# Syntax-driven Learning of Sub-sentential Translation Equivalents and Translation Rules from Parsed Parallel Sentences


## Abstract

We describe a multi-step process for automatically learning reliable sub-sentential syntactic phrases that are translation equivalents of each other and syntactic translation rules between two languages. The input to the process is a corpus of parallel sentences (which are assumed to be translation equivalents). The sentence pairs are assumed to be annotated with phrase-structure parse trees, and with word alignments representing translation equivalence at the word-level between the sentences. Our method consists of three steps. In the first step, we apply a newly developed algorithm for aligning parse-tree nodes between the two parallel trees. Parse trees nodes are aligned if their constituent yields are likely translation equivalents of each other. In the second step, we extract all aligned sub-sentence syntactic constituents from the parallel sentences, and create a syntax-based phrase-table. In the third and final step, we treat the node alignments as tree decomposition points and extract from the corpus all possible synchronous parallel subtrees. The synchronous parallel subtrees are also converted into synchronous context-free rules. The extracted syntax-based phrase tables and translation rules can then be used as resources for constructing broad-coverage syntax-based Machine Translation systems between the two languages. We apply the approach and evaluate it on both a small manually word-aligned Chinese-English parallel treebank, and on large scale automatically derived Chinese-English parallel corpora.


## 1 Introduction

**Alon**

This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.

## 2 Related Work

**(Alon+all)**

## 3 PFA Algorithm for Node Aligment

A phrase table typically used for Machine Translation, such as one built by the Moses(**?**) system, is a list of phrases on the source side, with their equivalent phrases on the target side, and sets of scores that tells how good the phrases are. What these tables do not tell us is whether a particular phrase is a noun-phrase, a verb-phrase, or something else. This information is very essential for using a phrase table in our Statistical Transfer-based MT approach. We thus wanted to build a phrase table that tells us the category of each phrase, both on the source side, and on the target side. We came up with the PFA Algorithm that uses a parsed parallel corpus and information about word alignments to build such a phrase table.

### 3.1 Objectives of the Algorithm

Given a pair of parallel sentences, and their corresponding parse trees, our goal is to extract contiguous pieces of text that are translation equivalents of each other. Also, we require that these "phrases" should belong to constituent structures in their parse trees. In short, we are looking to find a pair of nodes between the source and target trees whose yields carry the same meaning.

For the purpose of extracting constituent structures with like meaning, we use the word-alignment information of the underlying sentences. The assumption here is that if two words are aligned to each other, they carry the same meaning. In practice however, we know that automatically obtained word alignments are rather noisy. Hence, the algorithm should be robust to deal with such noisy word alignments.

### 3.2 Related Work

Aligning nodes in parallel trees and extraction of phrasal lexicons has been done by Samuelsson et al.(**?**). Their approach involves manual alignment of the nodes in the tree. The manual approach is very well suited for generating very reliable treebanks, however automatic methods of aligning nodes and extracting lexicons are useful in accumulating resources from large parallel data.

Tinsley et al.(**?**) have used statistical lexicons to align nodes between parallel trees. In our approach, we use word alignment information, which is more reliable than just a statistical lexicon(**?**).

Groves et al.(**?**) have suggested a method of aligning nodes between parallel trees automatically, based on word alignments. Along with the word alignment information, they also use information of the labels of nodes in the trees, and the general structure of the tree. Our approach is different in the sense that we only look at the word alignments, thereby making the system robust to using any parser, even different parsers on different language pairs. However, in general terms, this approach is the closest to our approach.

### 3.3 Wellformedness constraints

The PFA node alignment algorithm produces as output node-pairs $(S_i, T_j)$. Each of this pair tells us which node in the source tree was aligned to which node in the target tree. The alignments are required to satisfy these wellformedness criteria:

- If a node $S_i$ is linked to a node $T_j$, then any node in the sub-tree of node $S_i$ can be linked only to nodes in the subtree of node $T_j$.
- If a node $S_i$ is linked to a node $T_j$, then any node that dominates the node $S_i$ can be linked only to nodes that dominate the node $T_j$.
- If a node $S_i$ is linked to a node $T_j$, then the following must hold good for the word-alignments of the underlying sentences:
  - Every word in the yield of the node $S_i$ should either be aligned to one or more words in the yield of the node $T_j$, or it should be unaligned.
  - Every word in the yield of the node $T_j$ should either be aligned to one or more words in the yield of the node $S_i$, or it should be unaligned.
  - There should be at least one alignment between the yields of nodes $S_i$ and $T_j$. Thus, the words in the yields can not all be unaligned.

