中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-07-017

BibPro: A Citation Parser Based on Sequence Alignment Techniques

Kai-Hsiang Yang, Chien-Chih Chen, Jan-Ming Ho

Oct. 30, 2007 || Technical Report No. TR-IIS-07-017

http://www.iis.sinica.edu.tw/page/library/LIB/TechReport/tr2007/tr07.html

BibPro: A Citation Parser Based on Sequence Alignment Techniques

Kai-Hsiang Yang
Institute of Information Science,
Academia Sinica,
Taipei,Taiwan
khyang@iis.sinica.edu.tw

Chien-Chih Chen
Institute of Information Science,
Academia Sinica,
Taipei,Taiwan
rocky@iis.sinica.edu.tw

Jan-Ming Ho
Institute of Information Science,
Academia Sinica,
Taipei,Taiwan
hoho@iis.sinica.edu.tw

ABSTRACT
The dramatic increase in the number of academic publications has led to a growing demand for efficient organization of the resources to meet researchers' specific needs. As a result, a number of network services have compiled databases from the public resources scattered over the Internet. Furthermore, because the publications utilize many different citation formats, the problem of accurately extracting metadata from a publication list has also attracted a great deal of attention in recent years. In this paper, we extend our previous work by using a gene sequence alignment tool to recognize and parse citation strings from publication lists into citation metadata. We also propose a new tool called BibPro. The main difference between BibPro and our previously proposed tool is that BibPro does not need any knowledge databases (e.g., an author name database) to generate a feature index for a citation string. Instead, BibPro only uses the order of punctuation marks in a citation string as its feature index to represent the string's citation format. Second, by using this feature index, BibPro employs the Basic Local Alignment Search Tool (BLAST) to match the feature's citation sequence with the most similar citation formats in the citation database. The Needleman-Wunsch algorithm is then used to determine the best citation format for extracting the desired citation metadata. By utilizing the alignment information, which is determined by the best template, BibPro can systematically extract the fields of author, title, journal, volume, number (issue), month, year, and page information from different citation formats with a high level of precision. The experiment results show that, in terms of precision and recall, BibPro outperforms other systems (e.g., INFOMAP and ParaCite). The results also show that BibPro scales very well.

Categories and Subject Descriptors
H.3.3 [Information Systems]: Information Search and Retrieval.

General Terms
Algorithms, Documentation.

Keywords
Citation Extraction, Sequence Alignment, Digital Library, Data Integration, Data Cleaning.

1. INTRODUCTION
Parsing citations is essential for integrating bibliographical information published on the Internet. The technique can also be used in other applications, such as field-based searching, author analysis of publications, and citation analysis [5]. However, it is difficult to design a system that can automatically parse citations scattered over the Internet because, in addition to the problem of technical typing errors, there are many different citation styles/formats. Citations can include a number of fields (e.g., author, title, publication information) arranged in many different formats depending on the type of publication (e.g., book, journal, conference paper, research report, or technical report). Therefore, extracting the required fields from citations is a challenging task.

Numerous works on extracting metadata from citations are reported in literature [1-12, 16]. The approaches can be roughly classified into three categories: learning-based, template-based and rule-based approaches.

Learning-based methods utilize machine learning techniques (e.g.,. the Hidden Markov Model (HMM) [7, 8], Support Vector Machines (SVM) [6], and Conditional Random Fields (CRF) [5]). Among them, CRF achieves the best performance with an overall word accuracy of 95.37% on the Cora reference dataset [5, 17], which contains 500 references covering thirteen fields: author, title, editor, book title, date, journal, volume, tech, institution, pages, location, publisher, and note.

Template-based methods utilize template databases with various styles of citation templates (e.g., ParaCite [16] and INFOMAP [11,12]). ParaCite has been integrated with the EPrints.org software, and links with CiteBase, RefLink, and ISI Web of Science [16] are currently under consideration. INFOMAP is a hierarchical template-based reference metadata extraction method with an overall average accuracy level of 92.39% for the six major citation styles detailed in [12].

Rule-based methods are widely used in real-world applications. For example, CiteSeer [1-4] is a well-known search engine and digital library that uses heuristics to extract certain subfields. It identifies titles and author names in citations with roughly 80% accuracy and page numbers with roughly 40% accuracy [1].

In a previous work [10], we proposed a template-based citation parser that achieved approximately 80% precision, but it had a number of drawbacks. First, because template construction relies on an author name database for the token matching process, the size of the author database directly affects the accuracy of the

template database. Second, the parser uses several heuristic rules to transform a citation string into templates, but the rules may only work for some special cases. The same problem arises when metadata is extracted from a citation string by referencing a template. Finally, during the matching process, there is high probability of mismatching templates with other templates that have same similarity score. We call this the "template conflict" problem. Generally, the larger the template database, the more serious the problem will be.

