Aligning and Matching of English-Chinese Bilingual Texts of CNS News

Donghua XU
Department of Computer Science
School of Computing
National University of Singapore, Singapore 119260
xudh@pemail.net

Chew Lim TAN
Department of Computer Science
School of Computing
National University of Singapore, Singapore 119260
tancl@comp.nus.edu.sg

Abstract: This paper presents a project to align and match English-Chinese bilingual news files downloaded from China News Service's website. The work involves the alignment of bilingual texts at the sentence and clause levels. It addition, the work also requires matching of files as the English and Chinese news files downloaded from the web do not come in the same sequential order. These news files have their own characteristics and, furthermore, the issue of file-matching has its unique difficulties apart from the known problems of alignment work previously reported in the literature. To align the news files we combine the criteria of "anchor"(i.e., unambiguous corresponding text elements) and sentence length. We employ Dynamic Programming to first align at the paragraph level, then align at the sentence-clause level. The precision and recall of the alignment are satisfactory for free translation texts. To match English files and Chinese files, we only make use of "anchor". In file matching we encounter a "collision" problem due to contending matching candidates, and propose a recursive splitting algorithm to resolve the problem. We allow human intervention to improve the precision of matching, and succeeded in achieving a 100% precision with a fairly small amount of manual effort. Finally, to determine the various parameters used in aligning and matching, we utilize a Genetic Algorithm software package to obtain their optimized values.

Key words: text alignment, file matching, dynamic programming, genetic algorithm, news files

1. Introduction

While the traditional rule-based and knowledge-based machine translation (MT) approaches have achieved considerable progress over the past 50 years, two new MT techniques have in recent years emerged, taking advantage of the rapidly improving computing power. The two emergent techniques are Statistical MT (Brown *et al*. 1990) and Example-Based MT (Nagao 1984, Sato 1991). Both approaches require a huge quantity of translation examples from large existing translation corpora. Automatic alignment is an answer to such a need.

Several methods of automatic alignment have already been reported in the literature. Among them, we find basically three different approaches, namely, the lexical approach, the length-based statistical approach, and the combination of them. The lexical approach as in the works of Kay and Roscheisen(1991, 1993) and of Hwang and Nagao(1994), looks for relationships between lexical contents of the bilingual texts in order to find alignment pairs. The statistical approach as demonstrated in the methods by Brown *et al.*(1991) and by Gale and Church(1991, 1993), on the other hand, uses statistical correlation between sentence lengths of the bilingual texts as the basis of matching. Finally, other works such as those presented by Simard *et al.*(1992), Wu(1994), Tan & Nagao(1995) and Langlais(1997) combine both approaches to align texts.[1]

In recent works, pure length-based statistical approach has been gradually abandoned. Melamed(1996a) and Haruno and Yamazaki(1996) both used pure word correspondence, Chen & Chen(1995) used part-of-speech correspondence, while Ker and Chang(1997) even made use of word class correspondence. However these methods require some amount of lexical analysis if they are applied to eastern languages. For instance, for Chinese, a word-segmentation process should be run on the Chinese corpus before any lexical correspondences between Chinese and the other language can even be found. And this lexical analysis step is usually quite computational expensive.

As for the specific algorithm, most of the works use dynamic programming to produce an optimal aligning result, given that there are enough global correspondence information. We have mentioned that there are two kinds of such correspondence, one being length correspondence, the other lexical correspondence. Length correspondences reside in the parallel corpora themselves and can be determined by the program easily. They are thus globally and readily available. On the other hand, lexical correspondences are sometimes not in abundance or may not be globally available, in which case using dynamic programming based on lexical correspondences alone will fail to attain an accurate alignment. To solve the case of insufficient lexical correspondences, Kay and Roscheisen(1993) and Haruno and Yamazaki(1996) used a relaxation method of repetitive alignment to obtain more lexical information in each iteration of processing. On the other hand, Melamed(1996a) used a special geometric method to localize an aligning process, so that his greedy algorithm could attain an optimized mapping result at every local segment of the parallel corpus, and eventually gain a global mapping result.

Chen & Chen(1995) tested their part-of-speech correspondences using three standard optimizing algorithms (Dynamic Programming, Divide and Conquer, and Simulated Annealing) and found that Dynamic Programming and Simulated Annealing have much higher precision and recall than Divide and Conquer.

In our present work, we study the alignment between the English-Chinese bilingual news reports from China News Service (CNS). We will use both length and lexical correspondences. The lexical information we use is somewhat different from those used in previous works. We use "anchors" which are much less ambiguous than

[1] Comparisons between *symbolic* and *statistical* approaches in other fields of NLP have been vividly illustrated by Klavans and Resnik(1996).

normal words and can be picked out by the program without thorough word-segmentation or other such kind of deep analysis of the corpora. Because of the distinctive meaning of these anchors, they provide reliable correspondence for alignment. Moreover, because of their ease in identification, computational cost spent in picking them out is very small. Length correspondence itself can be directly measured from both texts, and combining with anchor correspondence, we can achieve a clause level alignment in some places where anchors distribute evenly in the parallel texts.

Our alignment work includes two levels of alignment. The first is to align paragraphs for each pair of English and Chinese files, and the second is to align clauses for each pair of aligned English and Chinese paragraphs. This reduces the recursion depth of dynamic programming and makes the algorithm faster than directly aligning at the clause level for the entire files.

Beyond the alignment work, we also address a problem of matching among bilingual files. Here "matching" means the process of deciding which English file matches with which Chinese file among a group of English files and a group of Chinese files. This work is unique because the problem has its own characteristics and we have to use a special strategy to solve the problem.

The next section will discuss features of the English-Chinese bilingual CNS news. Section 3 will address the alignment process, while section 4 will address the matching process. Section 5 will give an indepth discussion of existing works with the view to comparing and constrasting our method with others. Section 6 will introduce how we determine the values of the parameters defined in Section 3 and 4 using a genetic algorithm. Section 7 presents our experimental results of both alignment and matching. Section 8 will conclude this paper with a short summary and the directions of our future work.

2. Features of CNS News

With the recent rapid development of the Internet and the World Wide Web (WWW), we can now easily retrieve a large volume of linguistic resources that were not so easy to get in computer format before. The news reports released by the news agencies and other organizations will prove to be very valuable for researchers on computational linguistics, by virtue of the large quantity of such news coming in everyday and their relatively formal expressions.

China News Service is one of the two major news agencies in China. It offers news from China in both Chinese and English on the WWW. On their Web page, individual pieces of news in one language are contained in separate files. The files are fairly small (usually 1K or 2K per file). Alignment between corresponding files will not require too deep a recursion of Dynamic Programming. This seems to be an advantage for our automatic alignment. However, we need to first match a pair of files before we can align the bilingual texts from the file pair, and finding a pair of matched files from a large number of downloaded files is not a simple task. We will discuss this in Section 4.

Each news file consists of a number of paragraphs separated by clear delimiters. Each paragraph is composed of some complete sentences which end with "。！？" in Chinese and ".!?" in English. And each sentence contains a number of clauses divided by "，；：" both in Chinese and in English. Of course in English text "." and "," do not necessarily mean the end of a sentence and a clause respectively. The former might be a decimal point of a number or a mark of abbreviation, while the latter might be a delimiter at every three consecutive digits of a number. Fortunately, in the news reports we are studying, all the digits of a complete number are immediately adjacent. Thus we can simply check whether the characters before and after "."(",") are both digits to decide whether "."(",") is a part of a number. And the abbreviations with a "." such as "U.S." or "Mr." can be picked out by a table look-up. For simplicity of discussion, we will call the end punctuation of sentence (i.e. "。.!?") "period", and that of clause (i.e. ",;:") "comma" from now on.

After extensive observations, we determine that there are two kinds of match between paragraphs in the two languages, i.e. 1↔*n* and *n*↔1. The former means *one* English paragraph matches with *n* successive Chinese paragraphs, and the latter vice-versa. It depends on the human translator. Since the corpora are translated from Chinese into English, sometimes the translator would translate a Chinese paragraph into several English paragraphs, while sometimes he may combine several consecutive Chinese paragraphs when translating them into one English paragraph. (1↔1 is a special case of both cases above where one English paragraph matches with one Chinese paragraph). Moreover, we have found that some of the Chinese paragraphs that might be less important are not translated at all, so we define 0↔*n* as another kind of match, i.e. *zero* English paragraph matches with *n* consecutive Chinese paragraphs.

A feature with the English and Chinese language pair is their differences in sentential structures. The differences are believed to be wider than the corpora pairs involved in the earlier work. The alignment methods by Kay and Roscheisen(1993), Brown et al.(1991), and Gale and Church(1991, 1993) dealt with English, French and German, while Hwang and Nagao(1994) handled Japanese and Korean texts. The languages studied in these works, as pointed out in Wu(1994) and Tan and Nagao(1995), share cognates and similar grammatical structures. In Wu(1994), the corpora involved are Hong Kong Hansard, i.e. the record and the translation (between Cantonese and English) of the discourse on the Legislative Council. The translation reported in Wu(1994) tends to be literal. Thus despite the great difference between the two languages, there is a high rate of 1↔1 match at nearly 90% when aligning at the sentence level (Wu 1994). In our case however, through a rough estimation, we find that only about 50% of the sentences in CNS news can be matched as 1↔1. That means, if we just match the texts at the sentence level (i.e. not allowing breaking up of a sentence for alignment), very often we would have to pair quite a number of sentences en masse between both texts. This is also partly due to the fact that periods may be used differently in both languages to end sentences.

