# Finding Structural Correspondences from Bilingual Parsed Corpus for Corpus-based Translation

Hideo Watanabe*, Sadao Kurohashi** and Eiji Aramaki**

* IBM Research, Tokyo Research Laboratory
1623-14 Shimotsuruma, Yamato,
Kanagawa 242-8502, Japan
watanabe@trl.ibm.co.jp

** Graduate School of Informatics, Kyoto University
Yoshida-honmachi, Sakyo,
Kyoto 606-8501, Japan
kuro@i.kyoto-u.ac.jp,
aramaki@pine.kuee.kyoto-u.ac.jp

## Abstract

In this paper, we describe a system and methods for finding structural correspondences from the paired dependency structures of a source sentence and its translation in a target language. The system we have developed finds word correspondences first, then finds phrasal correspondences based on word correspondences. We have also developed a GUI system with which a user can check and correct the correspondences retrieved by the system. These structural correspondences will be used as raw translation patterns in a corpus-based translation system.

## 1 Introduction

So far, a number of methodologies and systems for machine translation using large corpora exist. They include example-based approaches [7, 8, 9, 12], pattern-based approaches [10, 11, 14], and statistical approaches. For instance, example-based approaches use a large set of translation patterns each of which is a pair of parsed structures of a source-language fragment and its target-language translation fragment. Figure 1 shows an example of translation by an example-based method, in which translation patterns (p1) and (p2) are selected as similar to a (left hand) Japanese dependency structure, and an (right hand) English dependency structure is constructed by merging the target parts of these translation patterns[1].

In this kind of system, it is very important to collect a large set of translation patterns easily and efficiently. Previous systems, however, collect such translation patterns mostly manually. Therefore, they have problems in terms of the development cost.

[1]Words in parenthesis at the nodes of the Japanese dependency structure are representative English translations, and are for explanation.

This paper tries to provide solutions for this issue by proposing methods for finding structural correspondences of parsed trees of a translation pair. These structural correspondences are used as bases of translation patterns in corpus-based approaches.

Figure 2 shows an example of extracting structural correspondences. In this figure, the left tree is a Japanese dependency tree, the right tree is a dependency tree of its English translation, dotted arrows represent word correspondence, and a pair of boxes connected by a solid line represent phrasal correspondence. We would like to extract these

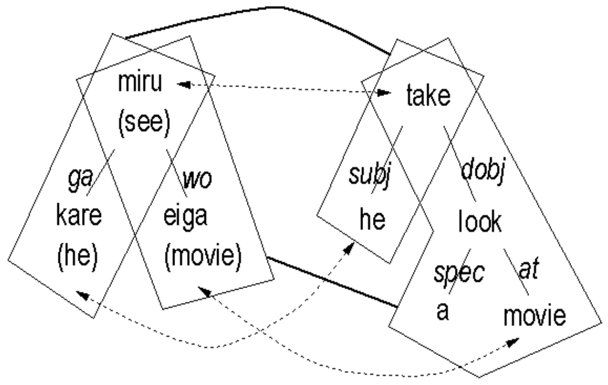


Figure 2: An Example of Finding Structural Correspondences

word and phrasal correspondences automatically. In what follows, we will describe details of procedures for finding these structural correspondences.

## 2 Finding Structural Correspondences

This section describes methods for finding structural correspondences for a paired parsed trees.

### 2.1 Data Structure

Before going into the details of finding structural correspondences, we describe the data format of a

