

# Parallel Web Text Mining for Cross-Language IR

**Jiang Chen and Jian-Yun Nie**

Département d'Informatique et Recherche Opérationnelle

Université de Montréal

C.P. 6128, succursale CENTRE-VILLE

Montreal (Quebec), Canada H3C 3J7

{chen, nie}@iro.umontreal.ca

February 24, 2000

## Abstract

One of the approaches to cross-language information retrieval (CLIR) is based on the use of parallel texts. In this paper, we will describe a parallel text mining system called PTMiner (Parallel Text Miner) for the Web environment. We will explain the underlying mining algorithm of this system as well as its implementation using a distributed model and database technology. The resulted corpora are used as the training material for statistical translation models. Preliminary experimental results using the models for CLIR are reported.

## 1 Introduction

Data mining, text mining and other knowledge discovering techniques have become an attractive research area in the past years. The enormous amount of information often offers potential solutions to some problems. This is the case of parallel texts that provide translation examples from a language to another. A pair of parallel texts is two such texts that are translation one for the other.

In our work, the need for parallel corpora is originated from the research of cross-language information retrieval (CLIR) using statistical translation model. There are several approaches to query translation in CLIR (Nie et al., 1999): using a machine translation (MT) system, using a bilingual dictionary or terminology base, and using a statistical translation model. For dictionary-based translation, because one word could have different meanings, it is very difficult to give a correct translation among many choices. MT systems have similar problems. They often select wrong words in their translations, and their selection process, which determines a unique translation word/term for an original word/term, prevents us from expanding original queries by synonyms. In addition, MT systems are not available for many language pairs. The idea of statistical translation model is to learn translation correspondences from a training parallel corpus. Such a translation model is much easier to establish than MT systems, and it could be sensitive to the domain of the training corpus. However, the obstacle to this approach is the poor availability of large parallel corpora for many language pairs. In this paper, we propose an approach that explores automatically the World Wide Web, a giant information resource where a large amount of parallel Web pages exists.

Web text mining extended the functions of traditional web search engines. Web content miners not only do simple text searching but also try to extract implicit information by categorizing, filtering and interpreting Web documents. To achieve these functions, people either develop intelligent web

agents (Brown et al., 1994; Balabanovic, Shoham, and Yun, 1995; Doorenbos, Etzioni, and Weld, 1996; Perkowitz and Etzioni, 1995) for various demands or set up multilevel databases based on the Web information and Web query systems (Konopnicki and Shmueli, 1995; Zaiane and Han, 1998). In our work, an intelligent parallel text miner is developed which, in addition, determines if two texts are parallel. Using this system, we actually collected two large-size parallel corpora for English-French and English-Chinese.

In the rest of the paper, we will explain the mining algorithm we adopted and its implementation using a distributed model. The translation models trained by the generated parallel corpora and their performance in CLIR will also be addressed.

## 2 Parallel Text Mining Algorithm

There are various ways to search for parallel texts from the Web, depending on the amount of parallel texts required. In the preliminary investigation of mining the Web for parallel texts, Resnik (1998) proposed a simple method which sends Boolean query in the form:

$$anchor : language1 \quad AND \quad anchor : language2$$

to AltaVista to locate pages that point to a pair of pages which contain an anchor text indicating the language of its parallel text. This is the case for an “index.html” file which contains pointers to two parallel texts anchored as “English version” and “French version”, respectively. However this simple method can only catch a small part of all the parallel pages. A lot of other parallel pages do not satisfy this condition. When a large size corpus is required as in our case, we have to search more thoroughly. The PTMiner system is a Web agent that is designed for this purpose. It is designed such that it is mostly language independent. The system can be adapted to other language pairs with only minor modifications. We will explain its mining algorithm in this section and its implementation in the next.

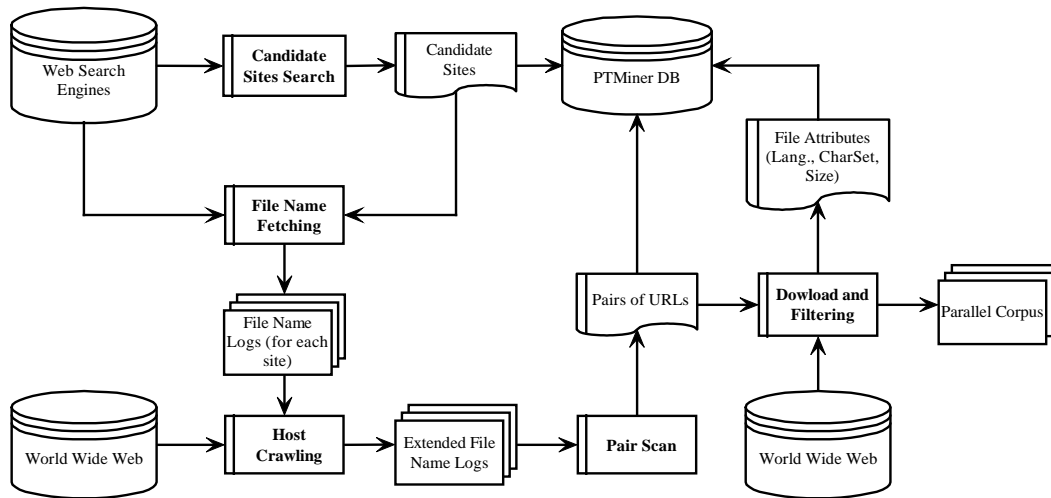


Figure 1: The workflow of the mining process.