In this paper, we propose a new framework, called BibPro, for the citation parser. BibPro retains the advantages of our previous work (e.g., it uses protein sequences to represent citations and the Basic Local Alignment Search Tool (BLAST) to find similar templates), and resolves the weaknesses. Instead of relying on an author name database and heuristic rules, BibPro uses the order of punctuation marks in a citation string as a feature to represent the string's citation style. Furthermore, to find the template with the highest similarity score, we use the Needleman-Wunsch algorithm [15] in conjunction with BLAST to extract metadata from citation strings and align the features (a protein sequence) with the templates in our template database. In other words, BibPro extracts metadata systematically from the strings by referencing the alignment information of the matched template. Because of these two modifications, BibPro does not need any heuristics, and thus overcomes the template conflict problem. Figure 1 shows an example of extracting metadata from a citation string, where BibPro can be used to extract several common fields, such as author, title, journal, volume, number (issue), page, month, and year information, from a citation string.

The remainder of this paper is organized as follows. In Section 2, we explain the concepts behind BibPro and describe its architecture. In Section 3, we detail the experiment results and compare BibPro with several related works. Then, in Section 4, we present our conclusions and discuss some interesting directions for future research.

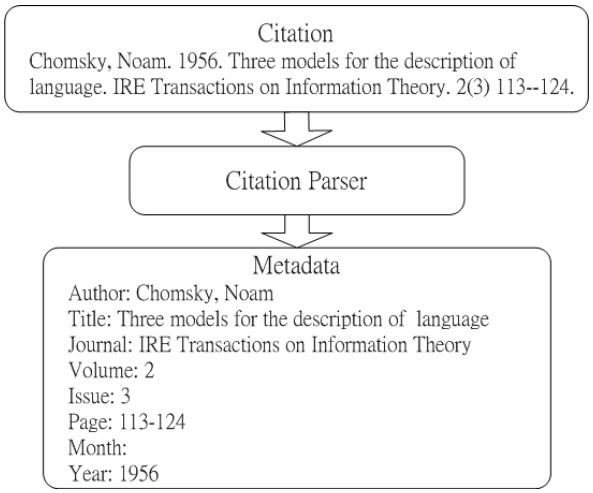

Figure 1. Extraction metadata from a citation string.

2. BIBPRO: CITATION PARSER

2.1 Basic Ideas

Our system is based on two concepts. The first uses a protein sequence to represent a citation string. We split a citation string into several tokens and use an amino acid symbol to represent each token. Figure 2 shows an example of a citation string transformed into a protein sequence "AAADTTTTDLLLLDYRPHS". When transforming a citation string into a protein sequence, Bibpro only transfers important features (e.g., the order of the fields and the field separators) from the citation string to the protein sequence. Redundant information is filtered out to simplify the problem and accelerate the parsing process. A protein sequence is designed to capture some features of the citation string. The sequence is then matched with previously known templates by BLAST [13, 14], a well-developed protein sequence matching program that searches protein sequence databases for sequences that are most similar to the target sequence. After a template has been selected as the first priority by BLAST, Bibpro partitions and extracts the desired metadata from the citation string.

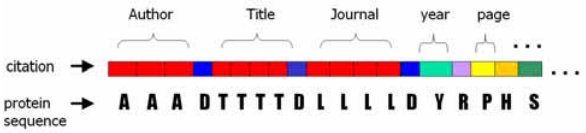

Figure 2. Transforming a citation string to a protein sequence.

Because citation strings of the same style have similar punctuation marks and reserved words, the order of the punctuation marks in a string must be fairly significant to identify the citation style. Our second concept utilizes this structural property as a feature index in BLAST to help match citation styles and parse citation strings according to their respective citation styles.

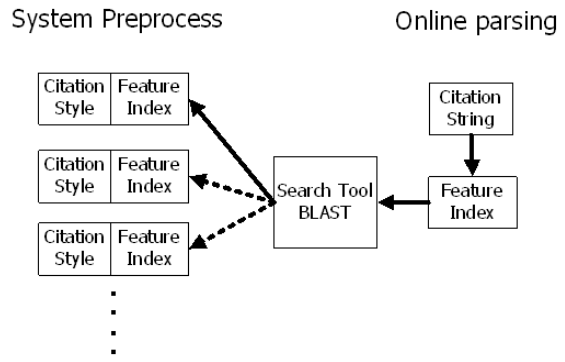

Figure 3. System preprocess and online parsing phases.

Based on these two concepts, Bibpro consists of two phases: a system preprocessing phase and an online parsing phase. The goal of the first phase is to generate feature indices for all previously known citation styles in advance so that BLAST will be able to better match citation styles in the second phase (see Figure 3a.). During the online parsing phase, BibPro uses BLAST [13, 14] to

find a citation style with a feature index similar to that of the citation string. BibPro then accurately extracts metadata from this citation string. In the following subsections, we introduce each component and the feature indices used in our system.