Therefore, what we would like to propose is to allow a sentence in one text to be matched with part of a sentence of the other text, if possible, in order to create a finer correspondence pair. We observe that a period in either text indicates a self-contained entity and may be used to find a break point in the other text even if there is no period but a comma at that point. The sentence which ends with that period may then be matched with the sequence of clauses ended with that break point in the other text.

Matching clauses will be more difficult and error-prone than simple sentence matching. Fortunately, another useful feature of these news reports is the obvious anchors in the texts, e.g. numerical digits (“二十三” in Chinese and “23” or “twenty-three” in English), or places (“北京” in Chinese and “Beijing” in English), or specific dates (“十月一日” in Chinese and “October 1” in English), etc. There are plenty of these anchors in most of the news. The meaning of these anchors would be much less ambiguous than normal words, and so we can easily find the match of these anchors in both texts and count the occurrence of these anchors, making it an important element to decide how much the two strings of the texts are likely to match with each other.

Furthermore, we find that by evaluating a “splitting” function to be discussed in section 3.4, we can even successfully achieve clause-to-clause match in some parts of the texts where matched anchors are distributed densely and evenly.

We divide the whole automatic alignment process into four steps:

1) Preprocess the English & Chinese corpora
2) Match the anchors in the English corpora with those in Chinese corpora
3) Align between paragraphs in English & Chinese corpora
4) Align between clauses of each pair of aligned paragraphs

These four steps are described one by one in the next section.

3. The Alignment Process

3.1 Preprocessing

In this step, the program reads in a pair of English and Chinese files and analyzes them. The head and tail and some other noisy information of each news file are filtered out, since they are not useful to the alignment task *per se*. A data structure is constructed for the paragraph-sentence-clause hierarchical organization of each corpus. Another important task of this step is to analyze the anchors in both corpora as discussed below.

First of all, the way to express a number is quite different in the two languages. For example, in Chinese the numbe “一千九百七十六亿六千万” would be changed into “197.66 billion” in English translation. Another example is “七成半” (in

Chinese “成” means 10%) which has the same meaning a “百分之七十五” and “七十五个百分点” and “四分之三”, and might appear as “75 per cent” or “three quarters” in English texts. We have to convert such a set of digit characters into a real number in the program considering all such cases in both texts, so that we can compare them later. A similar process is done to date-anchors such as “June, 1997” or “7 1997”. In other words, we match number and date anchors through interlingual representations.

Secondly, since these news are from China, they are concerned with events related to China. Thus places in China and countries in the world appear frequently in the news. We construct a table which contains the English and Chinese names of the countries in the world and provinces and big cities in China. We sort this table and use Binary Search to look it up when deciding whether a series of English or Chinese characters refer to a place. This table has now about 200 items, and thus with Binary Search, table look up can be done within 8 times comparison.

For each news file, we construct an Anchor Table to record all the anchors.

3.2 Matching Anchors

Having determined all the anchors in both corpora, we use each anchor in the English Anchor Table to find a match in the Chinese Anchor Table. Let MA represent a pair of matched anchors. There are some interesting details in anchor matching worth mentioning here.

First, we must give different weights to different types of anchors. An exact number such as “192.6” deserves a larger weight than a normal place name. This is because when translation is free, names of places sometimes are used in different locations in the two languages, while exact numbers usually only appear in the corresponding sense group. So we define several types of anchors and specify the weight of each type of matched anchors $W(MA)$, for example:

$$W(MA) = \begin{cases} W_N & \text{if } MA \text{ is an exact number} \\ W_P & \text{if } MA \text{ is a place} \\ W_D & \text{if } MA \text{ is a date} \end{cases} \tag{1}$$

Here the values of the three constants W_N, W_P and W_D have only relative meaning among themselves. So we specify the first one W_N = 100. The other two constants, W_P and W_D together with some other constants in the following passages, are obtained by a bootstrapping Genetic Algorithm process. We will explain this process in Section 6.

Secondly, when matching the anchors, we may find that an anchor repeats several times in a corpus. For example, if there are thr“北京” in the Chinese corpus and two “Beijing” in the English corpus, it is difficult for the program to directly decide which “北京” matches with which “Beijing”. Therefore we adopt a special chain to record the repeated anchors, and allow a “北京” to match with all “Beijing” as a whole. And to prevent the repeated anchors from messing up the matching process too much, we should give this kind of match a penalty by decreasing its weight. Thus we define the Factor of Repetition $F_R(MA)$:

$$F_R(MA) = \begin{cases} f_R & \text{if } MA \text{ appears more than once} \\ 1 & \text{if } MA \text{ appears only once} \end{cases} \tag{2}$$

In addition, because the translation is free, sometimes an anchor in one language would be changed into an approximate form in another language. E.g. “一百九十六” in Chinese which means 196 may be changed int “nearly 200” in English. So if an anchor in English Anchor Table cannot find an exact match in the Chinese Anchor Table, we allow it to search again for an approximate match. Of course, an approximate match should raise another penalty. We define the Factor of Approximation $F_A(MA)$ as :

$$F_A(MA) = \begin{cases} f_A & \text{if } MA \text{ is an approximate match} \\ 1 & \text{if } MA \text{ is an exact match} \end{cases} \tag{3}$$

Therefore, the value of a pair of matched anchors is defined as:

$$V(MA) = W(MA) \cdot F_R(MA) \cdot F_A(MA) \tag{4}$$

For the sake of simplicity and clarity, we do not include some other processing details in this equation. For example, in Chines “一” and “二” which mean 1 and 2, respectively, are often used as a part of an idiom or word, but not necessary an actual number. At the same time, “one” in English also is not always used as an actual number. So we would not say a match betwee “一” and “one” is very reliable. Thus in this case the match should get a penalty too.

3.3 Aligning Paragraphs

After matching the anchors, we can proceed to align the two texts. In this step we treat a paragraph as the basic unit of alignment and try to match English paragraph(s) with corresponding Chinese paragraph(s).

Let P_E, P_C be the total number of paragraphs, respectively, in the English corpus and the Chinese corpus. Let $match(u,v)$ represent the total cost of matching a part of the English corpus from the u th paragraph to the end of the corpus with a part of the Chinese corpus from the v th paragraph to the end of the corpus, where $u = 1,2,\cdots,P_E$, $v = 1,2,\cdots,P_C$. In our program, we obtain $match(u,v)$ as follows:

$$match(u,v) = \min\{M(u,v), match(u,v+1)\} \tag{5}$$

where $match(u,v+1)$ is the cost of ignoring the vth Chinese paragraph. We have mentioned that sometimes certain Chinese paragraphs are not translated into English at all. Thus besides $M(u,v)$ (i.e. the real cost of matching the English text from the uth English paragraph to the end with the Chinese text from the vth Chinese paragraph to the end), we also compute $match(u,v+1)$ and compare the two costs to choose the smaller one. If $match(u,v+1)$ is smaller, we can then determine that the vth Chinese paragraph matches with nothing, i.e. it is not translated. In this case, however, it does not necessarily mean that the English text from the uth English paragraph to the end matches with the Chinese text from the $(v+1)$th paragraph to the end. This is because when computing $match(u,v+1)$, through a similar process, we might find that $(v+1)$th Chinese paragraph also matches with nothing. In this way, we can find a match of $0 \Leftrightarrow n$, which means that successive n Chinese paragraphs match with nothing.

Since the news files are translated from Chinese into English, we assume that each English paragraph must have a "source" in Chinese. Therefore when there are still some English paragraphs not matched, we would not allow all the remaining Chinese paragraphs to match with nothing. Since the recursion works forward to the end of the corpora, when checking the terminating condition of the recursion at the end of each corpus, the program will follow different routines according to different conditions as follows:

1) If it exceeds the end of the Chinese corpus but does not exceed that of the English corpus, i.e. there are some English paragraphs still not matched, this match must be a wrong match. So the program will return an error flag to the upper level of recursion indicating that this match is invalid.

2) Otherwise, i.e. if it only exceeds the end of the English corpus, no matter whether or not it exceeds that of the Chinese corpus, this match may be regarded as a valid match. So the program will return 0 as the lowest level cost to the upper level of recursion.

Returning to Eq.(5), $M(u,v)$ is the actual cost of matching the English text from the uth paragraph to the end with the Chinese text from vth paragraph to the end. Using Dynamic Programming, it is to achieve its minimum through a recursive process as follows:

$$M(u,v) = \min_{i=1}^{P_E-u} \min_{j=1}^{\beta(\mathrm{i})} \{match(u+i, v+j) + dist(u,i,v,j)\} \tag{6}$$

where :

$$\beta(i) = \begin{cases} P_C - v & when\ i = 1 \\ 1 & otherwise \end{cases} \quad (7)$$

$$\begin{aligned} dist(u,i,v,j) = {} & f_l \cdot LD(u,i,v,j) \\ & - f_a \cdot AV(u,i,v,j) \\ & + f_{mp} \cdot MP(i,j) \end{aligned} \quad (8)$$

Eq.(7) shows that at the level of paragraph matching, only $1 \Leftrightarrow j$ and $i \Leftrightarrow 1$ matches are allowed. We do not place a specific limit on either i or j, instead we allow one English paragraph to match with up to $P_C - v$ Chinese paragraphs (i.e., from j th paragraph to the end of Chinese Corpus), and vice versa.