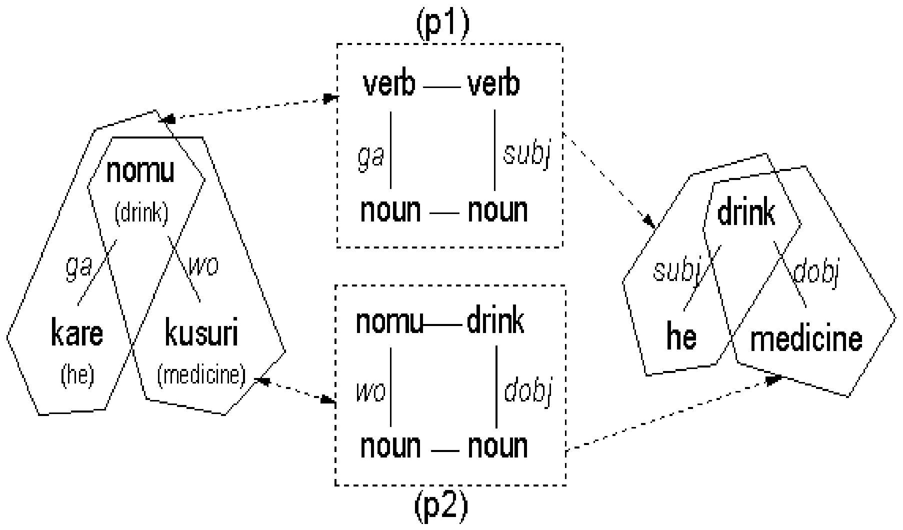


Figure 1: Translation Example by Example-based Translation

dependency structure. A dependency structure as used in this paper is a tree consisting of nodes and links (or arcs), where a node represents a content word, while a link represents a functional word or a relation between content words. For instance, as shown in Figure 2, a preposition "at" is represented as a link in English.

## 2.2 Finding Word Correspondences

The first task for finding structural correspondences is to find word correspondences between the nodes of a source parsed tree and the nodes of a target parsed tree.

Word correspondences are found by consulting a source-to-target translation dictionary. Most words can find a unique translation candidate in a target tree, but there are cases such that there are many translation candidates in a target parsed tree for a source word. Therefore, the main task of finding word correspondences is to determine the most plausible translation word among candidates. We call a pair of a source word and its translation candidate word in a target tree a *word correspondence candidate* denoted by $WC(s,t)$, where $s$ is a source word and $t$ is a target word. If $WC(s,t)$ is a word correspondence candidate such that there is no other $WC$ originating from $s$, then it is called $WA$ word correspondence.

The basic idea to select the most plausible word correspondence candidate is to select a candidate which is near to another word correspondence whose source is also near to a source word in question. Suppose a source word $s$ has multiple candidate translation target words $t_i$ $(i = 1, ..., n)$, that is, there are multiple $WC$s originating from $s$. We denote these multiple word correspondence candidates by $WC(s, t_i)$. For each $WC$ of $s$, this procedure finds the neighbor $WA$ correspondence whose distance to $WC$ is below a threshold. The distance between $WC(s_1, t_1)$ and $WA(s_2, t_2)$ is defined as the distance between $s_1$ and $s_2$ plus the distance between $s_2$ and $t_2$ where a distance between two nodes is defined as the number of nodes in the path whose ends are the two nodes. Among $WC$s of $s$ for which neighbor $WA$ is found, the one with the smallest distance is chosen as the word correspondence of $s$, and $WC$s which are not chosen are invalidated (or deleted). We call a word correspondence found by this procedure $WX$. We use 3 as the distance threshold of the above procedure currently. This procedure is applied to all source nodes which have multiple $WC$s. Figure 3 shows an example of $WX$ word correspondence. In this example, since the Japanese word "ki" has two English translation word candidates "time" and "period," there are two $WC$s ($WC_1$ and $WC_2$). The direct parent node "yuuryo" of "ki" has a $WA$ correspondence ($WA_1$) to "concern," and the direct child node "ikou" has also a $WA$ correspondence ($WA_2$) to "transition." In this case, since the distance between $WC_2$ and $WA_2$ is smaller than the distance between $WC_1$ and $WA_1$, $WC_2$ is changed to a $WX$, and $WC_1$ is adandoned.