2.2 Form Translation

In order to use BLAST to match similar citation styles, we need to transform incompatible citation strings into compatible protein sequences. Therefore, we need to consider a number of questions:

- How many symbols can be used in a protein sequence?
- How many fields should be extracted from a citation string?
- How do we transform a citation string into a protein sequence and retain its citation style information?

Having considered the above questions, we created an encoding table to define the relationships between the tokens in a citation string and the symbols in protein sequences, as shown in Table1.

Table 1. Encoding Table

A: Author	N: numeral
T: Title	Q: @ # $ % ^ & * + = \ | ~ _ / ! ? 。
L: Journal	I: ([{ < 「
F: Volume value	K:)] } > 」
W: Issue value	D: .
H: Page value	G: " “ ”
X: noise (unrecognized token)	R: ,
M: Month	C: - :
Y: Year (number: 1900-2010)	E: ' `
S: Issue key. e.g. “no”, “No”	Z: ;
P: Page key. e.g. “pp”, “page”	B: blank (use one “B ” to replace continuous “X ”)
V: Volume key. e.g. “Vol”, “vo”	

The design of the encoding table is based on the following observations:

- BLAST can only process sequences with 23 different symbols, so we use these 23 symbols to represent different fields, and use field separators to keep the citation style information in sequence.
- The most common fields in citation strings are: author, title, journal, volume, number, page, issue, month and year. Thus, we focus on extracting these fields from citation strings and assign a symbol to represent each field.
- The most common reserved words used in citation string are: "vo", "vol", "no", "NO", "pp", and "page". Since these words are also used to separate fields, we use a symbol to represent each kind of reserved word.
- The punctuation marks normally used to separate fields are: " , ", " . ", " ; ", " : ", " " " and " ' ". Hence we assign each punctuation mark a symbol to represent it.
- Brackets and parentheses are synonymous in citation strings, so we use one symbol to represent both.
- Several kinds of punctuation marks appear in the title field, such as: " - ", " ! ", " ? ". However, we only use one symbol to represent all of them because they do not separate fields.

Figures 4 and 5 show examples of citation strings transformed into protein sequences. Figure 4 shows that when a citation string’s partitions are pre-determined, we can correctly label each token. Because this form is the correct encoding of the protein sequence, we call it the "RESULT FORM", and if we can correctly transform a citation string to its RESULT FORM, it is easy to extract information for the citation string. However, when parsing a given citation string online, as Figure 5 shows, we can only label each token based on its content. If we find unrecognized tokens, we replace them with a "X". We call this protein sequence the "BASE FORM". Thus, the process of parsing citation strings can be stated more simply as the transformation of target citation strings from their BASE FORM into their RESULT FORM.

To transform a citation string from its BASE FORM into its RESULT FORM, we need to know its citation style. For this reason, we define several forms of a protein sequence for our mining process:

- STYLE FORM: To store information about different citation styles, BibPro generates a style form for each style in the preprocessing phase. The style form is then used to partition citation strings during the online parsing phase. The RESULT FORM can represent style information for specific citation strings, but it has a lot of excess information, such as the length of author, title, and journal fields. We therefore condense the redundant information in the RESULT FORM by using one of each symbol to represent the above fields, as shown in Figure 6. This sequence, called the "STYLE FORM", is used to represent a citation style.
- INDEX FORM: To recognize the style of a citation string, we need to define a form to represent the feature index shown in Figure 3. We only use the order of punctuation marks in citation strings as the feature index. Figure 7 shows an example of the transformation of a citation string from its BASE FORM into what we call its “INDEX FORM” by removing all other unrecognized tokens. The INDEX FORM is the protein sequence that BLAST will try to match with similar sequences in the template database, so it is like an index used in a conventional database.
- ALIGN FORM: Parsing a citation string online is a difficult task, even after BLAST has determined its corresponding template. This is because the system only knows the INDEX and STYLE FORM of the template, but many fields of the citation like the author, title, and journal fields may contain punctuation marks. To extract these fields correctly, we have to remove all the punctuation marks in them. Besides, the STYLE FORM only shows the order of fields, not the number of authors that appear in a citation string. Hence, we cannot partition the citation string correctly with the STYLE FORM alone. For this reason, we use a knowledge database that includes author and journal information to mark and group author and journal field tokens in citation strings during the online parsing phase. Figure 8 shows an example of author and journal field grouping during the transformation of the BASE FORM into what we call the "ALIGN FORM". The knowledge