In Eqs.(6) and (8), $dist(u,i,v,j)$ is the distance between the Chinese text from u th to $(u+i-1)$ th paragraph and the English text from v th to $(v+j-1)$ th paragraph. The larger the distance, the less likely these two parts of the bilingual texts are matched. In Eq.(6), $match(u+i, v+j)$ is actually the sum of all the distances of the most likely matched pairs in the remaining paragraphs. Thus we choose i and j that can make the value of $match(u+i, v+j) + dist(u,i,v,j)$ to be the smallest, and record it as the most likely match.

As shown in Eq.(8), $dist(u,i,v,j)$ is contributed by three functions, and the factors f_l, f_a, f_{mp} are constant numbers that can be determined through a Genetic Algorithm process that will be explained in Section 6. The first function $LD(u,i,v,j)$ is the length distance between this part of the English corpus and the Chinese corpus. As in the previous works, we assume that each Chinese character is responsible for generating some number of English characters. After estimating some sample corpora, we obtained the expectation of this number $\mathrm{r} = 2.61$, and the variance $\sigma^2 = 0.310$. Therefore we can compute $LD(u,i,v,j)$ by using a gaussian variable(Gale and Church 1991, 1993; Wu 1994):

$$LD(u,i,v,j) = \frac{l_e - l_c \cdot r}{\sqrt{l_c \sigma^2}} \quad (9)$$

where l_e and l_c are the length (measured by the number of characters) of the corresponding parts from the English and Chinese corpora, respectively.

The second function in Eq.(8), $AV(u,i,v,j)$, is the sum of the values of matched anchors in both corpora. A negative sign is also attached before it as this function is to strengthen the match between the two strings. We define:

$$AV(u,i,v,j) = \sum_{MA} V(MA) \quad (10)$$

where MA refers to every pair of matched anchors in these two parts of the English and Chinese texts.

Then comes the third function in Eq.(8), $MP(i, j)$, which is a penalty for a match involving multiple paragraphs. We know that the first function $LD(u,i,v,j)$ has a nice standard normal distribution. But the second function $AV(u,i,v,j)$ is monotonous, i.e. the more paragraphs involved, the larger $AV(u,i,v,j)$ would be. Sometimes when there are too many repeated anchors in consecutive paragraphs, it may erroneously offset the normal distribution of $LD(u,i,v,j)$. Thus we use the value of $AV(u,i,v,j)$ to define the penalty for multiple-paragraph matching:

$$MP(i, j) = AV(u,i,v,j) \cdot (i + j - 2) \tag{11}$$

As a matter of fact, $MP(i, j)$ serves to fine tune $dist(u,i,v,j)$, and therefore its factor f_{mp} should be much smaller than f_a .

3.4 Aligning Sentences and Clauses

After aligning the paragraphs, the next task is to align the sentences and clauses between each pair of the aligned paragraphs. For the sake of simplicity, we consider only cases where one English paragraph matches with one Chinese paragraph from now on. The following discussion may be easily extended to other cases involving multiple-paragraph matching.

Due to differences in sentence structures in both languages, punctuation marks are sometimes used at different locations to end sentences in the bilingual texts. Thus there may be no simple matching of one sentence in one text with another in the other. Instead, the matching boundary may be cutting a sentence in the other text. Thus, if possible, we would like to do matching at the clause level. We therefore regard a clause as the basic unit of alignment in this step. Let C_E , C_C represent the total number of clauses in the English paragraph and Chinese paragraph, respectively. Let $match(u,v)$ represent the total cost of matching from the uth clause to the end of the English paragraph with a string of Chinese text from the vth clause to the end of the Chinese paragraph, where $u = 1,2,\cdots,C_E$, $v = 1,2,\cdots,C_C$. In our program, we obtain $match(u,v)$ as follows:

$$match(u,v) = \min_{i=1}^{C_E-u} \min_{j=1}^{C_C-v} \left\{ match(u+i, v+j) + dist(u,i,v,j) \right\} \tag{12}$$

where:

$$
\begin{aligned}
dist(u,i,v,j) = & f_l \cdot LD(u,i,v,j) \\
& + f_{ms} \cdot MS(u,i,v,j) \\
& - f_a \cdot AVS(u,i,v,j) \\
& - f_{ws} \cdot WS(u,i,v,j)
\end{aligned} \tag{13}
$$

As we can see, there are some differences between Eqs.(12)-(13) in this step and Eqs.(5)-(8) in the preceding step. The first function $LD(u,i,v,j)$ in Eq.(13) is similar to that in the preceding step. The other functions which differ from each other in both steps are explained below.

Allowing matching at the clause level does not necessarily mean that every clause will be matched with another clause in the other text. This is because corresponding clauses in both texts need not be in the same order. Thus, when there are no simple matches between clauses, we allow a sequence of clauses in one text to match with a sequence of clauses in the other text, i.e. we allow $m \Leftrightarrow n$ ($1 \le m < C_E - u,\ 1 \le n < C_C - v$) matches. (More will be explained later as illustrated in Figs. 1 to 3). Moreover, as the corpora are free translation, we have found in some cases where one or two Chinese clauses are not literally translated in the English translation. However, permitting a $0 \Leftrightarrow n$ match could sometimes render erroneous matching involving clauses surrounding the non-translated clauses. Thus we do not allow $0 \Leftrightarrow n$ matching but instead allow $m \Leftrightarrow n$ alignment which may contain non-translated clauses in the matched sequences of clauses from both texts.

While matching of groups of clauses are allowed, we will still give an advantage to matching of whole sentences wherever possible. Therefore in Eq.(13), we provide a bonus for Whole Sentence match, i.e. $WS(u,i,v,j)$. Furthermore, we must adopt some measures to prevent a match from involving too many clauses and sentences in either language. Otherwise the final result would usually be only one match, i.e., the whole English paragraph matching with the whole Chinese paragraph. Thus we give a penalty to Multiple Sentence match, i.e. $MS(u,i,v,j)$. $MS(u,i,v,j)$ will prevent a match containing inter-sentence clauses, i.e., preventing clauses in different sentences from appearing in the same matching group. In other words, we try to discourage matching with such a group of clauses that ends with a comma but has a period inside.

Of course we also make use of the anchors in this step. But the function regarding the anchors, $AVS(u,i,v,j)$, is quite different from $AV(u,i,v,j)$ in the preceding section. We define $AVS(u,i,v,j)$ as:

$$
AVS(u,i,v,j) = S(u,i,v,j) \cdot \sum_{MA} V(MA) \tag{14}
$$

This is because, in the news-reports, we can frequently find sentences that contain a sequence of obvious anchors such as numerical data. Each clause in both languages would contain at least one of this sequence of anchors, and intuitively these clauses can be divided into very specific groups. Therefore we use a “Split” factor to decide whether it is appropriate to split a group of clauses into finer alignments. An action of "splitting" should take place only when it is clear enough that the first part and the second part of clauses can be split.

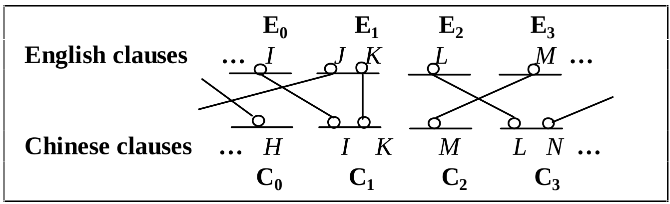

Fig. 1 A group of clauses

Fig. 1 depicts an example of a group of English clauses and Chinese clauses where matched anchors are quite dense. Each horizontal line segment represents an English or Chinese clause($\mathbf{E_0}$~$\mathbf{E_3}$, $\mathbf{C_0}$~$\mathbf{C_3}$). And each circle represents a matched anchor(*I*, *J*, *K*,…), e.g., anchor *I* in English clause $\mathbf{E_0}$ matches with anchor *I* in Chinese clause $\mathbf{C_1}$. In this figure, if we draw a line between $\mathbf{E_1 : C_1}$ and $\mathbf{E_2 : C_2}$, no anchors will match across this line. In other words, all English anchors in or before $\mathbf{E_1}$ only match with Chinese anchors in or before $\mathbf{C_1}$, and vice versa. Therefore splitting these clauses at this line will produce a relatively more specific alignment without losing accuracy, as depicted in Fig. 2. We call this line "Splitting Line". We also need to prevent the program from splitting a case as depicted in Fig. 3, where $\mathbf{C_2}$ has no matched anchor, as a result if we want to draw a Splitting Line, there is no direct way to determine whether to draw it before or after $\mathbf{C_2}$. In this case we have to give up the "splitting" at this place.

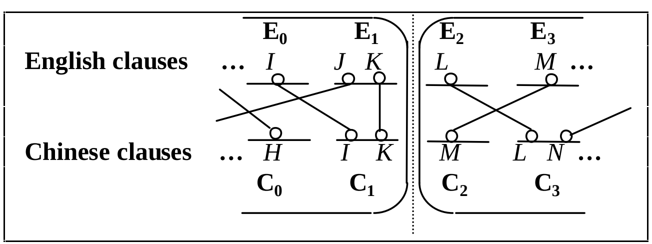

Fig. 2 Splitting clauses

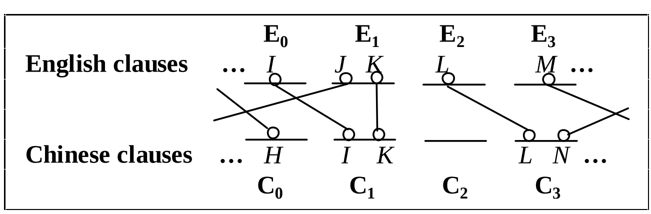

Fig. 3 Sequence of Clauses that should not be split

Therefore a Splitting Line should satisfy two prerequisites:

1) The pair of English and Chinese clauses immediately before and the pair after this line should each have at least one matched anchor.
2) No anchors in the parallel texts will match across this line.