In addition to $WX$ correspondences, we consider a special case such that given a word correspondence $W(s,t)$, if $s$ has only one child node which is

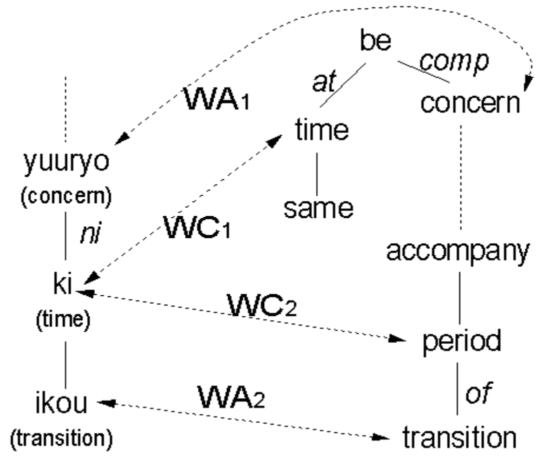


Figure 3: An Example of WX Word Correspondence

a leaf and $t$ has also only one child node which is a leaf, then we construct a new word correspondence called $WS$ from these two leaf nodes. This $WS$ procedure is applied to all word correspondences. Note that this word correspondence is not to select one of candidates, rather it is a new finding of word correspondence by utilizing a special structure. For instance, in Figure 3, if there is a word correspondence between "ki" and "period" and there is no word correspondence between "ikou" and "transition," then $WS(ikou, transition)$ will be found by this procedure.

These $WX$ and $WS$ procedures are continuously applied until no new word correspondences are found.

After applying the above $WX$ and $WS$ procedures, there are some target words $t$ such that $t$ is a destination of a $WC(s,t)$ and there is no other $WC$ whose destination is $t$. In this case, the $WC(s,t)$ correspondence candidate is chosen as a valid word correspondence between $s$ and $t$, and it is called a $WZ$ word correspondence.

We call a source node or a target node of a word correspondence an *anchor node* in what follows.

The above procedures for finding word correspondences are summarized as follows:

```
Find $WC$s by consulting translation dictionary;
Find $WA$s;
while (true) {
    find $WX$s;
    find $WS$s;
    if no new word corresp. is found, then break;
}
find $WZ$s;
```

## 2.3 Finding Phrasal Correspondences

The next step is to find phrasal correspondences based on word correspondences found by procedures described in the previous section. What we would like to retrieve here is a set of phrasal correspondences which covers all elements of a paired dependency trees.

In what follows, we call a portion of a tree which consists of nodes in a path from a node $n_1$ to another node $n_2$ which is a descendant of $n_1$ a *linear tree* denoted by $LT(n_1, n_2)$, and we denote a minimal subtree including specified nodes $n_1, ..., n_x$ by $T(n_1, ..., n_x)$. For instance, in the English tree structure (the right tree) in Figure 4, $LT(technology, science)$ is a rectangular area covering "technology," and "science," and $T(factor, country)$ is a polygonal area covering "factor," "affect," "policy," and "country."

The first step is to find a pair of word correspondences $W_1(s_1, t_1)$ and $W_2(s_2, t_2)$ such that $s_1$ and $s_2$ constructs a linear tree $LT(s_1, s_2)$ and there is no anchor node in the path from $s_1$ to $s_2$ other than $s_1$ and $s_2$, where $W_1$ and $W_2$ denote any type of word correspondences[2] and we assume there is a word correspondence between roots of source and target trees by default. We construct a phrasal correspondence from source nodes in $LT(s_1, s_2)$ and target nodes in $T(t_1, t_2)$, denoted by $P(LT(s_1, s_2), T(t_1, t_2))$. For instance, in Figure 4, $P_{11}$, $P_{12}$, $P_2$, $P_3$ and $P_4$ are source portions of phrasal correspondences found in this step.

The next step checks, for each $P$, if all anchor nodes of word correspondences whose source or target node is included in $P$ are also included in $P$. If a phrasal correspondence satisfies this condition, then it is called *closed*, otherwise it is called *open*. Further, nodes which are not included in the $P$ in question are called *open nodes*. If a $P$ is open, then it is merged with other phrasal correspondences having open nodes of $P$ so that the merged phrasal correspondence becomes closed.