### 3.4 Unaligned Words and Contiguity

If we consider a phrase in one language, and its translation equivalent in another language, we often find that there are a few unaligned words. One common example is: some languages use determiners, while others don't. In cases like this, we could have a pair of constituents between parallel trees that contain unaligned words, but still carry the same meaning. The PFA node-alignment algorithm allows for such "extra words" to be matched.

Different languages have different word orders. In English, an adjective always comes before a noun, while in French, in most cases, the adjective follows its noun. The node-alignment algorithm should be robust to different word orders. As long as one piece of contiguous text dominated by a node carries the same meaning as the yield of a node in the parallel tree, the two nodes must be aligned. The PFA algorithm satisfies this property.

### 3.5 Mathematics and Meaning

Assuming that natural language is infinite, there could be infinite sentences, and thus infinite possible 'meanings' conveyed in text. In our approach, we have mapped all these meanings onto the set of natural numbers. One number represents one meaning.

Number theory classifies the set of natural numbers into {unity, primes, and composites}. Because each number carries a meaning, we have the following:

1. Prime Numbers: They represent a unique meaning. In a particular context, the word 'house' could mean just one thing. That meaning would be captured by a prime number.

2. Composite Numbers: They can be written as the product of prime numbers. A composite number represents the composite meaning of the meaning of its prime factors. Thus, a number that represents the phrase 'blue house' would be represented by a number that is a product of the numbers assigned to 'blue' and to 'house'.

3. Unity: A number does not change when multiplied with unity. Hence, the number 1 represents a "don't care" meaning.

In a parse tree, the yields of all nodes are different texts. Thus, each node can be supposed to carry a different meaning. Since we can represent meaning as a number, we can store that number in the node.

### 3.6 Description of the PFA Algorithm

The PFA algorithm uses the concept of 'composite meaning as prime factorization', and hence the name (Prime Factorization and Alignments). The principal idea of the algorithm can be summarized in three points:

1. If we have a node in the tree $P \rightarrow C_1 C_2 \ldots C_k$, then we assume that the meaning captured by the node $P$ is the composite meaning that is captured by its children. Thus, the value that node $P$ would get would be the product of values of its child nodes.

2. Leaf nodes can be assumed to carry unique meaning, hence they would be assigned distinct prime numbers, based on the word stored at the leaf.

3. Consider that a node $S_i$ in the source tree gets the same value as the node $T_j$ in the target tree. We have defined that every number carries a distinct meaning of its own. Since the values are same, the yields of the two nodes must be translation equivalents of each other, and hence the nodes must be aligned. (Sometimes, the parse trees contain unit productions. In that case, the values of both the nodes in that production come out to be the same, and for purposes of alignment, we break the tie by taking the node closer to the bottom of the tree.)

The PFA algorithm thus assigns values to the leaf nodes, propogates the values up the tree, and finally compares the values across trees to align the nodes. To assign the values to the leaf nodes, we take the help of word-alignment information. Earlier, we stated the assumption that two aligned words are said to carry the same meaning. Thus, those two words would be represented by the same prime number. In general, we would use as many distinct prime numbers as the number of alignments. Leaves representing aligned words would be assigned the prime number corresponding to that alignment. There are two exceptions to consider:

1. Leaf nodes corresponding to unaligned words are assigned the value 1, and these words are considered to be "don't cares" of meaning.

2. There could be one-to-many word alignments. Each of these alignments is considered to carry the same meaning, and is assigned the same value. However, if a word has two alignments, the alignment is simply considered to be taken twice, and thus the corresponding leaf node would store the square of the alignment's prime value.