So the splitting function $S(u,i,v,j)$ is determined by looking for the splitting line inside the given group of clauses (*u*,*i*,*v*,*j*):

$$S(u,i,v,j) = \begin{cases} S_f & \text{if found a Splitting Line inside} \\ 1 & \text{Splitting Line not found} \end{cases} \tag{15}$$

Here we give a penalty to the match of the whole group of clauses(*u*,*i*,*v*,*j*) if a Splitting Line exists inside this group, so that matching the two sub-groups separated by the Splitting Line respectively would get an advantage over matching them all together. The above is the simplest example of the use of $S(u,i,v,j)$. In more complicated cases, function $S(u,i,v,j)$ also serves as a measure of the scattering of the matched anchors.

Figs. 4-6 show three typical kinds of CNS news texts aligned by our program. In Fig. 4, there are no obvious anchors. Thus we can only use the length of clauses as a clue to align them. As we can see, a period in English is used to break a complete Chinese sentence into two parts. In Fig. 5, in the first pair of sentences, there are some obvious anchors. But these anchors are cross-matching, and so we match the complete sentence as a whole without any clause level matching. In Fig. 6, there are quite a lot of obvious anchors, and all of these anchors scatter evenly in the same order in both texts. So we split the sentences into clause-to-clause match.

The vice president noted that judicial review corrected and contained the improper application of administrative power while at the same time eliminating or reducing negative impacts on the state administration.	这位副院长称，通过行政诉讼案件的审判，既依法纠正和制止了行政权力的不当行使，消除或减少了因违法行政给国家行政管理带来的负面影响。
It also protects and supports the intra vires acts of the administration.	也依法维护和支持了行政机关依法行使职权。

Fig. 4 Text alignment involving no obvious anchors

Mr. Luo said that 141,949 cases were handled by the Administrative Appeals Tribunals, between October 1990 and June 1995.	罗豪才说，从一九九0年十月至今年六月，全国各级人民法院共受理各类一审行政案¼乘耐蛞磺ㄅ虐偎氖尘偶　£

Decision appealed came from 40 administrative areas including land, public security, urban construction, industry and commerce, environmental protection, prices, finance, customs, forestry, mining, taxation and technological supervision.	讣 嘈蜕婕巴恋∅ 、公安、城建、工商、环保、物价、金融、海关、林业、矿产、税务、技术监督等四十多个行政管理领域。

Fig.5 Text alignment involving cross-matching anchors

In the first nine months of this year,	今年前三个季度，
China's foreign trade reached US$ 197.66 billion,	中国外贸进出口总值已达一千九百七十六亿六 千万美元，
or 25.5 per cent up when compared with the same period last year.	比上年同期增长百分之二十五点五£ ¬
Exports were valued at US$ 107.04 billion,	其中出口一千零七十亿四千万美元，
an increase of 34.8 per cent,	增长百分之三十四点八；
and imports at US$ 90.62 billion,	进口九百零六亿二千万美元，
an increase of 16.2 per cent,	增长百分之十六点二，
The trade surplus stood at US$ 16.42 billion.	外贸顺差一百六十四亿二千万美元。

Fig. 6 Text alignment involving evenly spaced out anchors appearing in the same order in both texts

4. Matching Files

On the CNS web page, the news files in the two languages are small. These bilingual files are not placed in the same order. Therefore it is very tedious to visually match the files according to the titles (and sometimes even content) on their web page and download them one by one manually. To solve this problem, we have also developed a program to first automatically download these English and Chinese files, and then use an algorithm that is quite different from what is described above to match these English and Chinese files. Of course this matching task is simpler because we only need to deal with $1 \Leftrightarrow 1$ match. Thus we do not have to go down into the recursions of Dynamic Programming. But there are also several notable difficulties which we will discuss in detail below.

Our program automatically downloads the news on a daily basis. For each day the program creates a new directory to store the news files for that day. Now the task is to match the English files and Chinese files in a directory.

First of all, we will need to measure the likelihood of matching between an English file and a Chinese file. The measure is determined only by anchors. We do not consider the length factor at the file level matching because of one feature of the CNS bilingual news mentioned in Section 2: Often, the translator will only translate some paragraphs in a Chinese news file into English, while leaving the other Chinese paragraphs in the file not translated at all. As a result there is no relatively fixed correspondence between the lengths of the English and Chinese news files.

Let $M(E, C)$ represent the measure of match between English file E and Chinese file C. $M(E, C)$ consists of specific anchor values:

$$M(E, C) = \sum V(A_E) \tag{16}$$

where $A_E \in \{All\ anchors\ in\ file\ E\}$ and $V(A_E)$ is the value of each A_E. Here we consider every anchor in the English file regardless of whether it matches with a Chinese anchor or not. Because theoretically every English anchor comes from a Chinese anchor, when an English anchor cannot find a Chinese match, it decreases the measure of likelihood that these two files are a matched pair.

Before computing $V(A_E)$, we divide a piece of news into three parts:

1) Headline, the title of the news;
2) Byline, the date of report and place of report in respect of this news;
3) Newsbody, the content of the news.

The anchors in these three parts should be treated separately. Let us look at the Byline first. As the dates of report of all files in the same directory are just the same, the only useful anchor in the Byline is the place of report (i.e. the place where the report was reported from). Interestingly, this place of report is in some sense the strongest anchor in the whole news. For example, if we find that the place of report of a Chinese file is "北京" (Beijing), while the place of report of an English file is "Tokyo", then we can immediately say that these two news files are not a good match. From our observation, only in rare cases (less than 1%) will news labeled with different places match with each other. But on the other hand, if the places of report of two news do match precisely, we cannot say for sure that these two news files also match exactly. There might be several (or even tens of) news reported from the same place (especially from Beijing) on the same day. In this case whether the files do match or not has to be left to the anchors in the other two parts, namely, Headline and Newsbody. So the program gives a sufficiently large penalty (a negative value) to the value of this pair of English and Chinese files if it finds that the two news files are not reported from the same place; but if the program finds that the places of report are the same, it does nothing.

In Byline:

$$V(A_E) = \begin{cases} 0 & \text{when } A_E \text{ has a Chinese match} \\ -P_B & \text{otherwise} \end{cases} \tag{17}$$

Where P_B is a sufficiently large positive number.

Then for each anchor in the Headline and Newsbody parts of an English file, the program looks for a corresponding anchor in the counterparts of a Chinese file. We describe the Headline part and Newsbody part separately.

In the Newsbody of a news, if the program does find a corresponding anchor, it gives the match a positive value as bonus. But if the program cannot find a corresponding anchor, it gives the match a negative value as penalty.

In Newsbody:

$$V(A_E) = \begin{cases} W(A_E)F_R(A_E)F_A(A_E) & \text{when } A_E \text{ has a Chinese match} \\ -W(A_E)F_N & \text{otherwise} \end{cases} \tag{18}$$

where $W(A_E)$, $F_R(A_E)$ and $F_A(A_E)$ have similar definition as $W(MA)$, $F_R(MA)$ and $F_A(MA)$ in Eq.(4). And F_N is a factor for non-matching that is a positive constant.

The case is quite different in Headline. A Headline is usually a short phrase, and the translator often chooses an English expression different from the Chinese expression when translating the title. Therefore sometimes some anchors that appear in the Chinese Headline might be missed in the English Headline, while sometimes some key anchors that appear in the Chinese Newsbody but not in the Chinese Headline will appear in the English Headline. So we would not like to give it any penalty if there are any mismatches of the Headline anchors. Instead the program does nothing in this case. Only when an anchor in the English Headline finds a corresponding anchor in the Chinese Headline will the program give it a positive bonus value.

In Headline:

$$V(A_E) = \begin{cases} W(A_E)F_R(A_E)F_A(A_E) & \text{when } A_E \text{ has a Chinese match} \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

In the aligning method described in the preceding section, we only give a bonus to matched anchors, but never give a penalty to unmatched anchors. Here we introduce the penalty to rule out the possibility of improper matching. The English file usually has a number of anchors, some of which match with anchors in the Chinese file, while others do not match. We add all these anchor values (positive, negative or zero) together, to compute *M(E,C)*, i.e. the final value of this file pair. If *M(E,C)* is negative, the unmatched anchors are stronger than matched anchors, we will discard this file pair at once. But if the result is positive, does it necessarily mean that this file pair is a good match?

From our observation, most file pairs with a small positive value are not good matches either. Therefore we use a positive threshold to filter out such file pairs. Note that the goal of setting up this threshold is two-fold. The first is of course to improve the accuracy of matching files, and the second is to improve the accuracy of the ensuing aligning process. The second aspect is based on a simple observation that the stronger the anchors that exist in the texts, the more precise the alignment can be achieved. In other words, the threshold helps to create good translation examples by selecting good bilingual texts to do alignment.