Next, each $P_x$ is checked if there is another $P_y$ which shares any nodes other than anchor nodes with $P_x$. If this is the case, these $P_x$ and $P_y$ are merged into one phrasal correspondence. In Figure 4, phrasal correspondences $P_{11}$ and $P_{12}$ are merged into $P_1$, since their source portions $LT$ $(haikei, koku)$ and $LT$ $(haikei, seisaku)$ share "doukou" which is not an anchor node.

Finally, any path whose nodes other than the root are not included in any $P$s but the root node is included in a $P$ is searched for. This procedure

[2]Since $WC$ is not a word correspondence (it is a candidate of word correspondence), it is not considered here.

is applied to both source and target trees. A path found by this procedure is called an *open path*, and its root node is called a *pivot*. If such an open path is found, it is processed as follows: For each pivot node, (a) if the pivot is not an anchor node, then open paths originating from the pivot is merged into a $P$ having the pivot, (b) if the pivot is an anchor node, then a new phrasal correspondence is created from open paths originating from the anchor nodes of the word correspondence.

In Figure 4, we get finally four phrasal correspondences $P_1$, $P_2$, $P_3$, and $P_4$.

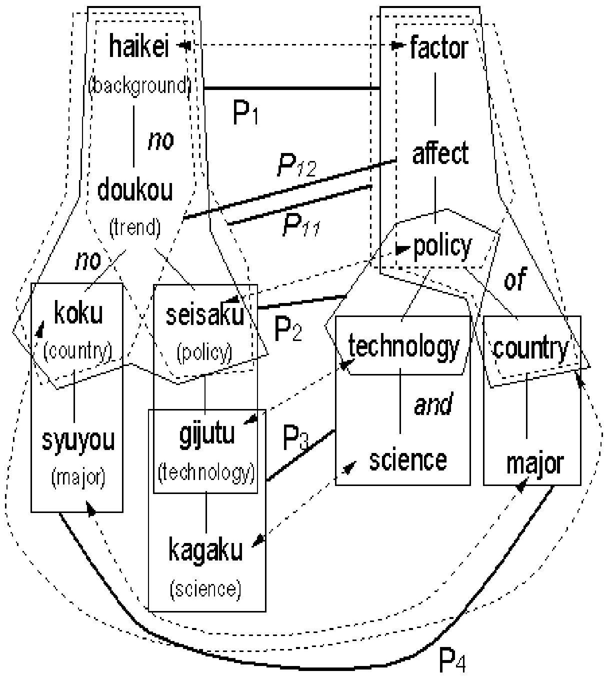


Figure 4: An Example of Finding Phrasal Correspondences

The above procedures for finding phrasal correspondences are summarized as follows:

```
Find initial $P$s;
Merge an open $P_i$ with other $P$s having
  open nodes of $P_i$;
Create new $P$s by merging $P$s
  which have more than 2 common nodes;
Find open path, and
  **if** the pivot is an anchor, **then**
    merge the path to $P$ having the anchor,
  **otherwise** create new $P$ by merging
    all open paths having the pivot;
```

## 3 Experiments

### 3.1 Corpus and Dictionary

We used documents from White Papers on Science and Technology (1994 to 1996) published by the Science and Technology Agency (STA) of the Japanese government. STA published these White Papers in both Japanese and English. The Communications Research Laboratory of the Ministry of Posts and Telecommunication of the Japanese government supplied us with the bilingual corpus which is already roughly aligned. We made a bilingual corpus consisting of parsed dependency structures by using the KNP[2] Japanese parser (developed by Kyoto University) for Japanese sentences and the ESG[5] English parser (developed by IBM Watson Research Center) for English sentences.

We made about 500 sentence pairs, each of which has a one-to-one sentence correspondence, from the raw data of the White Papers, and selected randomly about 130 sentence pairs for experiments. However, since a parser does not always produce correct parse trees, we excluded some sentence pairs which have severe parse errors, and finally got 115 sentence pairs as a test set.

As a translation word dictionary between Japanese and English, we first used J-to-E translation dictionary which has more than 100,000 entries, but we found that there are some word correspondences not covered in this dictionary. Therefore, we merged entries from E-to-J translation dictionary in order to get much broad coverage. The total number of entries are now more than 150,000.