Algorithm 1 describes a way to implement the PFA strategy. We implemented the algorithm in C++, and one small issue that came up when dealing with products of prime numbers is that for large trees, the value of the root node becomes a large

product, and can no longer be stored as a built-in C data type. Using the GMP library solved this problem.

**Algorithm 1** PFA Algorithm for Node-Alignment

**Require:** Source-Tree, Target-Tree, Word-Alignments

**for all** nodes $n$ in source, target trees **do**
  Value$(n) \leftarrow 1$
**end for**
Initialize Prime-number generator with 2.
**for all** $A$ in Word-Alignments **do**
  Value$(A) \leftarrow$ next prime number.
  **for all** word $w$ that alignment $A$ refers to **do**
    $L \leftarrow$ Leaf node corresponding to $w$.
    Value$(L) \leftarrow$ Value$(L) *$ Value$(A)$.
  **end for**
**end for**
**for all** node $n$ in source tree, traversed bottom up **do**
  Value$(n) \leftarrow \prod$ Value$(Child(n))$.
**end for**
**for all** node $n$ in target tree, traversed bottom up **do**
  Value$(n) \leftarrow \prod$ Value$(Child(n))$.
**end for**
Find unaligned nodes $n$ in source tree and $m$ in target tree with matching values, but not the value 1. If multiple nodes in the same tree have the same value, choose the node closer to the bottom of the tree. Align the node $n$ to node $m$. Repeat this step.

It can be verified that the PFA algorithm satisfies the wellformedness constraints. Also, since the product of numbers is commutative, the algorithm is robust to differing word orders within parallel constituent structures. Since unaligned words act as don't cares, the system allows for extra words (such as determiners) to be included in the constituent phrase, thereby finding the closest translation equivalence between the given sentences. We said that in case of a tie between values of two nodes, we choose the node closer to the bottom. Earlier, we claimed that this happens in case of unit productions. However, this could also happen if there was a set of unaligned words attached directly to the top-node, which is why its value was the same as the bottom-node. In that case, we do not align the top node, thereby making the system robust against noisy alignments.

The PFA algorithm run on a sample Chinese-English parallel sentence is shown in Figure 1. The value of each node as shown as a part of its label. The aligned nodes are marked by shapes. A triangle aligns to a triangle, and squares to squares.

## 4 Syntax-based Sub-sentential Phrase Extraction

The alignment of nodes as described in the previous section allows us to build a comprehensive phrasal lexicon out of the parallel corpus. As previously described, nodes are aligned only if the meaning that the carry under them in the trees is nearly identical. Clearly, if two nodes $S$ and $T$ are aligned to each other, the yield $y_1$ of node $S$ must be same in meaning as the yield $y_2$ of node $T$. We could add an entry to our lexicon, where $y_1$ maps onto $y_2$. The syntactic category of the source-phrase would be the label on the node $S$, and similarly the category of the target-phrase would be the label on the node $T$. By considering all aligned pairs generated from a parallel corpus, we can extract many such entries for our lexicon.

The set of phrases extracted from the sentence from Figure 1 is shown in Figure 2.

| Source Category | Target Category | Source | Target |
|---|---|---|---|
| IP | S | 澳洲 是 与 北韩 有 邦交 的 少数 国家 之一 。 | Australia is one of the few countries that have diplomatic relations with North Korea. |
| VP | VP | 是 与 北韩 有 邦交 的 少数 国家 之一 | is one of the few countries that have diplomatic relations with North Korea |
| NP | NP | 与 北韩 有 邦交 的 少数 国家 之一 | one of the few countries that have diplomatic relations with North Korea |
| VP | VP | 与 北韩 有 邦交 | have diplomatic relations with North Korea |
| NP | NP | 邦交 | diplomatic relations |
| NP | NP | 北韩 | North Korea |
| NP | NP | 澳洲 | Australia |

Figure 2: Phrases extracted from Aligned Nodes

The approach of building such a phrase table from large corpora of data is quite similar to the phrase tables traditionally built for phrase-based SMT methods. There, one starts with parallel corpus, aligns the words, extracts the phrase table and assigns scores to each phrase. In our approach too, we start with a parallel corpus, align the words, and then use the parse