We can summarize the mining process of PTMiner as the following steps (illustrated in Fig. 1):

- 1 Candidate sites search – We first determine the candidate sites that could contain parallel pages using the existing search engines on the Web;

- 2 File name fetching – For each candidate site, fetch the URLs of Web pages that are indexed by the search engines;
- 3 Host crawling – Starting from the URLs collected in the last step, crawl each candidate site separately for more URLs;
- 4 Pair scan – From the obtained URLs of each site, scan for possible parallel pairs;
- 5 Download and verifying – Download the parallel pages, determine file size, language, and character set of each page, and filter out non-parallel pairs.

## 2.1 About the Search Engines

Before we go through the details of each step, it is necessary to give a little survey on the Web search engines, which play an important role in this algorithm. PTMiner takes advantage of existing search engines to locate candidate sites and fetch URLs of each site to gather a starting URL set for host crawling.

Search engines constantly visit Web sites on the Internet in order to create catalogs of Web pages. They run automatically and index a huge amount of Web pages. Currently there are 10-20 major search engines on the Web with various index sizes and functions. A recent study (Anonymous, 1999a) estimates that current existing engines cover around 16% of all the Web. Among them AltaVista and Northern Light are the two largest ones in terms of pages indexed. They also have some features that are crucial to parallel text mining such as keyword searching within attributes and language identification.

Both AltaVista and Northern Light have language identification ability. It makes it possible for us to locate candidate bilingual sites. For Chinese-English parallel text search AltaVista is especially helpful not only because it is the only search engine that can identify Chinese pages but also because it can return results in all the character sets of Chinese. AltaVista does this by converting the pages it finds into Unicode, which can store characters for all languages.

## 2.2 Candidate Sites Search

We take advantage of the huge amount of Web sites indexed by search engines in determining candidate sites. This is done by sending some particular requests to the search engines. The requests are determined according to the following observations. In the sites where parallel texts exist, there are normally some pages in one language containing links to its version in the other language. Those links' anchor texts <sup>1</sup> usually indicate that. For example, in some English page there may be a link to its Chinese version with the anchor text “Chinese Version” or “in Chinese” (Fig. 2). The same phenomenon can be observed in Chinese pages. Chances are great that a site with parallel texts has such links in some of its documents. This fact is used as the criterion to determine candidate sites.

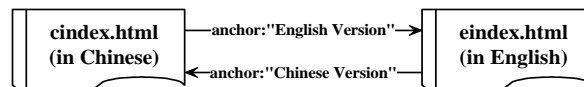


Figure 2: A pair of parallel pages linking to each other.

---

<sup>1</sup>An anchor text is a piece of text in a Web page which you can click to go to where it links to. To be instructive, it usually contains the key information of the linked page.

Therefore, to determine possible sites for English-Chinese parallel texts, we can request for English documents containing the following anchors:

*anchor : "english version" ["in english", ...]*

or Chinese documents containing the following anchors:

*anchor : "chinese version" ["in chinese", ...].*

From the two sets of pages obtained by the above queries we extract two sets of Web sites. The intersection or union of these two sets is then the candidate sites. Considering that search engines can only index part of all the pages of one site, we take the union of the two sets instead of the intersection. That is to say, a site is a candidate site when it is found to have either an English page linking to its Chinese version or vice versa. With this loosed criterion we could avoid losing chances because of the incompleteness of the search engines. Fig. 3 depicts the whole process of candidate searching.

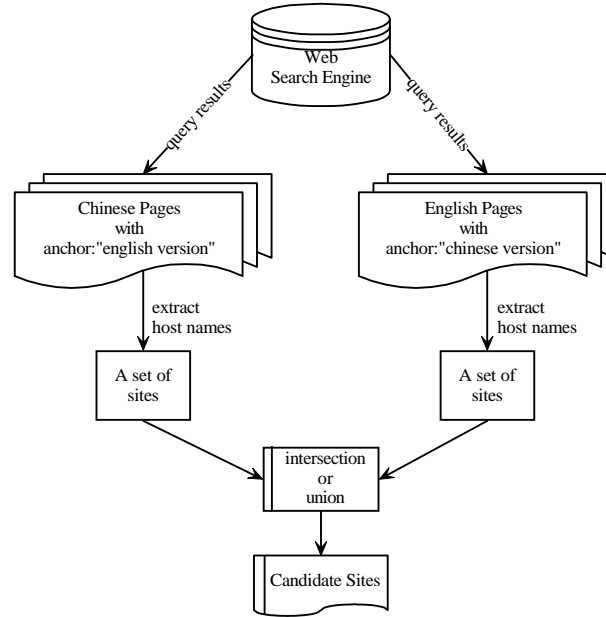


Figure 3: Candidate sites search process.

### 2.3 File Name Fetching

We assume that a pair of parallel texts exists on the same site. To search for parallel pairs from a site, PTMiner has to firstly obtain all (or at least part) on the HTML file names of the site. From these names pairs are scanned. The amount of file names we obtain from each site directly affects the amount of the pairs we can find. It is possible to use a Web crawler to explore the candidate sites completely. However, we can take advantage of the search engines again to accelerate the process. As the first step, we query the search engines in the form

*host : hostname*

to fetch the Web pages that they indexed from this site. If we only require a small amount of parallel texts, this result may be sufficient. For our purpose, we need to explore the sites more thoroughly by a host crawler because:

- the search engines do not index all the Web pages of a site;
- most search engines allow users to retrieve a limited number of documents (eg. 1000 in AltaVista).