### 3.2 Experimental Results

Table 1 shows the result of experiment for finding word correspondences. A row with ALL in the type column shows the total accuracy of word correspondences and other rows show the accuracy of each type. It is clear that $WA$ correspondences have a very high accuracy. Other word correspondences also have a relatively high accuracy.

Table 2 shows the result of experiments for finding phrasal correspondences. The row with ALL in the type column shows the total accuracy of phrasal correspondences found by the proposed procedure. This accuracy level is not promising and it is not useful for later processes since it needs human checking and correction. Therefore, we subcategorize each phrasal correspondences, and check the accuracy for each subcategory.

We consider the following subcategories for phrasal correspondences:

- MIN ... The minimal phrasal correspondence, that is, $P(LT(s_1, s_2), LT(t_1, t_2))$ such that there

| type | num. of found corresp. | num. of correct corresp. | success ratio (%) |
|---|---|---|---|
| ALL | 771 | 745 | 96.63 |
| WA | 612 | 600 | 98.03 |
| WX | 131 | 118 | 90.07 |
| WS | 13 | 12 | 92.3 |
| WZ | 15 | 15 | 100 |

Table 1: Experimental Result of Word Correspondences

are word correspondences $W(s_1, t_1)$ and $W(s_2, t_2)$, $s_2$ is a direct child of $s_1$ and $t_2$ is a direct child of $t_1$.

- LTX ... $P(LT(s_1, s_2), LT(t_1, t_2))$ such that all nodes other than $s_2$ and $t_2$ have only one child node.
- LTY ... $P(LT(s_1, s_2), LT(t_1, t_2))$ such that all nodes other than $s_1, s_2, t_1$ and $t_2$ have only one child node.

LTX is a special case of LTY, since $s_1$ and $t_1$ of LTX must have only one child node, on the other hand, ones of LTY may have more than two child nodes. A subcategory test for a phrasal correspondence is done in the above order. Examples of these subcategories are shown in Fig 5.

The result of these subcategories are also shown in Table 2. Subcategories MIN and LTX have very high accuracy and this result is very promising, since we can avoid manual checking for these phrasal correspondences, or we would check only these types of phrasal correspondences manually and discard other types.

As stated earlier, since we removed only sentences with severe parsing errors from the test set, please note that the above numbers of experimental results are calculated for a bilingual parsed corpus including parsing errors.

## 4 Discussion

There have been some studies on structural alignment of bilingual texts such as [1, 4, 13, 3, 6]. Our work is similar to these previous studies at the conceptual level, but different in some aspects. [1] reported a method for extracting translation templates by CKY parsing of bilingual sentences. This work is to get phrase-structure level phrasal correspondences, but our work is to get dependency-structure level phrasal correspondences. [4] proposed a method for extracting structural matching (pairs of dependency trees) by calculating matching similarities of two dependency structures. Their work focuses on the parsing ambiguity resolution by calculating structural matching. Further, [3, 6] proposed structural alignment of dependency structures. Their work assumed that least common ancestors of each fragment of a structural correspondence are preserved, but our work does not have such structural restriction. [13] is different to others in that it tries to find phrasal correspondences by comparing a MT result and its manual correction.

In addition to these differences, the main difference is to find classes (or categories) of phrasal correspondences which have high accuracy. In general, since bilingual structural alignment is very complicated and difficult task, it is very hard to get more than 90% accuracy in total. If we get only such an accuracy rate, the result is not useful, since we need manual checks for the all correspondences retrieved. But, if we can get some classes of phrasal correspondence with, for instance, more than 90% accuracy rate, then we can reduce manual checking for phrasal correspondences in such classes, and this reduces the development cost of translation patterns used in later corpus-based translation process. As shown in the previous section, we could find that all classes of word correspondences and two subclasses of phrasal correspondences are more than 90% accurate.

When actually using this automatically retrieved structural correspondence data, we must consider how to manually correct the incomplete parts and how to reuse manual correction data if the parser results are changed.