As we can see, we allow an English file to match with nothing, that is, no Chinese file can match with it with a value larger than the threshold. This phenomenon has two cases:

1) What CNS put on their web page are only some (and not all) of the English and Chinese news that they publish formally. They choose English news and Chinese news independently and somewhat randomly. Therefore some English files indeed have no Chinese matches. This kind of English files might accidentally match some Chinese files with a positive value, therefore the threshold should be set high enough to discard this kind of English files.
2) It is also likely that an English file and a Chinese file are real match, but the value of this pair is below the threshold. We simply discard this file pair because we believe that this kind of file pair is not usable for the ensuing alignment process. Of course, we could lower the threshold to get more file pairs for alignment but at the risk of picking up wrong matches.

On the other hand, there are also cases where an English file matches with more than one Chinese files with a value larger than the threshold. For such cases, we establish a candidate list to record these Chinese files in the descendent order of the value. As depicted below:

$$L_X \quad : \; E_X \rightarrow C_{X1}, C_{X2}, C_{X3} \ldots$$

where E_X is an English file, and C_X's are candidate Chinese files for E_X ; and the order of placing C_X's satisfies:

$$M(E_X, C_{X1}) > M(E_X, C_{X2}) > M(E_X, C_{X3}) > \ldots$$

Naturally, the first Chinese file C_{X1} in the candidate list is the most likely matching candidate for E_X. But there are times when two English files have the same most-likely matching candidate Chinese file. E.g., as depicted below where the heads of the two candidate lists are both C_{X1}.

$$L_X \quad : \; E_X \rightarrow C_{X1}, C_{X2}, C_{X3} \ldots$$
$$L_Y \quad : \; E_Y \rightarrow C_{X1}, C_{Y2}, C_{Y3} \ldots$$

We call this a "collision", and say that L_X *and* L_Y *collide over* C_{X1}. Collision is reasonable since the news reports by nature tend to crowd around some subjects on certain days. A couple of news concerning the same subject may contain similar sets of anchors. As a result the two English files on the same subject are competing for the same Chinese file as their most likely candidate.

When a collision occurs, the simplest approach is to check which value is larger. If $M(E_X, C_{X1}) > M(E_Y, C_{X1})$, we can determine that C_{X1} is the most likely match for E_X, and not E_Y. In this case E_Y's match has to rely on its next candidate C_{Y2}. If there is no C_{Y2}, i.e., only one Chinese file (C_{X1}) can match E_Y with a value above the threshold, then E_Y matches with nothing. If there is a C_{Y2}, we have a new list:

$$L_Y' \quad : E_Y \Rightarrow C_{Y2}, C_{Y3} \ldots$$

We have to check around to see whether L_Y' collides with other lists. If no collision occurs, we can determine that $E_Y \Leftrightarrow C_{Y2}$. Otherwise, we will find a list L_Z that has the same leading candidate C_{Y2} :

$$L_Z \quad : E_Z \Rightarrow C_{Y2}, C_{Z2}, C_{Z3} \ldots$$

The collision of L_Z and L_Y' can be resolved by a similar method. Of course the similar process might produce a new list that collides with a fourth list L_W, and so on. This is called a "collision chain". This process continues until no more collision occurs.

This strategy is straightforward, but it does not always work. First, the anchor values of the contending texts may be close to each other such that a slightly larger value does not necessarily indicate a closer match. Secondly, there may be groups of English files with each group competing for a particular Chinese file as the common head of their candidate lists. A Chinese file, that appears as the head of a group of candidate lists, may also be an intermediate member in other groups of candidate lists. Through a process of elimination similar to that mentioned above, an intermediate member could surface as a header thus competing with an existing group that has the same header. In other words, the resolution process could result in a cyclic collision chain. To overcome the above problem, we propose a human assisted resolution strategy.

To facilitate this strategy, we first employ a greedy algorithm to repetitively prune the seemingly "weak" candidate pairs as discussed above, until finally only the strongest ones are left. The human assisted resolution strategy is then applied on the remaining collision groups. The following pseudo code gives an overview of the greedy algorithm. Here, each collision group contains lists of contending files having the same header file. Each collision group, where possible, is split into two subgroups. The weaker subgroup will have the common header file pruned, such that member lists in the weaker subgroup will now join other collision groups depending their new header files. Thus when a collision group is split, the entire space of collision groups is updated because of new members joining other collision groups, and the whole process of searching a collision group to split starts afresh. When a collision group cannot be split, the process looks for the next collision group to split. This double loop process is iterated until none of the collision groups can be further split.

```
LOOP1:
        Search for collision groups from the whole space
```

```
        LOOP2:
        FOREACH (collision groups)
                IF (this collision group is splittable)
                    split this collision group;
                    GOTO LOOP1;
                ELSE
                    get next collision group;
        ENDLOOP2
ENDLOOP1
```

We will now elaborate further on the splitting process and to show that the double loop iteration in the above algorithm will eventually terminate. We will now use a simplified denotation $V(L_X)$ to represent $M(E_X\ ,\ C_{X1})$ hereafter. We first group candidate lists into groups CG_A, CG_B, …. with the groups' candidate lists colliding over Chinese files, C_A, C_B, …., respectively. Note that some of these groups may have only one candidate list, in which case, there is actually no collision. However, these group will evolve as discussed below. Some of the single-list groups may have other lists joining them later, while multiple-list groups may end up with only one list in the end. Thus, we will call them collision groups regardless of whether they have only one member or more. However, in each round of searching, we will select a collision group which has multiple lists as the purpose is to resolve collision through a process of splitting to be described below.

In the first round of searching, we find a collision group, say, CG_A that contains n candidate lists (L_1 , L_2 , …, L_n) colliding over Chinese file C_A . Here the colliding lists (L_1 , L_2 , …, L_n) are placed in the descending order of $V(L_i)$, where $1 \leq i \leq n$.

L_1 : $E_1 \rightarrow C_A , C_{12} , C_{13} \ldots$
L_2 : $E_2 \rightarrow C_A , C_{22} , C_{33} \ldots$
……
L_n : $E_n \rightarrow C_A , C_{n2} , C_{n3} \ldots$

Since $V(L_1)$ is the largest, we regard L_1 as a base, and measure the distance between the value of each subsequent list and L_1 to decide whether these lists can remain in the collision group. In practice, we use a threshold $T_A=V(L_1)*S$, where S is a constant number between 0 and 1 and close to 1. To decide whether L_i has the right to stay in CG_A , we compare $V(L_i)$ and T_A. We now scan the group from L_2 on until we come to the first L_P that satisfies $V(L_P) < T_A$. If such a L_P can be found, then P is called the split position, and such a collision group is called a "*splittable*" group. The collision group is then split into two subgroups at this position.

The first subgroup is (L_1 , L_2 , …, L_{P-1}). All the lists from L_2 to L_{P-1} are strong enough to compete with L_1, and therefore, in the meantime, we keep this subgroup of candidate lists in CG_A. This subgroup is itself an "*unsplittable*" collision group.

The second subgroup is (L_P , L_{P+1} , …, L_n):

L_P : $E_P \rightarrow C_A, C_{P2}, C_{P3} \ldots$
L_{P+1} : $E_{(P+1)} \rightarrow C_A, C_{(P+1)2}, C_{(P+1)3} \ldots$
……
L_n : $E_n \rightarrow C_A, C_{n2}, C_{n3} \ldots$

These lists are too weak and therefore are kicked out of CG_A. Since the English files of these lists have been deprived of the right to match with C_A , we should take out C_A from these lists to form new candidate lists:

L_P' : $E_P \rightarrow C_{P2}, C_{P3} \ldots$
L_{P+1}' : $E_{(P+1)} \rightarrow C_{(P+1)2}, C_{(P+1)3} \ldots$
……
L_n' : $E_n \rightarrow C_{n2}, C_{n3} \ldots$

If a L_Q $(P \le Q \le n)$ contains only one Chinese file C_A , then L_Q' is an empty list and is eliminated from further consideration as it is deemed to have no matching Chinese file.

On the other hand, if L_Q' is not empty, we replace the original list L_Q with the new list L_Q' and we say that L_Q' is **derived** from L_Q. We also say that L_Q' is **derived** from CG_A. Note that some of these new lists may collide with other collision groups discovered later. Also note that $V(L_Q') \le V(L_Q)$. This fact will be very important in the later rounds.

In the second round, we pick another collision group. We may come across CG_A again, but it is already *unsplittable* and so we keep on searching and find another collision group CG_B , which might include some lists derived from CG_A. We repeat the splitting process, and obtain an *unsplittable* CG_B and some other new lists derived from CG_B. One important point to note is that some of these new lists derived from CG_B may have C_A has their leading candidates. Thus they will collide back with collision group CG_A , and make it *splittable* again.

In the third round of search, we come across CG_A again, and this time we might also find some lists derived from CG_B colliding with CG_A . These derived lists and CG_A together form a new collision group. We have to split this new collision group again. If, for example, a L_K' derived from CG_B is now colliding with CG_A , there are three cases in the splitting process as discussed below.

1) $V(L_K') < T_A$,
   Then L_K' should be kicked out by CG_A. Therefore we need to generate a further derived list L_K'' which may join another collision group or form a new collision group with itself as the only list in the group.

2) $V(L_1) > V(L_K') > T_A$,
   Then L_K' becomes a new member of the *unsplittable* collision group CG_A .

3) $V(L_K') > V(L_1)$,

Then we have to use L_K' as a base to re-compute T_A. With the new T_A, we split the group again. This splitting process now may generate new derived lists, some of which may again collide with CG_B.

In the fourth round of searching, we may find some lists derived from the third round, say, L_J' , colliding with CG_B. But unlike that in the third round, this collision in the fourth round has no chance to make any change to CG_B . The reasons are stated below.