Therefore, we continue our searching for files with a host crawler which uses the documents found by the search engines as the starting point.

## 2.4 Host Crawling

A host crawler is slightly different from a Web crawler. Web crawlers are used by the search engines to find new Web pages. They go through innumerable pages and hosts throughout the Web. A host crawler is a Web crawler that crawls documents on a given host only. A breadth-first crawling algorithm is applied in the host crawler of PTMiner. The principle is that when a link to an unexplored document on the same site is found in a document, it is added to a list that will be explored later.

## 2.5 Pair Scan

After collecting file names for each candidate site, the task left is to find out parallel pairs from them. A straightforward method would be to compare every couple of files. It is not applicable because of the following reasons:

- 1 The complexity of this algorithm has a quadratic order in terms of the number of the files. When we have to process thousands of files for each site, the computing time is not affordable.
- 2 All the files have to be downloaded locally to be processed. It gives a high load on the network and local file system. Needless to say, much time will be wasted on downloading pages that will turn out to be non-parallel.

Again, we try to use some heuristic rules to guess which files may be parallel texts before downloading them. The rules are based on “external features” of the documents. By external feature, we refer to the features that may be known without analyzing the contents of the file such as its URL, size, and date. This is in contrast with the “internal features” such as language, character set, and HTML structure which cannot be known until we have downloaded the page and analyzed its contents.

The heuristic criterion comes from the following observation: We observe that parallel text pairs usually have similar name patterns. The difference between the names of two parallel pages usually lies in a segment which indicates the language. For example, “file-ch.html” (in Chinese) vs. “file-en.html” (in English). The difference may also appear in the path, such as “.../chinese/.../file.html” vs. “.../english/.../file.html”. Fig. 4 shows some possible ways parallel pairs exit. The name patterns described above are commonly used by the webmasters to help organizing their sites. Hence we can suppose that a pair of pages with this kind of pattern are most probably a pair of parallel pages.

The general idea of the pair scanning algorithm is: For each file name, guess the possible corresponding file names for the other language, and check if such a file really exists.

To do this, we establish four lists of prefixes, and suffixes for the two languages. For example:

$$\begin{aligned} \text{English Prefix} = \{ & \text{"e", "en", "eng", "engl", "english",} \\ & \text{"e_", "en_", "eng_", "engl_", "english_",} \\ & \text{"e-", "en-", "eng-", "engl-", "english-"} \} \end{aligned}$$

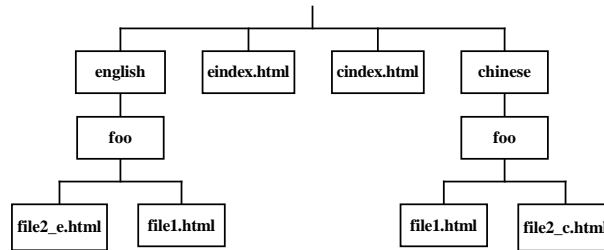


Figure 4: Parallel pairs in a directory tree.

For each file in one language, if a segment in its name corresponds to one of the language affixes, several new names are generated by changing the segment to the possible corresponding affixes of the other language. If a generated name corresponds to an existing file, then the file is considered as a possible parallel document of the original file.

Although in this process, many variations of file name are checked, the computing time for each file is a constant. The whole processing time increases linearly with the number of the files.

## 2.6 Filtering

We further examine the contents of the paired files to determine if they are really parallel according to some further external or internal features. This may further improve the pairing precision. The following methods have been used in our approach.

### 2.6.1 Text Length

Apparently a good parallel pair usually has similar file lengths. The simplest way is then to compare the lengths of the two files. The only problem is to set a reasonable threshold that can filter out mostly wrong pairs without sacrificing too many good ones, i.e., balance between recall and precision. The usual difference ratio depends on which language pair we are dealing with. For example, Chinese-English parallel texts usually have more difference ratio in length than that of English-French parallel texts. The filtering threshold has to be set from the actual observations.

### 2.6.2 Language and Character Set

It is also obvious that the two files of a pair have to be in the two languages of interest. By identifying language and character set we could filter out the pairs that do not satisfy this basic criterion. Some Web pages explicitly indicate the language and the character set. More often such information is omitted by authors. Therefore, we need a language identification tool for this task.

The SILC system (Isabelle, Foster, and Plamondon, 1997) is a language and coding identification system developed by the RALI laboratory of University of Montreal. It possesses a probabilistic model estimated on tri-grams. Using these models, the system is able to determine the most probable language and coding of a text among about 30 languages and more than a hundred codings.

In the current PTMiner system, we actually use length and language as the two criteria in the filtering process.

### 2.6.3 HTML Structure and Alignment

In the STRAND system (Resnik, 1998), the candidate pairs are evaluated by aligning them according to their HTML structures and computing confidence values. Pairs are assumed to be wrong if they

have too many mismatching markups or low confidence values.

Comparing HTML structures seems to be a sound way to evaluate candidate pairs since parallel pairs usually have similar HTML structures. However, we also noticed that parallel texts may have quite different HTML structures. One of the reasons is that the two files may be created using two different HTML editors. For example one is used for English, and another one for Chinese depending on the language handling capability of the editors. Therefore, caution has to be taken when measuring structure similarity numerically.