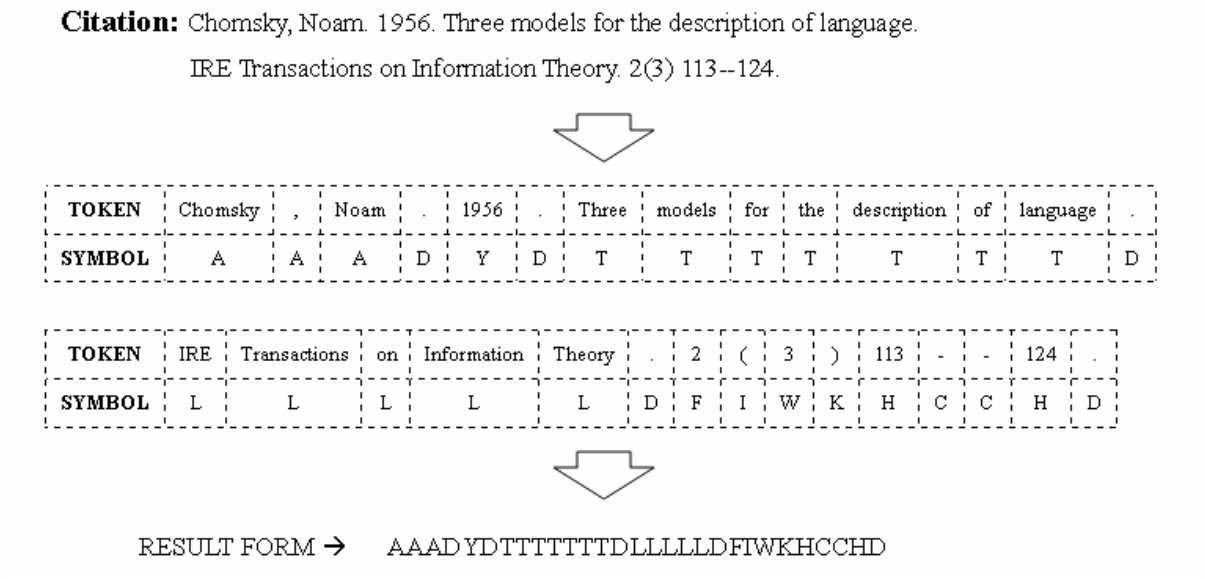

Figure 4. Transforming a citation string into a RESULT FORM.

database is not essential to our system, but it helps improve the level of accuracy. The ALIGN FORM is only used to process citation strings in the online parsing phase.

Remember that the goal of BibPro is to correctly parse the given citation string, that is, to transform the given citation string to its final RESULT FORM. When parsing the citation string, BibPro does not know the RESULT FORM but could generate an answer by using the BLAST tool to match similar citation strings in our database. The matching process is based on the citation string's INDEX FORM, so that BibPro can find out candidate citation strings with similar INDEX FORM. According to these candidates' STYLE FORM, BibPro then uses the Needleman-Wunsch algorithm to perform global alignment between the STYLE FORM and the ALIGN FORM, and extract metadata from the given citation string. Figure 9 shows the result of global alignment. With the alignment, BibPro is able get the RESULT FORM from the ALIGN FORM by adding "A" (author), "L"

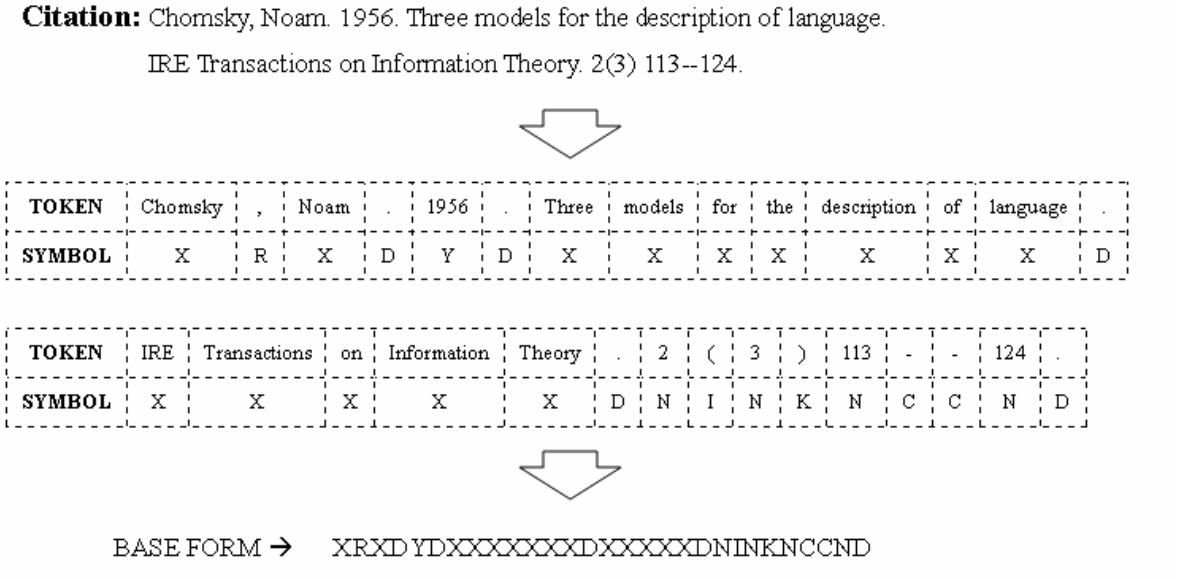

Figure 5. Transforming a citation string into a BASE FORM.