Since L_J was kicked out from CG_A in the third round by case (3) above, we have:

$$V(L_J') < V(L_J) < V(L_K'),$$

and since L_K was kicked out from CG_B in the second round, we have:

$$V(L_K') < V(L_K) < T_B ,$$

Therefore:

$$V(L_J') < T_B .$$

This means L_J' is destined to be kicked out by CG_B (similar to case 1 described in the third round). Therefore CG_B will not be affected in this round. Of course we need to further derive L_J'' from L_J' and replace L_J' with L_J''. But, since L_J'' was derived from both CG_B and CG_A , it has lost its previous heading candidates C_B and C_A. This rules out the possibility of endless looping of the collision chain.

We repeat this process until finally there are no more *splittable* collision groups. Now any collision group that has only a single candidate list decides that its English file and the leading Chinese candidate are a matched pair. For the collision groups that contain more than one candidate list, the program then must prompt the user to choose a correct matching combination. Assuming a scenario involving a collision group as follows:

L_1 : $E_1 \rightarrow C_A , C_{11} , C_{12}$
L_2 : $E_2 \rightarrow C_A , C_{21} , C_{32}$
L_3 : $E_3 \rightarrow C_A , C_{31} , C_{32}$

The system will display the first few lines of every file involved in the above group, and prompts all the possible matching combinations for the user to choose. For example, the user can choose to match E_1 with C_{11}, E_2 with C_A and E_3 with C_{31}. If among all the displayed Chinese files, the user cannot find a match for an English file, the user can just discard this English file. The assumption is that after the collision resolution process, the remaining collision groups would be small and simple to resolve, such that the user can choose a matching combination easily. In practice, most final collision groups contain only two lists (no collision group would contain more than three lists) even if we lower the threshold. Also, no final list contains more than three candidate Chinese files. In any case, to match among less than 10 (usually 4 or 5) files a time is still much easier than to match among more than 100 files manually.

5. Discussion

A major issue in text alignment is the measure of correspondence between parallel corpora in two different languages. Matching criteria may be divided into two categories: internal correspondence and *a priori* information. Internal correspondences are those matching evidences in parallel texts that can be found out by the program independently without consulting extra resources. *A priori* information refers to existing extra matching correspondences that can be consulted by the program during the alignment process.

5.1 Text Length

Text length can be regarded as an internal evidence by and large, although measuring text segment's length depends on the determination of segment boundaries which may involve extra heuristics in centain cases. The observation that "longer sentence in one language tends to be translated into longer sentence in another language" is very simple and straightforward, yet it was a surprise that such a simple correspondence would work so well in the early alignment work (Brown et al. 1990, Gale and Church, 1991, 1993).

However, pure length-based alignment method has been criticized ever since it appeared for its drawbacks: (a) It is not robust. It relies heavily on a clearn corpus of almost literal translation; (b) It lacks the ability to automatically recover from alignment error: a deletion of a sentence in a series of sentences with a similar length would cause the alignment to err as far as it can; and (c) Although the correlation of text length seems to exist betwen any language pairs, the extent on this correlation may not be so reliable in other language pairs as in between similar languages which share cognates or grammatical similarities. Therefore pure length-based alignment did not appear any more after these two pioneering papers.

5.2 Lexical Correlation

Lexical information can be either internal or *a priori*. Lexical correlation may be at the level of verbal, syntactic or semantic. Syntactic and semantic correlation, as exemplified by the work of Chen & Chen (1995) using part of speech tagging and Matsumoto et al (1993) using dependency structure graphs, respectively, require much deeper text analyses than verbal correlation. Verbal correlation is thus more often used for its simplicity.

One useful verbal correlation is found in language pairs that have historical relation basing on the similarity of appearance in the two languages. A good example is *cognate* correlation between English and French proposed by Simard et al. (1992). Another example is the concurrence of *Kanji* between Chinese and Japanese introduced by Tan and Nagao (1994). This kind of appearance correlation can be directly located by a pattern matching scheme without consulting an external lexicon, so it could be considered as an internal evidence.

A related verbal correlation is also concerning the same appearance of some tokens in the two texts, not because of any historical relation between the languages, but because of translation of certain special tokens, such as expressions of numbers, dates, or technical terms. For example, financial articles may contain a lot of numbers and technical manuals may contain a lot of jargons that were copied into the target texts in the translation process. So it is not unimaginable that in some cases just a simple matching of these tokens would help alignment greatly.

A third possible type of verbal correlation is the correlation of pronunciation as suggested by Melamed (1996b). When translating certain terms such as name of a person from one language into a different language, the translator usually chooses the character sequences in the target language that have the pronunciation closest to the pronunciation of the name in the source language. For similar language pairs, this correlation could appear as cognates, but for much different pairs, this correlation could be hidden under different written character systems.

Besides direct matching, verbal correlation can also be determined by looking up a bilingual lexicon. Since this lexicon already exists before the alignment, we regard it as *a priori*. Bilingual lexicons usually contain entries of word translation. For Indo-European languages, the words in the text are already separated by white spaces as delimiters. But for languages such as Chinese and Japanese, there is no obvious word delimiters. A text segmentation process is required to separate the words in the first place. Especially, when a large Machine Readable Bilingual Dictionary (MRBD) is available and a great proportion of words in the text can find entries in the MRBD, a thorough word segmentation is desirable. If no large-scale MRBD is available, a small hand-crafted lexicon can be also helpful, given that the entry words are very reliable and unambiguous (Wu 1994, Xu & Tan 1996). Such a small lexicon usually does not require a thorough segmentation of the texts: a program can just scan every character in the texts in sequence and test if a string starting with this character is an entry of the lexicon.

Without consulting an existing lexicon, verbal correlation can also be discovered by comparing distributions of every pair of words in the parallel texts, like the works in (Kay & Roscheisen 1993) and (Fung & Church 1994). The assumption that "words that are mutual translation in two languages should have similar distribution in the parallel texts" is as straightforward as the length-correlation observation.

5.3 Our Method

Our method deals with bilingual news articles which are a kind of free translation and in some cases involve translation omissions. Thus length-based criterion alone is not enough for the alignment task. However we do not rely on any in-depth text analysis such as syntactic or semantic measures to deal with the free translation problems. Instead, only a small MRBD is used to provide keywords as anchors in the form of *a priori* information. This has dispensed with the need for the Chinese word segmentation in order to consult the lexicon. While the reliance on anchors resembles the work of Wu (1994), our method has to deal with more complicated matching than the predominantly literal translation in the Hansard record reported in (Wu 1994).

The minimal use of verbal correlation through the anchor matching also allows us to work on alignment at the clause level. Our clausal alignment approach may be considered a special case of the phrasal alignment proposed by Wu (1997). However, the Inversion Transduction Grammar proposed in (Wu 1997) to achieve phrasal alignment requires a lot of *a priori* knowledge including grammar rules and their associated probabilities as opposed to the simple anchor matching in our work to realize clausal alignment.

The file matching in our work is unique as it involves matching of bilingual text files that are in different sequential orders. Translation omission further complicates the matter such that the text length cannot be used as a correspondence measure. Anchor matching is used as the only measure for matching and the resolving of "matching collision". It is interesting to note that our "matching collision" problem in file matching is comparable to the problem of "indirect association" in the study of word concurrence reported in Melamed (1997). His solution to "indirect association", i.e. the "competitive linking algorithm", is also similar to our solution, but looks much simpler than ours. As he pointed out: "The simplicity of the competitive linking algorithm depends on the one-to-one assumption: each word translates to at most one other word". On the other hand, our collision resolution algorithm looks so complex because we are trying to keep a *set* of strongest candidates, i.e., we try to achieve many-to-one matching in our method.

6. Determination of Parameters

We used a Genetic Algorithm software package called "GAucsd"(Schraudolph 1992) to determine the appropriate values of all the parameters discussed in the earlier sections. These parameters are the weighting factors for the various components in the cost function. GAucsd implements the complex procedure of Genetic Algorithm for general function-optimization purpose. We modified our alignment program to take as input an array of floating numbers containing the tentative values of these parameters, and return as output the sum of precision and recall of the alignment based on these parameter values. We allow the GAucsd package to supply different sets of the parameter values into the alignment program to find out the set of parameter values that can lead to the largest possible value of the sum of alignment precision and recall. From the viewpoint of GAucsd, the alignment program is only a fitness evaluation function, an array of parameters is a genotype, and the sum of precision and recall returned by the alignment program is the fitness measure of this genotype.

6.1 The Role of Gaucsd

The role of GAucsd package is to keep feeding different sets of parameter values into the alignment program until a satisfactory set of parameters is reached. Genetic Algorithm is used in GAuscd to reproduce new sets of parameters from a collection of the fittest sets of parameters obtained in the previous generation. GAucsd then concatenates the binary representation[2] of the whole set of parameter values (all floating numbers) to form a binary string which serves as a genotype for the later genetic operations. The initial population contains 80 randomly generated genotypes.

After initialization, GAucsd iteratively goes through the following three steps:

1) Fitness Evaluation: GAucsd decodes every genotype to obtain a set of floating numbers and feed them to the alignment program which returns the fitness value of the genotype.