Parallel text alignment is still an experimental area. The measurement of confidence value of alignment is even more complicate. For example, the alignment algorithm we used in the training of the statistical translation model gives acceptable alignment results but no confidence value that we can “confidently” use as an evaluation criterion. Intuitively it would be possible to determine a confidence value from the probability of resulted alignment. This is still a research issue. Therefore, in the current algorithm this criterion is not used.

### 3 PTMiner – A Multi-Tier Distributed Text Miner

PTMiner is a multi-tier distributed Web mining agent. Many practical issues were considered in its design. We explain its features in the following.

- **Effectiveness** As introduced in the last section, the major steps of PTMiner are candidate sites search, file name fetching, host crawling, pair scan, download and filtering. Using this algorithm, PTMiner successfully generated a Chinese-English parallel text corpus.
- **Efficiency** The mining process could take long time, especially when host crawler is used. By adopting a distributed model, PTMiner can process several candidate sites in parallel and thus reduce greatly processing time.
- **Scalability** If needed, PTMiner can be used to generate very large Web-collected parallel text corpora provided that there are enough such parallel texts on the Web. If a small-size corpus is demanded, it can produce it with only the information from Web search engines (without using host crawler) in a short period of time.
- **Platform Independence** Taking advantage of the Java technology, most modules of PTMiner can run in various systems.
- **Maintenance** PTMiner is a distributed system involving various processes in various machines. It is however easy to start and restart. A centralized monitoring GUI interface is provided for user to watch clearly the working situation of all the processes, the content of the PTMiner database as well as the overall mining progress.
- **Result Analysis** The meta-information of mining results is stored in the PTMiner database which greatly facilitates the result analysis. The user can get all kinds of statistical data by flexibly using SQL queries.
- **Adaptability** The general idea behind the algorithm of PTMiner is language independent. Thus the system can be easily adapted to the mining of other language pairs.
- **Text Mining Architecture** With its modularized design, distributed model and application of database system, the system architecture of PTMiner is suitable for other text mining tasks. The separation of file name collection (search) and pair scan (discovery) is consistent with the general process of many text mining tasks.

### 3.1 PTMiner Architecture and Implementation

Fig. 5 illustrates the system architecture of PTMiner. Arrows indicates the directions of data flow between modules. It also shows how modules communicate with each other (through JDBC connection, CORBA remote method call or UDP packet).

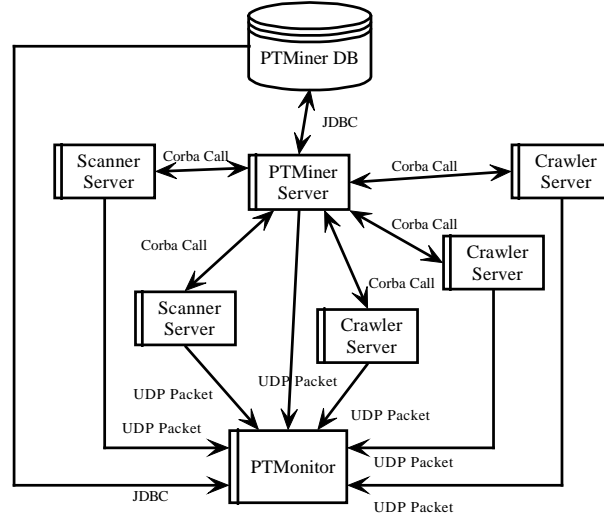


Figure 5: The architecture of PTMiner.

Briefly speaking, the central control unit of the system is the PTMiner server which reads candidate sites from the database and assigns them to Crawler and Scanner servers. Crawlers and Scanners reside in different machines. They register in the database when starting. Each site has to be passed to a Crawler server to collect file names of this site, and then a Scanner server to scan for parallel pairs. Mining results are stored into the database. PTMonitor is a central GUI interface receiving messages (in UDP packets) from all servers. It is also a viewer of the database content.

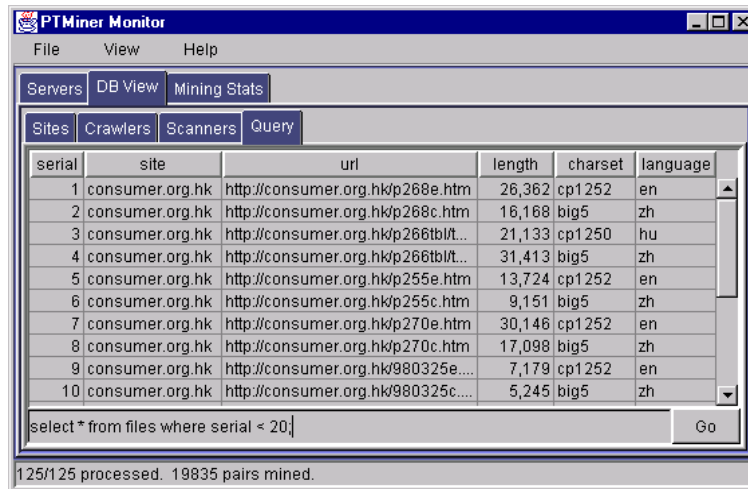
We now describe each module in more detail.

#### 3.1.1 PTMiner Database

The PTMiner database serves as the storage of intermediate and final mining results as well as working situation of the servers. For example, the *files* table (Fig. 6) contains the information of all the parallel pages, their URL, host, length, language and character set. The database is implemented with MySQL, a multi-threaded SQL database server. MySQL is a fast, robust, and easy-to-use database server which fits the needs of PTMiner very well. It is multi-threaded which makes parallel processing possible.