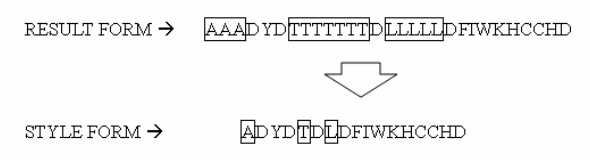

Figure 6. Transforming RESULT FORM into STYLE FORM.

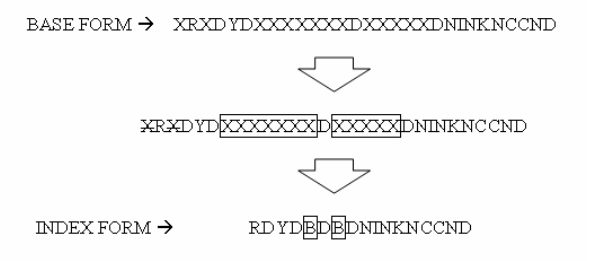

Figure 7. Transforming BASE FORM into INDEX FORM.

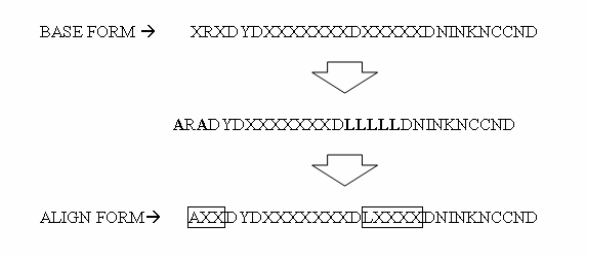

Figure 8. Transforming BASE FORM into ALIGN FORM.

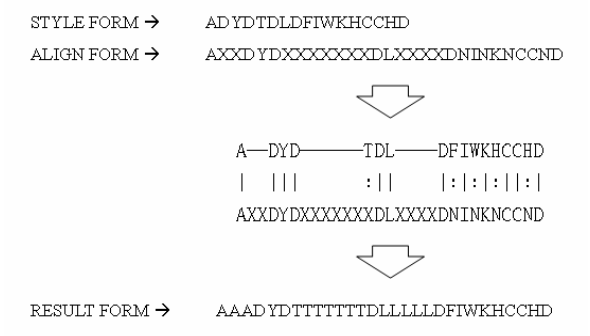

Figure 9. Aligning STYLE FORM and ALIGN FORM to get a RESULT FORM.

(journal), and "T" (title) in the correct positions and by changing "N" to its corresponding amino acid (e.g., an amino acid "N" may become F [volume number], "W" [issue number] or "H" [page number]]. Finally, by checking the original citation string and the RESULT FORM, we can parse all the metadata correctly.

2.3 System Architecture and Design

Figure 10 shows a simple flow diagram of BibPro's processes. In the first step, BibPro collects data, including citation strings and their corresponding partition information, from the Internet and uses it to build a template database. The system is then able to provide an online citation parsing service. Hence, we can divide BibPro into two basic systems: a template generating system and a parsing system.

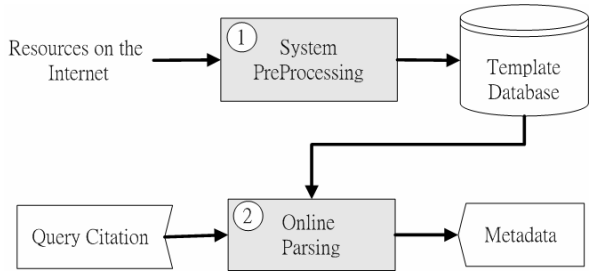

Figure 10. System work flow of BibPro.

2.3.1 Template Generating System

The goal of the template generating system is to construct a large database of templates, each of which represents a citation style. We divide the template generating process into two phases. The first phase collects data, including citation strings and their corresponding metadata (partition answers), from the Internet. The second phase uses this data to build the templates. We developed programs to retrieve BibTeX files from the Internet. Since the files are field-based, we can easily parse them to get the metadata for a citation string. Then, we use the title field as a search query to search for a citation in CiteSeer or another search engine, e.g., Google. In this way, we can get many citation strings and their corresponding metadata, as shown in Figure 11.

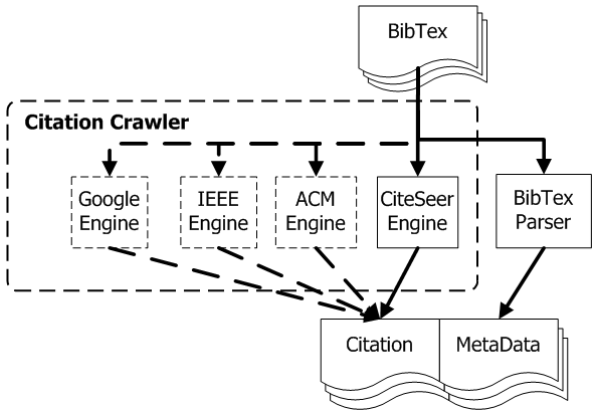

Figure 11. Process of collecting template data.