2) Selection: GAucsd uses a “spinning wheel” algorithm to pick out a number of the fittest genotypes according to their fitness values, and remove all the other genotypes.[3]

3) Genetic Operation: GAucsd performs two genetic operations on the genotypes selected in step 2) above to create a new generation:

- Mutation. Each bit of every genotype mutates(turning from 1 to 0 or vice versa) at a certain probability called Mutation_Rate. In our experiment, we set Mutation_Rate at 1.3%.
- Crossover. A pair of genotypes exchange a segment of their bit strings to generate two new genotypes. Crossover also occurs at a probability called Crossover_Rate. In our experiment, the Crossover_Rate is set at 60%.

Then go back to 1).

We specify the convergence threshold = .99, which means when 99% of the genotypes have the same fitness value, the task is completed.

6.2 The Role of the Alignment Program

For a pair of English file and Chinese file, a manual alignment is done to serve as a yardstick for computing precision and recall. Here the definitions of precision and recall are similar to that in (Chen & Chen 1995). With each set of parameters supplied by

[2] In practice, each parameter is represented in 10 bits.
[3] This is only a rough description. See Schraudolph(1992) for more details.

GAucsd, the alignment program will then run the dynamic programming algorithm to align this pair of English file and Chinese file. It then compares the trial alignment result with the manual alignment result to obtain the precision and recall. The sum of precision and recall indicates the degree of similarity between this trial alignment result and the earlier manual alignment result. The goal is thus to find a combination of parameter values that achieve the greatest similarity between the automatic alignment and the manual alignment.

6.3 Results

Note that if the sample set of English and Chinese texts is too small, there may be many very different combinations of parameter values that can achieve the value 2 in the sum of precision and recall. We must therefore have a large enough sample set to make the final parameters reliable. We tested this method on around 20K of English texts and 13K of Chinese texts. With the sum of precision and recall achieving 1.97, we obtained the best combination of parameter values as follows:

f_R = .12 (factor of repeated anchor)
f_A = .33 (factor of approximate match of anchor)

f_a = 1 (factor for anchor, set to 1 to obtain the values of the other factors below relative to f_a)
f_l = .55 (factor for length)
f_{mp} = .076 (penalty factor for multiple-paragraph matching)
f_{ms} = .74 (penalty factor for multiple-sentence matching)
f_{ws} = .82 (bonus factor for whole sentence matching)
S_f = .39 (splitting factor when a splitting line exists)
W_P = 23 (weight for a pair of matched anchors if it is a place name)
W_D = 37 (weight for a pair of matched anchors if it is a date)

7. Experimental Results

7.1 File Matching

To date we have downloaded 17,852 Chinese files and 3,869 English files from CNS website[4]. The process of automatic file-matching produced 1,642 confident pairs of English and Chinese files. During the file-matching process, the program prompted 14 times altogether for the user to choose a correct matching combination among two to four possible matching combinations. Among all the 14 times, for 6 times the user only needs to press an "Enter", i.e., to accept the default choice produced by the program according to the match value. That is to say, when the difference between matching values is small enough, it is very likely (8/14 as we tested) that the match with the

[4] The numbers of Chinese and English files differ so greatly because the translators at CNS do not translate all the Chinese news into English. Only about 20% of the Chinese news files are translated.

larger value is not the correct match. This proves the necessity of introducing human intervention.

We have checked 300 randomly chosen pairs and found all of them are correct matches, i.e., the precision is 100%. Without loss of generality, we can assume that the matching precision as a whole is 100%.

To check whether a pair is correct, we only need to take a glimpse at the title or at most the first few lines of this pair of English and Chinese files. But in contrast, recall is not so easy to compute because checking how many file pairs indeed exist involves manual matching, and as we explained before, manually matching files among more than a hundred files is really a laborious task. Nevertheless, we also took time to examine 8 directories which contain 526 Chinese files and 107 English files altogether, and as a result we found 83 pairs of real matches, among which 47 were paired up by the matching program. Assuming these 8 directories represent all the 266 directories of news files we have downloaded, the recall of matching process is $47/83 \approx 57\%$.

The recall seems fairly low. But as we discussed in the last section, there is a trade-off between recall and precision. Raising the threshold of matching value results in an improvement in precision but a drop in recall. On the other hand, relaxing the threshold will have a contrary result. The reasons that we set the threshold so high to give an advantage to precision are:

1) We intend to make sure that these selected file pairs will have a high accuracy in the sentence-clause level aligning. In general, as long as the corpora are taken from real-life texts without specialists' digesting or editing, there will always be lots of texts in the corpora that are not very useful to linguistics researchers. The not-so-useful texts can be classified into two categories: **a)**large amount of repetition of the same or similar simple strings, such as vote records in the Hansards; **b)**texts that researchers still cannot make good use of in the present research[5]. Picking out and discarding such texts should not be considered a waste of resources, but rather, this action eliminates these not-so-useful texts and will only make the process of other useful texts more efficient.

2) Our initial objective is to apply our method to large-scale corpora (though the scale of our bilingual corpora at hand still cannot be compared with those real large ones such as Canada Hansards). Given that the size of the corpora is large enough, part of the corpora resulted from the file matching process will still be large enough. Thus even with a lower recall, we will still be able to obtain a sufficiently large number of high precision examples.

7.2 Aligning

[5] In the case of file matching, the news files that could not be aligned with high accuracy in the later aligning process are not so useful.

We applied the aligning process to all the pairs of English and Chinese files obtained from the preceding file matching process. To analyze the result, we checked 40 pairs of randomly selected news files. The basic data of these files are shown in **Table 1.**

	English	Chinese
Files	40	40
Paragraphs	257	304
Sentences	601	632
Clauses	1,570	1,697
Characters	71,384	33,276

Table 1 Data of news files under testing

The precision of the algorithm is measured by dividing the number of correct pairs by the total pairs obtained from the algorithm . We have subjected these files to three different testing methods. The first was done by only allowing the length measure to govern the alignment process. The second was done by only allowing anchors, and in the last method the whole function was used, namely, both length and anchors. The results are tabulated in **Table 2**.

		Using Length Measure	Using Anchors	Using both Length and Anchor
1	No. of correct paragraph pairs found by the program	155	149	210
2	No. of total paragraph pairs	213	219	217
3	Precision of paragraph matching	155/213=72.7%	149/219=68.0%	210/217=96.8%
4	No. of actually correct paragraph pairs	214	214	214
5	Recall of paragraph matching	155/214=72.4%	149/214=69.6%	210/214=98.1%
6	No. of correct clause pairs	249	386	472
7	No. of total clause pairs	421	553	526
8	Precision of clause matching	59.1%	69.8%	89.7%
9	No. of correct clause pairs in correct paragraph pairs	249	386	472
10	No. of total clause pairs in correct paragraph pairs	302	474	507
11	Precision of clause matching in correct paragraph pairs	82.4%	81.4%	93.1%

Table 2 Alignment results obtained from 3 different testing methods.

Of the 40 news files, the number of paragraphs aligned correctly by each of the three methods were 155, 149, 210, respectively, as shown in Row(1) of **Table 2**. While the total number of paragraph pairs obtained by the three methods were similar to each other as shown in Row(2), it does not necessarily mean the matches were similar. Actually the first two methods made most of the mistakes at different places. Since we allowed matching of multiple paragraphs, it is believed that the similarity of the total number of paragraph pairs is coincidental. This indicates that both the length measure and anchors are needed to produce good alignment results at the level of paragraph matching. Examination of the contents shows that where translation patterns in the bilingual texts are quite regular (such as without skipping Chinese paragraphs in translation), the length measure alone would be sufficient. On the other hand, when there are plenty of anchors, the anchors can become a dominating factor.

In Row(4), 214 is the number of correct paragraph pairs done by hand. Dividing the correct paragraph pairs found by the program by 214, we get the recall of these three methods. As we can see, only the third method can produce a high recall.

Note that we adopt the measure of "recall" only at the level of paragraph matching, and not at the level of clause matching. This is because at the level of paragraph matching, we only allow three kinds of alignment: $0 \Leftrightarrow n$, $1 \Leftrightarrow n$ and $n \Leftrightarrow 1$. Pairing of paragraphs under any of the three condition is fixed and deterministic. On the other hand, at the level of clause matching, we allow $m \Leftrightarrow n$ aligning. Matching between larger numbers of clauses may subsume matching between smaller numbers of clauses. Yet both matching can be considered correct. Thus, there is no unique mapping at the clause level, and therefore it may not be possible to count the total number of correct pairs at the clause level. Hence the recall of clause-aligning is undecidable. That is why we only use precision to measure the alignment result at the clause level.

Next in Row(6)-(8) it is seen that not only the number of correct pairs, but also that of total pairs produced are quite different among the three methods. Using the length measure alone lacks the ability to divide long sentences into short clause-to-clause alignment. So the total pairs produced is much less than the other two methods, as shown in Row(7). But when we say a pair is correct, we mean that the contents of this pair in both languages are best matched, without finer alignment between sub-components of the pair. So we might call a pair of sentences a correct pair even if it can be divided into many clause-to-clause pairs. As we can see in Row(8), the only acceptable precision is by using both length and anchor measures.

Noting that quite a number of wrong clause pairs exist in those wrong paragraph pairs (actually all the clause pairs produced from the wrong paragraph pairs should be considered wrong), we therefore count the clause pairs produced from the correct paragraph pairs only, as listed in Row(9)-(11). It is interesting to see in Row(11) that given the correct alignment of paragraphs, the precision figures of using the length measure alone and using the anchor measure alone have both increased significantly. And the precision of using both measures in this case hits a high of 93.1%. Considering the texts are free translation, this is quite a satisfactory result.