#### 3.1.2 Site Fetcher

The site fetcher module is a stand-alone program (not shown in Fig. 5) which implements the first step, candidate sites search, of the mining algorithm. As mentioned earlier, it sends queries to AltaVista and retrieves a set of candidate sites. The sites are stored into the database for future processing.



serial	site	url	length	charset	language
1	consumer.org.hk	http://consumer.org.hk/p268e.htm	26,362	cp1252	en
2	consumer.org.hk	http://consumer.org.hk/p268c.htm	16,168	big5	zh
3	consumer.org.hk	http://consumer.org.hk/p266tblt...	21,133	cp1250	hu
4	consumer.org.hk	http://consumer.org.hk/p266tblt...	31,413	big5	zh
5	consumer.org.hk	http://consumer.org.hk/p255e.htm	13,724	cp1252	en
6	consumer.org.hk	http://consumer.org.hk/p255c.htm	9,151	big5	zh
7	consumer.org.hk	http://consumer.org.hk/p270e.htm	30,146	cp1252	en
8	consumer.org.hk	http://consumer.org.hk/p270c.htm	17,098	big5	zh
9	consumer.org.hk	http://consumer.org.hk/980325e....	7,179	cp1252	en
10	consumer.org.hk	http://consumer.org.hk/980325c....	5,245	big5	zh

select \* from files where serial < 20;

Go

125/125 processed. 19835 pairs mined.

Figure 6: The *files* table.

### 3.1.3 Crawler Server

When a Crawler server is started, it first registers in the database and also notifies PTMonitor. It provides two methods that can be invoked by the PTMiner server, *fetch* and *crawl*. The *fetch* method takes the name of a candidate site and fetches file names from AltaVista and Northern Light. The result file name log will be read by the *crawl* method as the initial set for the host crawler. The *crawl* method can be skipped if host crawling is not necessary. Messages showing progress or exceptions encountered are sent to PTMonitor in UDP packets.

### 3.1.4 Scanner Server

Similarly to the Crawler server, the Scanner server registers itself in the database and sends messages to PTMonitor. Its *scan* method takes a site name, opens the corresponding file name log, and then scans for parallel pairs. The results are stored into the database.

### 3.1.5 PTMiner Server

As stated above, the PTMiner server is the central control unit of the system. It synchronizes the real workers, Crawler servers and Scanner servers, according to information in the database.

### 3.1.6 PTMonitor

The objective of PTMonitor is to facilitate the monitoring of the whole mining process. It provides the user with various kinds of information including:

- Messages from Crawler servers, Scanner servers, and the PTMiner server – As shown by Fig. 7, each server is represented by a node in the tree and each node has an associated text area. PTMonitor listens at some port for UDP packets from the servers. Once a packet has arrived, it determines where it comes from and appends the message at the corresponding text area.
- Contents of the database – With JDBC connection, PTMonitor reads from the database and display content of some tables (Fig. 6). It also gives users the freedom of constructing new queries.

- Statistical data – As shown in Fig. 8 PTMonitor can also display statistical data giving an overview to the mining progress.



Figure 7: PTMonitor showing messages from the servers.

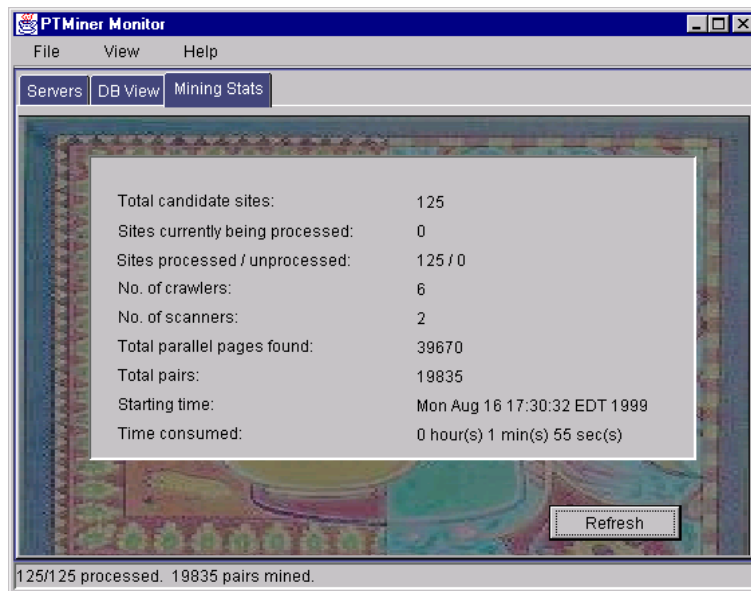


Figure 8: PTMonitor showing mining progress.

### 3.1.7 Allocating System Resource

One advantage of the PTMiner system is that most of its modules are implemented in Java which enables them to run in practically any machine. This feature brings convenience in distributing working objects. Most modules of PTMiner consume very low (around 1% or less) percentage of CPU time. Thus they could be established in any machine without influencing other users. The only module that costs most CPU time is the Scanner server. Fortunately, its actual working time on each site is much shorter than that of Crawler servers. We may need many Crawler servers but only one or two Scanner servers. Fig. 9 is an example of the working situation of PTMiner.

