Conference of the European Association for Machine Translation (EAMT)*.

Søgaard, Anders and Jonas Kuhn. 2009. Empirical lower bounds on alignment error rates in syntax-based machine translation. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation*, SSST '09, pages 19–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Søgaard, Anders and Dekai Wu. 2009. Empirical lower bounds on translation unit error rate for the full class of inversion transduction grammars. In *Proceedings of the 11th International Workshop on Parsing Technologies (IWPT-2009), 7-9 October 2009, Paris, France*, pages 33–36. The Association for Computational Linguistics.

Tinsley, John, Mary Hearne, and Andy Way. 2009. Exploiting parallel treebanks to improve phrase-based statistical machine translation. In Alexander F. Gelbukh, editor, *Proceedings of the Computational Linguistics and Intelligent Text Processing, 10th International Conference, (CICLing 2009)*, volume 5449 of *Lecture Notes in Computer Science*, pages 318–331. Springer.

Tinsley, John and Andy Way. 2009. Automatically generated parallel treebanks and their exploitability in machine translation. *Machine Translation*, 23(1):1–22.

van Noord, G.J. 1995. The intersection of finite state automata and definite clause grammars. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL'95)*, pages 159–165.

Wellington, Benjamin, Sonjia Waxmonsky, and I. Dan Melamed. 2006. Empirical lower bounds on the complexity of translational equivalence. In *Proceedings of the Annual Meeting of the ACL*.

Wu, D. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, Santa Cruz.

Wu, D. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 3(23):377–403.

Wu, D. and H. Wong. 1998. Machine translation wih a stochastic grammatical channel. In *Proceedings of ACL-COLING'98*, pages 1408–1415, Columbus, OH, USA.

Wu, Dekai, 2010. *Alignment. In CRC Handbook of Natural Language Processing, Second Edition*, pages 367–408. CRC Press.

Wu, Dekai, Marine Carpuat, and Yihai Shen. 2006. Inversion transduction grammar coverage of arabic-english word alignment for tree-structured statistical machine translation. In *Proceedings of the IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*.

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

Zens, R., F. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In *Proceedings of KI: Advances in Artificial Intelligence*, pages 18–32.

Zens, Richard and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the Annual Meeting of the ACL*, pages 144–151.

Zhechev, Ventsislav and Andy Way. 2008. Automatic generation of parallel treebanks. In Donia Scott and Hans Uszkoreit, editors, *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 1105–1112.

Zollmann, A. and A. Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of the North-American Chapter of the ACL (NAACL'06)*, pages 138–141.