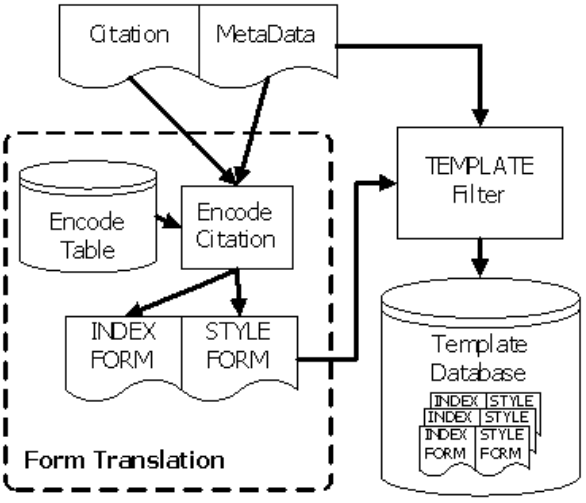

Figure 12. The structure of building template database.

BibPro can then build the template database. Because each citation string's partition answers are known, it is easy to transform citation strings into their STYLE and INDEX FORMs, as described in Section 2.2. We treat these two sequences as one record in the template database. However, we can not store the record in the database directly because the data collected from a citation string may be inconsistent with its metadata. Moreover, our token-based form translation may encounter problems if different fields share the same token. For this reason, we designed a template filter to ensure that a template is consistent with its original citation string. The template filter is designed according to some simple rules (e.g., the author, title and journal fields can not appear more than once in a citation string). This filter enables BibPro to build the template database automatically. Figure 12 shows the process of building a template database.

2.3.2 Parsing System

Once the template database has been compiled, BibPro can parse a citation string on-the-fly. Like the template-generating process,

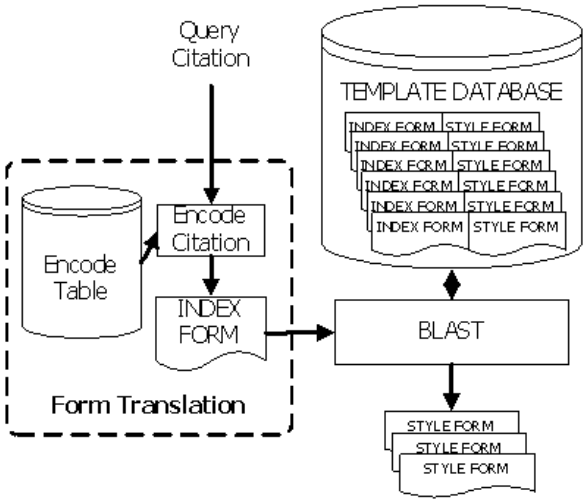

Figure 13. Process of matching template by the BLAST.

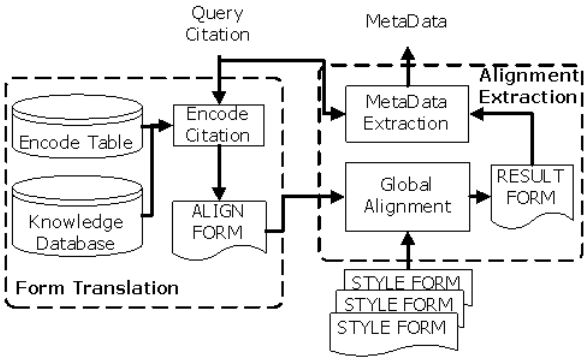

Figure 14. Process of extracting metadata.

the parsing process can also be divided into two phases. In the first phase, BLAST searches the template database for the template most similar to that of the citation string. Then, the encoding table transforms the citation string into its INDEX FORM and ALIGN FORM. BibPro then uses the INDEX FORM as a query string for BLAST to match against and find the corresponding STYLE FORM. Since BLAST needs a scoring table to evaluate the search results, we modified the score table to fit the encoding table's definitions. The complete process is illustrated in Figure 13.

A problem may arise if the template database becomes too large because BLAST is likely to match many STYLE FORMs with the same similarity score. To solve this problem, in the second phase, BibPro uses the Needleman-Wunsch algorithm to compute the ALIGN FORMs of all the matched STYLE FORMs with the same score. Since the algorithm also needs a score table to evaluate the score, we added the author and journal information, which is included in both the ALIGN and STYLE FORMs, to the score table (see Appendix 1) . After calculating the scores, BibPro chooses the STYLE FORM with the highest score and thereby avoids the template conflict problem. Note that during the alignment computation step, BibPro continues to extract metadata from the citation string. Figure 14 illustrates the processes in the second phase.