7.3 Analysis of Computational Complexity

We will briefly analyze the complexity of the file matching and text alignment algorithms. For the alignment work, its complexity is similar to existing works that make use of dynamic programming. The upper bound of the computing time is $O(m*n)$ where m and n are the numbers of basic English and Chinese text units, respectively, for alignment. In the case of alignment of paragraphs, the text units are paragraphs while in alignment of sentences and clauses, the text units are clauses.

For the file matching, most of its computational cost is spent on analyzing the English and Chinese files, and on the calculation of the anchor values of every pair of English and Chinese files, and if the file number is large enough, mainly on the latter. So the computational complexity is at least $o(m*n)$, where m and n similarly refer to the numbers of the respective basic text units, but in this case, the basic units are the English and Chinese files downloaded for file matching.

The precise analysis of the computational complexity of the collision resolver is somewhat difficult, but we can still roughly determine its upper bound. The essential task of the resolver is the repetitive splitting of the collision groups, and each time deriving some candidate file pairs. In a hypothetical extreme case, all the $m*n$ file pairs downloaded for file matching are processed and finally driven out by the resolver, and there would be no more than $m*n$ rounds of searching and splitting. Hence the upper bound of the Collision Resolver is $O(m*n)$.

Of course, collisions, in reality, are a lot simpler than the extreme case. As far as what we have seen from our experiment, the collision resolution did not noticeably affect the whole computing time of file-matching. Simply put, the overall computational complexity of the file analysis, anchor value computation and collision resolution is $\theta(m*n)$, i.e., the whole computing time would depend only on $m*n$.

8. Conclusion

8.1 Summary

We have presented an algorithm for automatic alignment of English and Chinese bilingual CNS news and a method to match these bilingual files before aligning. While works on automatic alignment on other text pairs have been reported in the literature, our work here is unique in several respects. Firstly, the method handles two languages that are widely different in their sentential structures. Secondly, because of the sentential structure differences, we have decided not to constrain matching at sentence boundaries as in the earlier works. Thirdly, we avail ourselves of the feature of news reports, namely occurrence of common anchors, and make them a lexical cue to be combined with the length-based statistical approach to improve the alignment. Fourthly, by measuring the scattering of the anchors, we can achieve a clause-to-clause match in some parts of the texts. Finally, we use a genetic algorithm to determine the weighting factors among the various components in the cost function for the dynamic programming.

Because we allow matching at the clause level, matching is more difficult and error-prone. However, the use of both the length measure and anchors has proven to be a remarkable means of guiding the automatic alignment with a precision rate up to more than 90% in corpora of free translation.

As for the problem of file-matching, there is an issue of matching collision. We discuss at length on the various cases of collision and propose a strategy to resolve collisions. This strategy repeatedly keeps searching the whole set of candidate files to split those lists that are in conflict until there is no more list that can be split further. The final *unsplittable* lists which survive through this repetitive splitting process are used to prompt to the user for human selection. Here we introduce human intervention to improve the precision of matching, and as we tested, we achieved a precision of 100% and recall of 57% with a fairly small amount of human efforts in collision resolution.

8.2 Future Work

The project only produces a special-purpose experimental system. We have focused on the news files collected from CNS website, and the system can only work with these news files according to their news format. Besides, the corpora are not large enough. Therefore a likely ensuing work is to collect more parallel news from possibly other resources, and modularize the programs so that they can easily deal with other news formats.

A possible enhancement to this system may be a self-adaptive special-term list. At the moment, the list is only a simple bilingual lexicon, and to enhance it, we could add a weight to each lexicon entry to tell the program how reliable the correspondence between this pair of English and Chinese anchors is. And each time when an alignment finishes, the program could compute and write new weights into this lexicon.

In fact this lexicon is just a miniature Machine Readable Bilingual Dictionary(MRBD). The reasons we use our present hand-crafted lexicon are: 1) the proper nouns are less ambiguous, and 2) we do not have an MRBD at hand. However when one day the need arises, with adequate computing power and an available MRBD, we could build a more powerful system by looking up every word in the corpora from MRBD and use them as criteria to align the corpora. And of course, every entry of the MRBD should have a weight, which could be computed by the previously aligned parallel corpora. The special-terms, of course, should have much higher initial weights, so that their already recognized unambiguity can be reflected. And on a self-adaptive enhancement basis, the weights of entries in the MRBD could evolve through time. When the weight of an entry is raised to a threshold, we could say we have found a new special-term which is deemed reliable for the alignment job.

9. Acknowledgments

Special thanks are due to China News Service for their permission to use their news reports.

References

Brown, Peter F., Lai, Jennifer C. and Mercer, Robert L.: 1991, 'Aligning sentences in parallel corpora', in *Proceedings of 29th Annual Meetings of the ACL*, pp.169-176.

Brown, Peter F.; Cocke, John; Pietra, Stephen A. Della; Pietra, Vincent J. Della; Jelinek, Fredrick; Lafferty, John D.; Mercer, Robert L. and Roossin, Paul S: 1990, 'A Statistical Approach To Machine Translation'. Computational Linguistics, Vol 16, No.2.

Chen, Kuang-Hua and Chen, Hsin-His: 1995, 'Aligning Bilingual corpus: Especially for Language Pairs from Different Families', *Information Sciences-Applications* v04 N2 pp.57-81.

Chen, Stanley F.: 1993, 'Aligning Sentences In Bilingual Corpora Using Lexical Information'. In the *proceedings of 31st Annual Meeting of ACL*, pp.9-16

Fung, Pascale and Church, Kenneth W., 'K-vec: A new approach for aligning parallel texts', In the *Proceedings of 15th International Conference on Computational Linguistics (COLING-94)*, pp.1096-1102, 1994.

Fung, Pascale and Mckeown, Kathleen: 1997, 'A Technical Word And Term Translation Aid', *Machine Translation*, vol. 12, pp.53-87

Gale, William A. and Church, Kenneth.W.: 1991, 'A program for aligning sentences in bilingual corpora' *Proceedings of 29th Annual Meeting of the ACL*, pp.177-184.

Gale, William A. and Church, Kenneth. W.: 1993, 'A program for aligning sentences in bilingual corpora' *Computational Linguistics*, vol.19, no.1, pp.75-102.

Haruno, Masahiko and Yamazaki, Takefumi: 1996, 'High-performance Bilingual Text Alignment Using Statistical And Dictionary Information', In the *proceedings of ACL-96*.

Hwang, Dosam and Nagao, Makoto: 1994, 'Aligning of Japanese and Korean texts by analogy', 情报处理学会研究报告, 94-NL-99, vol.94, no.9, pp.87-94.

Kay, Martin: 1997, 'The Proper Place of Men and Machines in Language Translation', *Machine Translation* vol 12, pp. 3-23

Kay, Martin and Roscheisen, Martin: 1993, 'Text-translation alignment,' *Computational Linguistics*, vol.19, no.1, pp.121-142.

Ker, Sue J. and Chang , Jason S.: 1997, 'A Class-based Approach to Word Alignment', *Computational Linguistics*, vol. 23, no. 2., pp. 313-341

Klavans, Judith L. and Resnik, Philip: 1996, *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, eds.

Langlais, Philippe, 'A System to Align Complex Bilingual Corpora', in TMH-QPSR 4/1997, KTH, Stockholm, Sweden, 1997.

Matsumoto, Yuji; Ishimoto, Hiroyuki and Utsuro, Takehito, 'Structural matching of parallel texts', In the *Proceedings of Conference of Association of Computational Linguistics, ACL-93*, 1993.

Melamed, I. Dan: 1996a, 'A Geometric Approach to Mapping Bitext Correspondence', In the *proceedings of Conference on Empirical Methods in Natural Language Processing 1996*.

Melamed, I. Dan: 1996b, 'Automatic construction of clean broad-coverage translation lexicons", In the *Proceedings of 2nd Conference of the Association for Machine Translation in the America*, 1996.

Melamed, I. Dan: 1997, 'A word-to-Word Model of Translational Equivalence', In the *proceedings of ACL97*.

Nagao, Makoto: 1984, 'A framework of a mechanical translation between Japanese and English by analogy principle', in *Artificial and Human Intelligence*, eds. Elithorn and R. Baner, Elevier Science Publishers, B.V.

Sato, Satoshi: 1991, *Example-Based Machine Translation*, Ph.D. Thesis, Kyoto University.

Schraudolph, Nicol N.: 1992, 'A User's Guide to GAucsd 1.4', *Technical Report* CS92-249, CSE Department, UC San Diego.

Simard, Michel; Foster, George. F. and Isabelle, Pierre: 1992, 'Using Cognates to Align Sentences in Bilingual Corpora', in *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pp.67-81, Montreal, Canada.

Tan, Chew Lim and Nagao, Makoto: 1995, 'Automatic Alignment of Japanese-Chinese Bilingual Texts', *IEICE Transactions On Information and Systems* Vol.E78-D, No.1.

Wu, Dekai: 1994, 'Aligning Parallel English Chinese Text Statistically with Lexical Criteria,' *ACL*-94

Wu, Dekai: 1997, 'Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora', *Computational Linguistics*, Vol. 23, no. 3, pp.377-403

Xu, Donghua and Tan, Chew Lim: 1996, 'Automatic Alignment of English-Chinese Bilingual Texts of CNS News', In *Proceedings of International Conference on Chinese Computing'96*, pp.90-97

Xu, Donghua: Feb 1997, *Matching And Aligning of English and Chinese Bilingual Texts of CNS News*, M.Sc. Thesis, National University of Singapore.