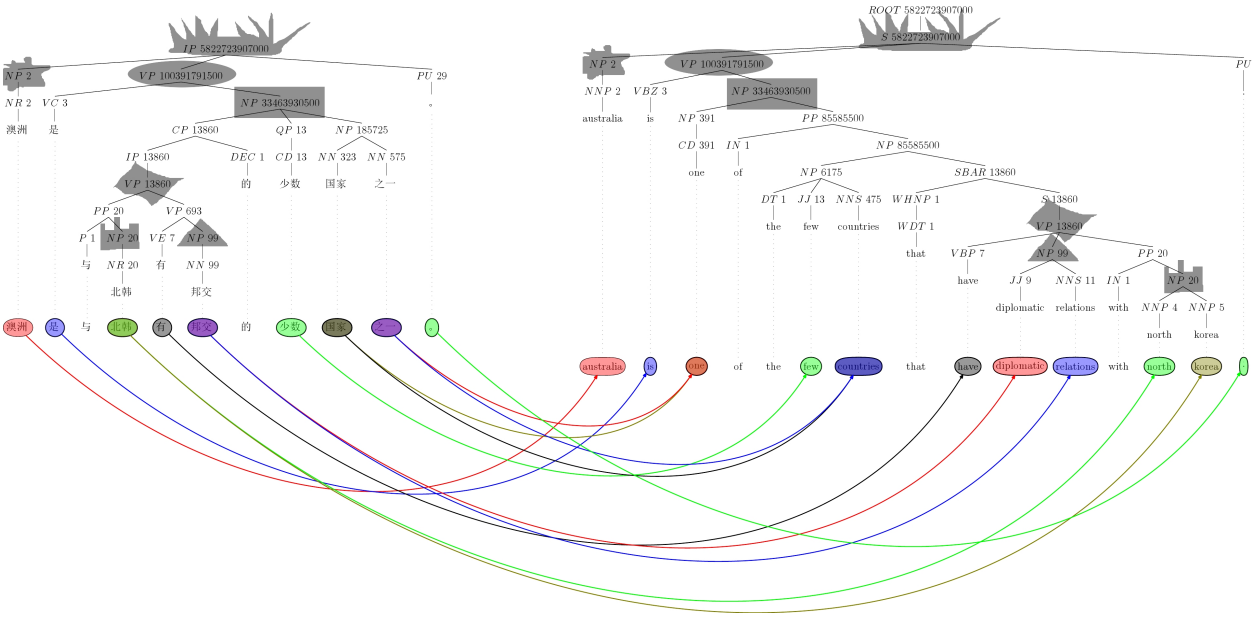

Figure 1: Node-Aligned parallel sentences

trees to extract a syntax based phrase table. In our initial experiments for using this phrase table in the Statistical Transfer-based decoder, we have naïvely used the relative frequency of the phrases as a scoring function. We are considering using more sophesticated scoring strategies in our upcoming experiments.

## 5 Evaluation of the PFA algorithm

THIS SECTION IS UNDER CONSTRUCTION!

The PFA algorithm uses only the parse trees and word alignment information. We ran the algorithm over a corpus of about 3000 sentences from the Chinese-English Treebank. For this set of parallel sentences, we have manually verified parse trees as well as manually generated word alignment information. We generated node alignments for the corpus, and used these alignments to extract a phrasal lexicon. Chinese-English bilingual speakers verified the lexicon, and attested that the entries are extremely precise.

The PFA algorithm does not require that the word alignments be highly accurate, or manually generated. This allows us to run the algorithm on large corpora that are automatically parsed and automatically word aligned. In order to study the effect of noisy alignments on the phrasal lexicon extracted from the output of PFA algorithm, we conducted tests on the Treebank corpus mentioned above.