As for the former issue, we need an easy-to-use tool to modify correspondences to reduce the cost of manual operation. We have developed a GUI tool as shown in Figure 6. In this figure, the bottom half presents a pair of source and target dependency structures with word correspondences (solid lines) and phrasal correspondences (sequences of shaded circles). You can easily correct correspondences by looking at this graphical presentation.

As for the latter issue, we must develop methods for reusing the manual correction data as much as possible even if the parser outputs are changed. We have developed a tool for attaching phrasal correspondences by using existing phrasal correspondence data. This is implemented as follows: Each phrasal correspondence is assigned a signature which is a pair of source and target sentences, each of which has bracketed segments which are included in the phrasal correspondence. For instance,

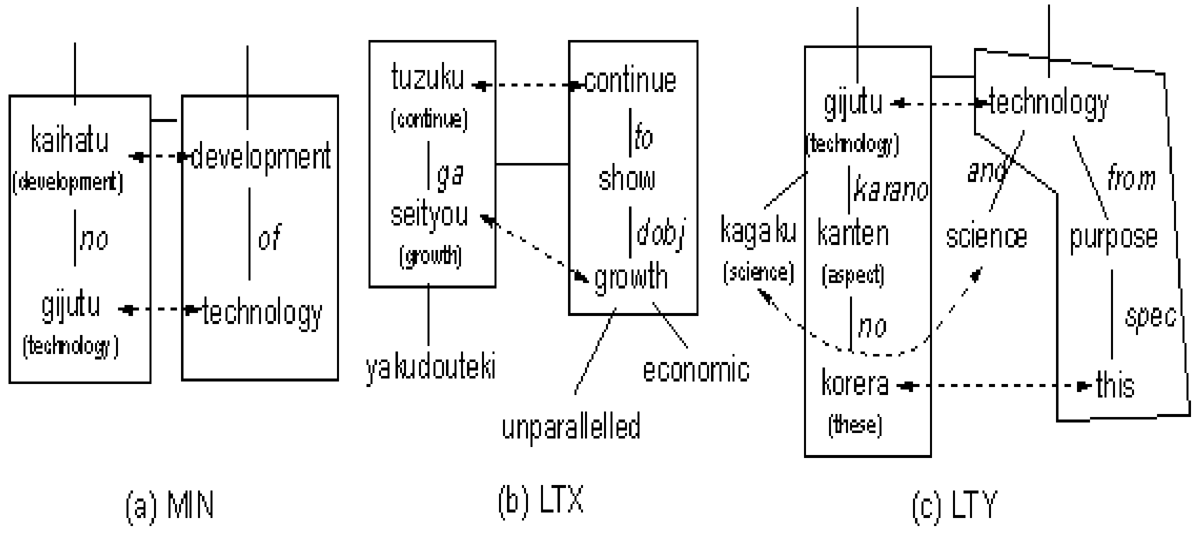


Figure 5: Examples of Categories of Phrasal Correspondences

| type | A: num. of found corresp. | B: num. of correct corresp. | C: success ratio B/A (%) | D: num. of nodes covered by A | E: num. of nodes covered by B | F: success ratio E/D (%) |
|---|---|---|---|---|---|---|
| ALL | 678 | 431 | 63.56 | 7248 | 4278 | 59.02 |
| MIN | 223 | 215 | 96.41 | 1234 | 1194 | 96.76 |
| LTX | 17 | 17 | 100 | 153 | 153 | 100 |
| LTY | 27 | 20 | 74.07 | 253 | 191 | 75.49 |

Table 2: Experimental Result of Phrasal Correspondences

the following signature is made for a phrasal correspondence (c) in Figure 5:

⟨*sig*⟩
... [korera no kanten karano] kagaku [gijutu] ...
... science and [technology from this purpose] ...
⟨/*sig*⟩

In the above example, segments between '[' and ']' represent a phrasal correspondence.

If new parsed dependency structures for a sentence pair is given, for each phrasal correspondence signature of the sentence pair, nodes in the structures which are inside brackets of the signature are marked, and if there is a minimal subtree consisting of only marked nodes, then a phrasal correspondence is reconstructed from the phrasal correspondence signature. By using this tool, we can efficiently reuse the manual efforts as much as possible even if parsers are updated.