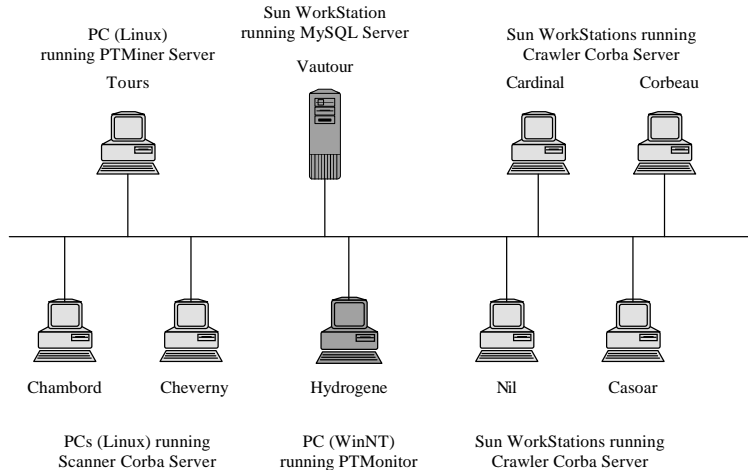


Figure 9: A network view of the PTMiner system.

## 4 Generated Corpora and Application in CLIR

In this section, we will introduce the results of parallel text mining and their application in translation model training and CLIR.

### 4.1 The Corpora

Using the above approach, two large-size parallel corpora have been produced so far. For English-French, 14198 pairs of parallel documents are retrieved from about 30% of 5474 candidate sites. The corpus actually includes 135M French texts and 118M English texts. Because English-French parallel Web pages are so popular, host crawling was not necessary and the mining time is relatively short, about 75 hours.

For English-Chinese, 185 candidate sites were searched from the domain *hk*. We limited the mining domain to *hk* because Hong Kong is a perfect English-Chinese bilingual city where high quality parallel Web sites exist. Because of the small amount of candidate sites, host crawler was used thoroughly to explore each site. The resulted corpus contains 14820 pairs of texts including 117.2M Chinese texts and 136.5M English texts. The mining process lasted about a week.

Length comparison and language identification were used in the evaluation. We examined 100-200 randomly picked pairs. The English-French corpus gives a precision of over 95% (*i.e.* more than 95% of the selected text pairs are really parallel). The English-Chinese corpus has a lower but acceptable precision of 90%.

### 4.2 Statistical Translation Model Training and the Evaluation Lexicons

Learning from the previous translation examples by human is often used in machine translation. Most work of this kind establishes probabilistic models from parallel corpora. Based on one of the statistical models proposed by Brown et al. (1993), the basic principle of our translation model is: given aligned sentences, if two words often co-occur in the source and target sentences, there is a high chance that they are translations of each other. Specifically, the model learns (from a large set of alignments) the probability,  $p(t|s)$ , of having a word  $t$  in the translation of a sentence containing a word  $s$ . For an input sentence, the model then calculates a sequence of words that are most probable in its translation.

In order to align mined documents into parallel sentences, a series of pre-training processing has

to be done to the raw texts before they can be used to train the translation model. The process depends on the language handled to some extent. For example, the major operations on Chinese-English corpus include coding scheme transformation (for Chinese), sentence level segmentation, parallel text alignment, Chinese word segmentation and English expression extraction.

The parallel Web pages we collect from various sites have different qualities. Some are highly parallel and easy to align while some can be very noisy. It is even harder to align English-Chinese parallel text because of the great difference between the syntactic structures and writing systems of the two languages. A number of alignment techniques have been proposed, varying from statistical methods (Brown, Lai, and Mercer, 1991; Gale and Church, 1991) to lexical methods (Kay and Röscheisen, 1993; Chen, 1993). What we adopted is the method of Simard et al. (1992). By considering both length similarity and cognateness as alignment criteria, the method is more robust dealing with noises than pure length-based methods. Cognates are identical sequences of characters in corresponding words in two languages. They are commonly found in English and French. In the case of English-Chinese alignment where there are no cognates shared by the two languages, only the HTML markups in both texts are taken as cognates. Because the HTML structures of a pair of pages are normally similar, the markups are found to be helpful for alignment.

To evaluate the precision of the English-Chinese translation model trained by the Web corpus, two sample lexicons (in two directions) of 200 words were examined. The 200 words for each lexicon are randomly picked from the training source. We examine the most probable translation for each word. The Chinese-English lexicon was found to have the precision of 77%. The English-Chinese lexicon has a higher precision of 81.5%. Part of the lexicons are shown in Fig. 10 where t/f indicates if the translation is true or false.

English word	t/f	Translation	Probability	Chinese word	t/f	Translation	Probability
a.m.	t	上午	0.201472	办事处	t	office	0.375868
access	f	公开	0.071705	保护	t	protection	0.343071
adaptation	t	适应	0.179633	报告	t	report	0.358592
add	t	补充	0.317435	备	t	prepare	0.189513
adopt	t	采用	0.231637	本地	t	local	0.421837
agent	t	代理人	0.224902	便会	f	follow	0.023685
agree	t	同意	0.36569	标准	t	standard	0.445453
airline	t	航空公司	0.344001	补校	f	adult	0.044959
amendment	t	修订	0.367518	不足	t	inadequate	0.093012
appliance	t	用具	0.136319	部分	t	part	0.313676
apply	t	适用	0.19448	财经	t	financial	0.16608
attendance	t	列席	0.171769	参观	t	visit	0.309642
auditor	f	审核	0.15011	草案	t	bill	0.401997
average	t	平均	0.467646	车辆	t	vehicle	0.467034
base_on	f	计算	0.107304	储蓄	t	saving	0.176695

Figure 10: Part of the evaluation lexicons.