The node alignments extracted from "Manually Parsed, Manually Aligned" Treebank corpus were taken as a gold standard for comparison, especially since bilingual speakers had attested about its precision. This gold standard was compared to the node alignments extracted from "Automatically Parsed' Automatically Aligned" version of the corpus. The Stanford parser(**?**) was used to parse the Chinese and English text. GIZA was used to align the words in both the Chinese-English and English-Chinese directions. Different strategies of combining Viterbi alignments were used, and a set of node-alignments was generated from each of these alignments and the parse trees. We evaluated the precision and recall of the entries in these node-alignments against the gold standard. The results of the tests are summarized in Table 1. We observe that for purposes of node-alignment, the sym2-algorithm for combining bidirectional alignments (from the Thot toolkit(**?**)) performed best.

## 6 Synchronous Sub-tree and CFG Rule Extraction

Synchronous reordering rules have been used to improve Machine Translation quality. Recent success

| Viterbi Combination Strategy | Precision | Recall |
|---|---|---|
| Intersection | 0.6278 | 0.5525 |
| Union | 0.8054 | 0.2778 |
| Sym1 | 0.7182 | 0.4525 |
| Sym2 | 0.7170 | 0.4602 |
| Grow-Diag-Final | 0.4040 | 0.2500 |

Table 1: Evaluation of Node Alignment on Treebank Corpus

of syntactic and hierarchical machine translation systems have inspired the research in the learning of transfer rules automatically from corpus. While on one end we have grammar formalisms like ITG that enforce a particular reordering constraint to induce reordering rules from corpus, some more syntactic formalisms like Lexical Functional Grammar has also been used. In almost all the methods syntactic information is used on one side of the parallel corpus, typically the target side, in the form of a syntactic parsed tree. Then using word alignment information, a target side reordering rule is induced, that is typically hierarchical in nature. Andy Way (**?**) also discuss using syntactic information on the source side. Our work is closest in nature to that of Michael Galley (**?**; **?**), in that they extract rules for a Syntactic MT system using only information only on the target side of the language pair. We use syntactic information from both the languages in the form of syntactic parses, and extract much precise rules at the cost of some recall.

In this section we discuss our syntactic rule extraction process. We treat the node alignments extracted in the previous section as tree decomposition points and extract from the corpus all possible synchronous parallel subtrees. The synchronous parallel subtrees are also converted into synchronous context-free rules.

### 6.1 Parallel sub-tree pair extraction

The tree pair extraction is driven by the source side syntactic tree. We traverse the source tree in a top down fashion from the root to the leaf. At each node we check to see if there exists a parallel node alignment in the target syntactic tree. If a parallel node alignment exists then the node pair forms a synchronous decomposition point. The sub-tree at the node in the source parse tree and the corresponding subtree at the aligned node in the target parse tree form a synchronous subtree pair. All sub subtree pairs are initially extracted to form a parallel subtree database. The subtree pairs are similar in formalism to the Synchronous Tree Insertion Grammar **??**

As can be seen in the example in **??** the two points act as decomposition points.

### 6.2 Synchronous Transfer Rule Creation

The transfer rules in our system are extracted out of the synchronous sub-tree pairs. Each of the extracted sub-tree pairs is flattened into a synchronous context free style rule. The parent label of the source sub-tree becomes the source side parent category of the rule and the parent label of the target sub-tree becomes the target side parent category. A decomposition point in the sub-tree pair is flattened to form a node label in the rule. Our transfer rule formalism can be seen below.

```
VP::VP[NP "the" VP]->[NP VP]
(
(*score* 0.054)
(X1::X2)
)
```

Our transfer rule formalism is similar in spirit to the Lexicalized functional grammar. It was designed to capture not only the structural reordering but also the feature based constraints that can be used in the parsing stage of the Transfer Based MT.

In brief, each rule has a unique identifier followed by the context-free form of the production rules for both source and target sides. The source side production is used in the analysis phase of the transfer engine and the target side production is used in the transfer phase. The alignments information that decides the ordering of the constituents in the production rule is provided with 'X' followed by index for source side and 'Y' followed by an index for the target side. The rule can also have lexicalized items on either sides, in which case no alignment information is required for those index position. Finally the rules also can have an under-specified feature structure that has both 'x' side constraints that get used in the analysis phase and 'y' side constraints that get used in the generation phase. There are also agreement constraints that involve both 'x' side and 'y' side, which help in resolving ambiguity during

the transfer phase. In this work we do not consider the learning of feature constraints as part of the rule learning framework. We only concentrate on producing context-free style synchronous transfer rules.