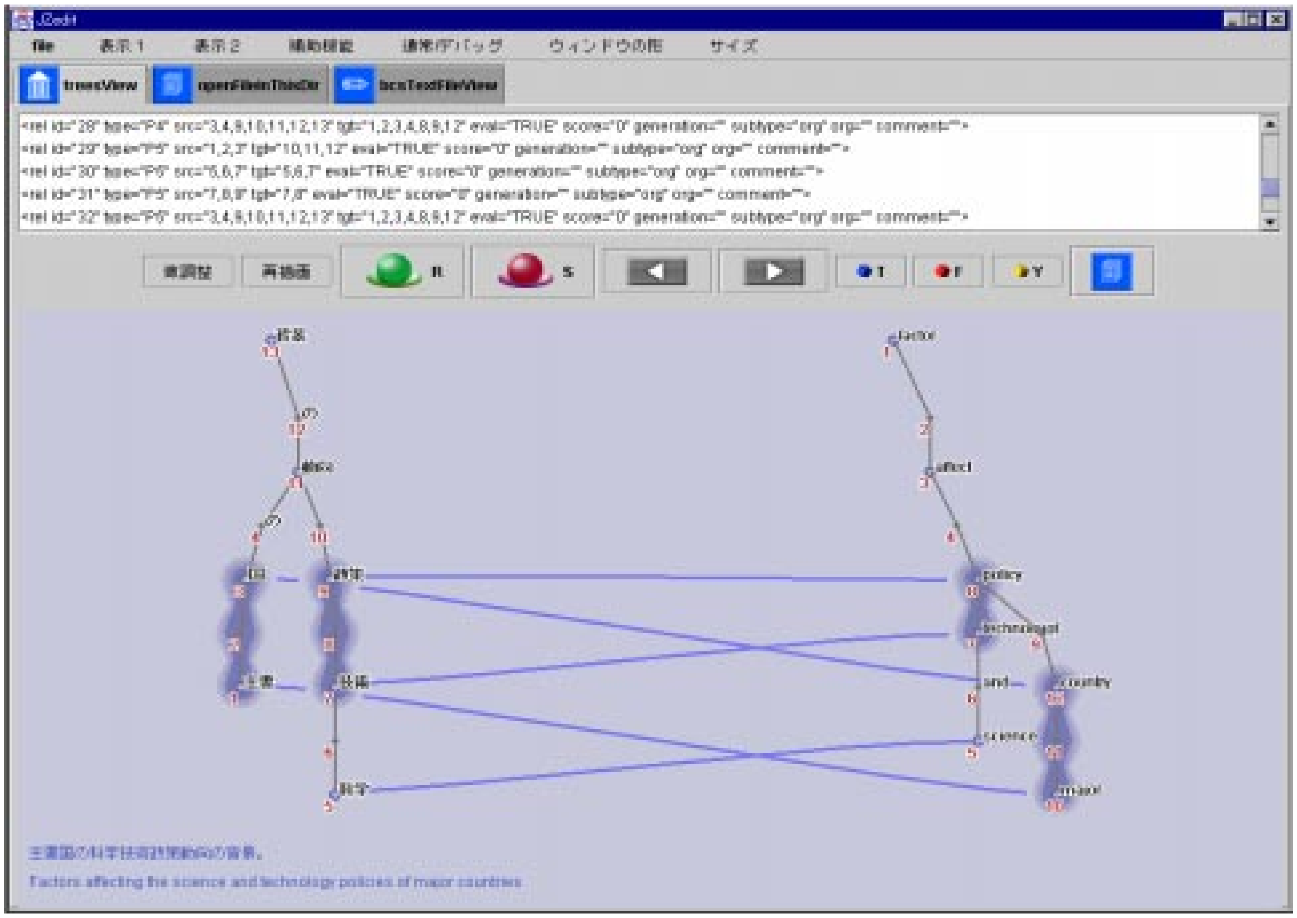
## 5 Conclusion

In this paper, we have proposed methods for finding structural correspondences (word correspondences and phrasal correspondences) of bilingual parsed corpus. Further, we showed that the precision of word correspondences and some categories of phrasal correspondences found by our methods are highly accurate, and these correspondences can reduce the cost of translation pattern accumulation.