Globally, the precision achieved is relatively high. If such a translation model is used for MT it is obviously not good enough. However, for CLIR such a precision may be acceptable. We will examine the effectiveness of using this translation model in CLIR in the next section.

We also notice that wrong translations usually have lower probabilities. For example, the translations of “access”, “based\_on” and “follow” in Fig. 10 have lower probabilities than the correct

translations. Therefore it is possible to calculate a confidence value based on the probability value obtained. We will investigate this problem in our future research.

Using similar statistical model, Wu et al. (1995) extracted large-scale English-Chinese lexicon from the HKUST corpus which contains a set of official documents of Hong Kong government. Translation model is however different from lexicons. It is difficult to compare the precisions obtained in the two studies because the test data are different. However, the precision reported in Wu et al. (1995) is similar to what we obtained.

### 4.3 Preliminary Results of CLIR Experiments

The final goal of this work is to test the performance in CLIR of the translation models trained by the Web parallel corpora. We give some preliminary results as the following.

#### 4.3.1 English-French CLIR

The experiment results of English-French CLIR have been given in the recent paper (Nie et al., 1999). Here we briefly show some results.

The experiments were actually conducted on the English AP and French SDA corpora in TREC6 and TREC7, using the *Smart* information retrieval system. Two translation models have been used, one trained with the parallel texts from the Web, and the other trained with a manually established parallel corpus - the Canadian Hansard which contains debates of the Canadian parliament in both English and French. The performances of the translation models are compared with that of monolingual IR, CLIR using an MT system and dictionary-based translation. A series of more recent results are shown in Table. 1 where performances are measured in terms of average precision. The MT system was found to be able to perform query translation with reasonable quality. The translation model gave lesser but still comparable precision. It out-performed the dictionary-based approach. It was also found that combining dictionary appropriately with the translation model can considerably improve the average precision.

	F-E (Trec6)	F-E (Trec7)	E-F (Trec6)	E-F (Trec7)
Mono-Lingual IR	0.2865	0.3202	0.3686	0.2764
MT System	0.3098 (107.0%)	0.3293 (102.8%)	0.2727 (74.0%)	0.2327 (84.2%)
Dictionary	0.1707 (59.0%)	0.1701 (53.1%)	0.2305 (62.5%)	0.1352 (48.9%)
Translation Model	0.2389 (82.5%)	0.3146 (98.3%)	0.2504 (67.9%)	0.2289 (82.8%)
Hansard Model	0.2166 (74.8%)	0.3124 (97.6%)	0.2501 (67.9%)	0.2587 (93.6%)

Table 1: English-French CLIR results.

In comparison with the model trained with a manually established parallel corpus (Hansard), we see that our model based on Web documents achieves comparable performance. This result is encouraging. It shows that the Web parallel texts may substitute a manual parallel corpus for CLIR.

#### 4.3.2 English-Chinese CLIR

Given the encouraging results of English-French CLIR, we continued to investigate the possibility of using translation model in English-Chinese CLIR.

The English test corpus (for C-E CLIR) is the AP corpus in TREC6 and TREC7. The short English queries in TREC6 were translated manually into Chinese and then translated back to English by the translation model.

The Chinese test corpus is the one used in TREC5 and TREC6 Chinese track. It contains both Chinese queries and their English translations. Our experiments on these two corpora gave the results shown in Table. 2.

	C-E	E-C
Mono-Lingual IR	0.3861	0.3976
Translation Model	0.1504 (39.0%mono)	0.1841 (46.3%mono)
Dictionary	0.1530 (39.6%mono)	0.1427 (35.9%mono)
TM + DICT	0.2583 (66.9%mono)	0.2232 (56.1%mono)

Table 2: English-Chinese CLIR results.

In both E-C and C-E CLIR, the translation model achieved around 40% of mono-lingual precision. To compare with the dictionary-based approach, we adopted a Chinese-English dictionary, CEDICT (Denisowski, 1999), and an English-Chinese online dictionary (Anonymous, 1999b) to translate queries. For each word of the source query, all the possible translations given by the dictionary are included in the translated query. The Chinese-English dictionary has about the same performance as the translation model, while the English-Chinese dictionary has lower precision than that of the translation model.

We also tried to combine the translations given by the translation model and the dictionary. In both C-E and E-C CLIR, significant improvements were achieved (as shown in Table. 2). The improvements show that the translations given by the translation model and the dictionary complement each other well for IR purpose. The translation model may give either exact translations or incorrect but related words. Even though these words are not correct in the sense of translation, they are very possibly related to the subject of the query and thus helpful for IR purpose. The dictionary-based approach expands a query in another dimension. It gives all the possible translations for each word including those that are missed by the translation model.

In the recent work of Kwok (1999), a MT system was used in E-C query translation of the same Chinese collections using a different IR system. The MT system gave more than 50% precision of monolingual IR result. By query expansion and using dictionary as compliment, over 70% precision of monolingual result was achieved. In our work, we also found it is helpful to combine the translation model with dictionary. The best E-C CLIR precision using the translation model (and dictionary) is 56.1%mono. It is lower than what Kwok achieved using MT system. However, the difference is not large.

In comparison with Table. 1, we notice that both the approaches based on dictionary and the translation model for C-E CLIR are not so good as for E-F CLIR. This result is predictable to some extent since the difference between English and Chinese is much larger than that between English and French. This problem exists in many phases of this work, from text alignment to query translation. We list some factors affecting CLIR precision as the following.