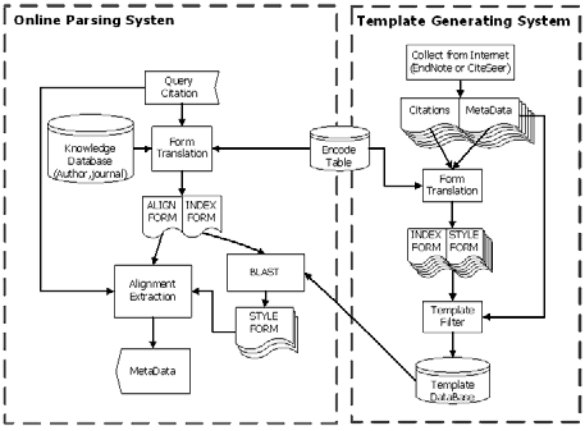

Figure 15. System structure of BibPro.

Figure 15 shows the combination of the parsing system and template generating system in BibPro.

3. EXPERIMENTS AND ANALYSIS

To conduct a comprehensive evaluation, we compared BibPro with several other systems. Because we cannot obtain their source codes, and each of them only provides its own dataset and performance measurements, we used their datasets with corresponding performance measurements when comparing with different systems, in order to have a fair comparison.

3.1 Datasets

We chose three datasets for our experiments. The first dataset, which was compiled by [12], comprised six citation styles, namely, JMIS, ACM, IEEE, APA, MISQ, and ISR, and included 160,000 citation strings. We randomly selected 10,000 strings to build the template database and another 10,000 citation strings for testing. We refer this dataset as D1.

The second dataset was created by the Cora project [5, 17]. It comprises 500 citation strings, each of which contains 13 fields: author, title, editor, book title, date, journal, volume, tech, institution, pages, location, publisher, and note. We used 350 citation strings for training and the remaining 150 for testing. We refer this dataset as D2.

The third dataset was obtained from CiteSeer and contained 6,500 citation strings. We developed two programs: one to retrieve the BibTeX files from each citation string on the Internet; and the other for choosing the title field to search the citations in CiteSeer so that we could compile the citation strings and their corresponding metadata. We used 2,500 citation strings for training and 4,000 for testing. We refer this dataset as D3.

The citation strings in D1 are more regular than those in D2 and D3 because they were generated from existing data. Moreover, the citation styles only differ in the order of the fields and the separators of the fields. In other words, there are no variations in the citation string formats. The D2 dataset is more complex than D1; however, it only contains 500 records, which are insufficient to express every type of citation style. We therefore collected real data from the Internet to generate the D3 dataset, which is more varied and fits real-world applications better.

3.2 Performance Measurements

We use different performance measurements for the datasets in our experiments. The first measurement, which is also used in [11, 12] is

$$\text{Accuracy} = \frac{\text{Number of correctly extracted fields}}{\text{Total number of fields}}$$

This accuracy measurement, called EVAL1, is used to evaluate the system's performance on the D1 dataset.

The second measurement, defined in [5], is calculated as follows:

- Word accuracy: assume that A is the number of true positive words, B is the number of false negative words, C is the number of false positive words, D is the number of true negative words, and A + B + C + D represents the total number of words. The word accuracy is calculated by: $\frac{A + D}{A + B + C + D}$
- F1-measure: The Precision, Recall and F1 measures are defined as follows.

$$\text{Precision} = \frac{A}{A + C} \quad \text{Recall} = \frac{A}{A + B} \quad \text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

This measurement, called EVAL2, is used to evaluate the system's performance on the D2 dataset.

The third measurement is used for the D3 dataset. For this dataset, the metadata in BibTeX that we collected from the Internet should be consistent with the metadata of the citation string. Unfortunately, some of the BibTeX metadata from the Internet does not fit the corresponding citation string. To resolve this problem, we developed the following measurement to determine whether the data is correctly parsed.

$$\text{Field Precision} = \frac{\#[\text{Token}_{\text{parsed field}} \cap \text{Token}_{\text{BibTex field}}]}{\#[\text{Token}_{\text{query citation}} \cap \text{Token}_{\text{BibTex}}]}$$

where $\text{Token}_{\text{parsed field}}$ denotes tokens that appear in the parsed subfield; $\text{Token}_{\text{query citation}}$ denotes tokens that appear in the query citation string; $\text{Token}_{\text{BibTex field}}$ denotes tokens that appear in a specific subfield in the BibTeX file; and $\text{Token}_{\text{BibTex}}$ denotes all tokens that appear in the BibTeX file.

The denominator represents the number of the tokens in both the citation string and the BibTeX file, while numerator represents the number of correctly parsed tokens. We use this measurement, called EVAL3, to compare BibPro with ParaCite.

Using these three measurements, we can compare BibPro with other systems and derive more reliable experiment results.