In addition to these results, we showed a GUI tool for manual correction and a tool for reusing previous correspondence data.

As future directions, we will find more subclasses with high accuracy to reduce the cost for translation pattern preparation.

We believe that these methods and tools can accelerate the collection of a large set of translation patterns and the development of a corpus-based translation system.



```
<rel id="28" type="P4" src="3,4,9,10,11,12,13" tgt="1,2,3,4,8,9,12" eval="TRUE" score="0" generation="" subtype="org" org="" comment="">
<rel id="29" type="P5" src="1,2,3" tgt="10,11,12" eval="TRUE" score="0" generation="" subtype="org" org="" comment="">
<rel id="30" type="P6" src="5,6,7" tgt="5,6,7" eval="TRUE" score="0" generation="" subtype="org" org="" comment="">
<rel id="31" type="P5" src="7,8,9" tgt="7,8" eval="TRUE" score="0" generation="" subtype="org" org="" comment="">
<rel id="32" type="P6" src="3,4,9,10,11,12,13" tgt="1,2,3,4,8,9,12" eval="TRUE" score="0" generation="" subtype="org" org="" comment="">
```

主要国の科学技術政策動向の背景。

Factors affecting the science and technology policies of major countries

Figure 6: An GUI tool for presenting/manipulating structural correspondences

## References

[1] Kaji, H., Kida, Y., and Morimoto, Y., "Learning Translation Templates from Bilingual Texts," Proc. of Coling 92, pp. 672–678, 1992.

[2] Kurohashi, S., and Nagao, M., "A Syntactic Analysis Method of Long Japanese Sentences based on the Detection of Conjunctive Structures," Computational Linguistics, Vol. 20, No. 4, 1994.

[3] Grishman, R., "Iterative Alignment of Syntactic Structures for a Bilingual Corpus," Proc. of 2nd Workshop for Very Large Corpora, pp. 57–68, 1994.

[4] Matsumoto, Y., Ishimoto, H., and Utsuro, T., "Structural Matching of Parallel Texts," Proc. of the 31st of ACL, pp. 23–30, 1993.

[5] McCord, C. M., "Slot Grammars," Computational Linguistics, Vol. 6, pp. 31–43, 1980.

[6] Meyers, A., Yanharber, R., and Grishman, R., "Alignment of Shared Forests for Bilingual Corpora," Proc. of the 16th of COLING, pp. 460–465, June 1996.

[7] Nagao, M., "A Framework of a Mechanical Translation between Japanese and English by Analogy Principle," Elithorn, A. and Banerji, R. (eds.) : *Artificial and Human Intelligence,* NATO 1984.

[8] Sato, S., and Nagao, M. "Toward Memory-based Translation," Proc. of 13th COLING, August 1990.

[9] Sumita, E., Iida, H., and Kohyama, H. "Translating with Examples: A New Approach to Machine Translation," Proc. of Info Japan 90, 1990.

[10] Takeda, K., "Pattern-Based Context-Free Grammars for Machine Translation," Proc. of 34th ACL, pp. 144–151, June 1996.

[11] Takeda, K., "Pattern-Based Machine Translation," Proc. of 16th COLING, Vol. 2, pp. 1155–1158, August 1996.

[12] Watanabe, H. "A Similarity-Driven Transfer System," Proc. of the 14th COLING, Vol. 2, pp. 770-776, 1992.

[13] Watanabe, H. "A Method for Extracting Translation Patterns from Translation Examples," Proc. of the 5th Int. Conf. on Theoretical and Methodological Issues in Machine Translation, pp. 292-301, 1993.

[14] Watanabe, H., and Takeda, K., "A Pattern-based Machine Translation System Extended by Example-based Processing," Proc. of the 36th ACL & 17th COLING, Vol. 2, pp. 1369-1373, 1998.