Our approach to learning synchronous transfer rules is to obtain precise rules that are generic in nature. Rule induction is quite error prone and requires highly accurate word alignments and parse structures. We therefore only learning rules from controlled data with manual alignments. However, such data is quite scarce and difficult to create. Therefore, in our rule learning process, we generalize as much as possible on top of the rules learnt. Apart from generalizing to the constituent level, we also generalize to part-of-speech category, any lexical item in the source side of the rule that is one-to-one aligned with a lexical item in the target side of the rule. Since in this scenario, we have access to the part-of-speech information of both source and target side, we retain the information in the process of generalization. Any lexical item, that is unaligned is retained as lexicalized in the final rule.

The phrase table extracted from the corpus and the rules are scored together. The scores are source conditioned probabilities and the final synchronous probabilistic context free rules are used in our Transfer based MT. The rule scores are calculated as below.

$$P(s/t) = \frac{count(sourcetype, targettype, source, target)}{count(source)} \quad (1)$$

As part of our experiments we ran the rule learning on 10K corpus that was released as part of the LDC English Chinese parallel tree bank. The corpus was node aligned as described in Section 3. Rules were then extracted and flattened out to form synchronous context free rules. An example of the synchronous tree pairs extracted from example in Figure 4 is shown below in Figure 1. After generalization and flattening, we obtain rules as shown in Figure 4.

## 7 Analysis of Chinese-English System

We used the pipeline of node-alignment followed by rule-extraction to build resources for the Statistical Transfer-based translation system from Chinese to English. The syntax-based phrase table was constructed from two large corpora. One of them was a corpus of about 1.2M sentences. Using these sentences, we built a syntax-based phrase table of about 9.2M entries. The other data source was about 2.6M sentences, but many of its entries were from a Chinese-English lexicon. For this corpus, we extracted 8.75M entries in the phrase-table.

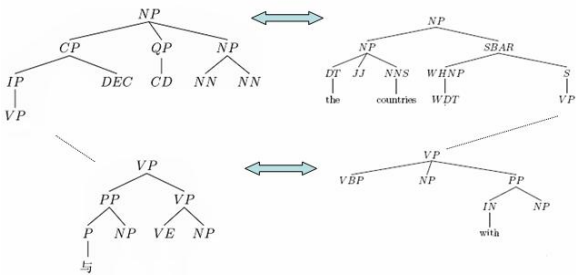


Figure 3: Sample sub-tree pairs extracted from aligned nodes

This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.

## 8 Conclusions

**?**

This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.This section is not yet written down.

```
VP::VP [VC NP] -> [VBZ NP]
(
(*score* 0.5)
(X1::Y1);(X2::Y2)
)

VP::VP [VC NP] -> [VBZ NP]
(
(*score* 0.5)
(X1::Y1);(X2::Y2)
)

VP::VP [北 NP VE NP] -> [ VBP NP with NP]
(
(*score* 0.5)
(X2::Y4);(X3::Y1);(X4::Y2)
)

IP::S [ NP VP ] -> [NP VP ]
(
(*score* 0.5)
(X1::Y1) ; (X2::Y2)
)
NP::NP [VP 北 CD 有 邦交 ] -> [one of the CD countries that VP]
(
(*score* 0.5)
;; Alignments
(X1::Y7); (X3::Y4)
)
```

Figure 4: Rules extracted from the Treepairs

| Corpus | Size (sens) | Rules with Structure | Rules (count>=2) | Complete Lexical rules |
|---|---|---|---|---|
| Parallel Treebank (3K) | 3,343 | 45,266 | 1,962 | 11,521 |
| 993 sentences | 993 | 12,661 | 331 | 2,199 |
| Parallel Treebank (7K) | 6,541 | 41,998 | 1,756 | 16,081 |
| Merged Corpus set | 10K | 94,117 | 3160 | 29,340 |

Figure 5: Rule statistics extracted from Chinese-English Corpora