- The Web-collected corpus is noisy and it is difficult to align English-Chinese texts. The currently used alignment method has poorer performance than that in English-French alignment. This then leads to poorer performance of the translation model.

- For E-C CLIR, although queries in both languages are provided, the English queries were not strictly translated from the original Chinese ones. For example, 人权状况 (*human right situation*) was translated into *human right issue*. We can not expect the translation model to translate *issue* back to 状况 (*situation*).
- The training source and the collections are from different domains. The Web corpus are retrieved from the parallel sites in Hong Kong while the Chinese collection is from *People's Daily* and *Xinhua Daily* which are newspapers in mainland China. As a result, some important terms such as 最惠国 (*most-favored-nation*) and 一国两制 (*one-nation-two-systems*) in the collection are not known by the model.

## 5 Summary

The objective of this work is to investigate the feasibility of using statistical translation model trained by Web-collected corpus in CLIR. In this paper, we have introduced the algorithm and implementation we used in parallel text mining, and reported some results we obtained in CLIR using translation models. Although there are rooms for improvement, we can draw some conclusions.

- It is possible to obtain parallel corpora from the Web. With the PTMiner system, we successfully established two large-scale parallel corpora in relatively short time. The system can be easily adapted to other language pairs.
- The English-French translation model showed comparable CLIR performance to a model trained with a manually established parallel corpus and an MT system.
- For English-Chinese, despite the noisy nature of the corpus and great language difference, the evaluation lexicons generated by the translation model gave acceptable precision. While experiments in CLIR didn't show results as encouraging as those in English-French CLIR, those can be improved in various ways such as:
  - Improve the alignment method by adapting cognate definitions to HTML markups and incorporating a lexicon.
  - Use some query expansion techniques to include synonyms.
  - Remove some common functional words in translated queries. These words do not help for IR purposes and may decrease the precision.

We expect to prove in the near future that a fine-tuned English-Chinese translation model can provide satisfying query translations for CLIR uses.

## References

- Anonymous. 1999a. New search engine to snare all the Web. <http://techweb.com/wire/story/TWB19990809S0002>, August.
- Anonymous. 1999b. Sunrain.net - English-Chinese dictionary. [http://sunrain.net/r\\_ecdict\\_e.htm](http://sunrain.net/r_ecdict_e.htm).
- Balabanovic, M., Yoav Shoham, and Y. Yun. 1995. An adaptive agent for automated web browsing. *Journal of Visual Communication and Image Representation*, 6(4).
- Brown, C. M., B. B. Danzig, D. Hardy, U. Manber, and M. F. Schwartz. 1994. The harvest information discovery and access system. In *Proc. 2nd International World Wide Web Conference*.

- Brown, P. F., J. C. Lai, and R. L. Mercer. 1991. Aligning sentences in parallel corpora. In *29th Annual Meeting of the Association for Computational Linguistics*, pages 89–94, Berkeley, Calif.
- Brown, P. F., S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311.
- Chen, S. F. 1993. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31th Annual Meeting of the Association for Computational Linguistics*, pages 9–16, Columbus, Ohio.
- Denisowski, Paul. 1999. Cedict (chinese-english dictionary) project. <http://www.mindspring.com/~paul.denisowski/cedict.html>.
- Doorenbos, R. B., O. Etzioni, and D. S. Weld. 1996. A scalable comparison shopping agent for the World Wide Web. Technical Report Technical Report 96-01-03, Dept. of Computer Science and Engineering, University of Washington.
- Gale, William A. and Kenneth W. Church. 1991. A program for aligning sentences in bilingual corpora. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 177–184, Berkeley, Calif.
- Isabelle, P., G. Foster, and P. Plamondon. 1997. SILC: un système d’identification de la langue et du codage. <http://www-rali.iro.umontreal.ca/ProjetSILC.en.html>.
- Kay, M. and M. Röscheisen. 1993. Text-translation alignment. *Computational Linguistics*, 19:121–142.
- Konopnicki, D. and O. Shmueli. 1995. W3QS: A query system for the World Wide Web. In *Proc. of the 21th VLDB Conference*, pages 54–65, Zurich.
- Kwok, K. L. 1999. English-chinese cross-language retrieval based on a translation package. In *Workshop of Machine Translation for Cross Language Information Retrieval, Machine Translation Summit VII*, Singapore.
- Nie, Jianyun, Michel Simard, Pierre Isabelle, and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining parallel texts from the Web. In *ACM SIGIR’99*, pages 74–81, August.
- Perkowitz, M. and O. Etzioni. 1995. Category translation: learning to understand information on the internet. In *Proc. 15th International Joint Conference on AI*, pages 930–936, Montreal, Canada.
- Resnik, Philip. 1998. Parallel stands: A preliminary investigation into mining the Web for bilingual text. In *AMTA’98*, October.
- Simard, Michel, George F. Foster, and Pierre Isabelle. 1992. Using cognates to align sentences in bilingual corpora. In *Proceedings of TMI-92*, Montreal, Quebec.
- Wu, Dekai. 1995. Large-scale automatic extraction of an English-Chinese lexicon. *Machine Translation*, 9(3-4):285–313.
- Zaiane, O. and J. Han. 1998. WebML: Querying the World-Wide Web for resources and knowledge. In *Proc. (CIKM’98) Int’l Workshop on Web Information and Data Management (WIDM’98)*, pages 9–12, Bethesda, Maryland, November.