3.3 Experimental Results

3.3.1 Comparison with INFOMAP

The first experiment compares BibPro with INFOMAP [11, 12]. We used EVAL1 on the D1 dataset; the results are shown in Table 2. BibPro outperforms INFOMAP with an overall average accuracy for the six styles of 97.68% versus 92.39% for INFOMAP. Furthermore, in all fields, except the journal field, BibPro achieves a higher average accuracy level than INFOMAP. More specifically, BibPro is at least 5% more accurate in the author, title, issue and page fields. Similarly, of the six different citation styles mentioned earlier, BibPro excels in all styles except the MISQ style. To verify the scalability, we use the same template database and evaluation to test the full 150,000 citation strings, and the overall average accuracy is 94.85% as shown in Table 3. The results show that BibPro can achieve a better performance than INFOMAP. Furthermore, it is reliable when the dataset is regular and clean.

Table 2. Extraction results of BibPro and INFOMAP on D1 using EVAL1.

	Citation Style	Author	Title	Journal	Volume	Issue	Year	Page	Overall Avg.
Bib Pro	APA	99.67%	96.38%	97.06%	98.99%	98.12%	99.42%	98.71%	**98.33%**
	IEEE	98.72%	98.12%	99.12%	99.30%	98.39%	99.40%	98.40%	**98.78%**
	ACM	97.14%	95.01%	93.93%	97.19%	97.03%	98.88%	97.92%	**96.73%**
	ISR	99.48%	96.17%	96.96%	99.15%	98.39%	99.35%	98.55%	**98.29%**
	MISQ	98.59%	97.99%	98.98%	99.41%	98.61%	99.54%	98.83%	**98.85%**
	JMIS	91.95%	87.90%	90.46%	99.23%	98.03%	99.46%	98.76%	**95.11%**
	Avg.	**97.59%**	**95.26%**	**96.09%**	**98.88%**	**98.09%**	**99.34%**	**98.53%**	**97.68%**
INFO MAP	APA	92.32%	71.80%	94.33%	97.39%	84.92%	96.48%	95.09%	**90.33%**
	IEEE	94.17%	89.05%	92.07%	95.45%	84.49%	97.18%	89.81%	**91.75%**
	ACM	88.36%	91.10%	99.41%	80.28%	87.73%	96.47%	83.95%	**89.61%**
	ISR	91.93%	78.33%	95.32%	95.28%	87.00%	96.34%	90.61%	**90.69%**
	MISQ	97.73%	97.92%	100.00 %	99.99%	99.98%	99.94%	99.64%	**99.31%**
	JMIS	76.55%	72.57%	99.99%	99.98%	99.97%	99.93%	99.69%	**92.67%**
	Avg.	**90.18%**	**83.46%**	**96.85%**	**94.73%**	**90.68%**	**97.72%**	**93.13%**	**92.39%**

Table 3. Extraction results of BibPro on 150,000 citation strings using EVAL1.

Citation style	Author	Title	Journal	Volume	Issue	Year	Page	Overall Avg.
APA	99.01%	85.55%	89.40%	98.68%	96.97%	99.48%	98.34%	**95.34%**
IEEE	96.99%	93.72%	97.45%	99.09%	97.45%	99.32%	98.57%	**97.51%**
ACM	93.34%	83.77%	84.45%	95.12%	94.26%	98.11%	97.47%	**92.36%**
ISR	97.48%	84.71%	88.41%	97.37%	96.73%	98.54%	98.33%	**94.51%**
MISQ	97.30%	93.46%	95.84%	98.90%	97.03%	99.61%	98.86%	**97.29%**
JMIS	89.48%	77.96%	86.32%	98.42%	94.89%	98.89%	98.85%	**92.11%**
Avg.	**95.60%**	**86.53%**	**90.31%**	**97.93%**	**96.22%**	**98.99%**	**98.40%**	**94.85%**

3.3.2 Comparison with CRF and HMM

In the second experiment, we compared BibPro with the CRF and HMM systems, using EVAL2 as the performance measurement for the D2 dataset. The results are shown in Table 4. We compare BibPro with these systems because it is designed to extract the most common fields for citation strings; therefore, we could only measure the accuracy of the author, title, journal, volume, issue, page, month and year fields in a citation string. Moreover, we use the month and year fields to represent the date field and ignore the issue field because it was not included in the D2 dataset. The results show that BibPro is more accurate than HMM, but less accurate than CRF. However, since the D2 dataset only contains 500 records, it is not large enough to evaluate the performance of a real-world system. Furthermore, the D2 dataset comprises multiple styles that are difficult to differentiate. However, since BibPro automatically builds a feature list for each known template during the token matching step, it does not work very well with citation strings that have ambiguous tokens, such as those in D2. Thus, the results suggest that the size of the dataset and the variety of the citation strings in the dataset may have a strong impact on the system's performance.

Table 4 Extraction results of HMM, CRF and BibPro on D2 using EVAL2.

	HMM		**CRF**		**BibPro**	
	acc.	F1	acc.	F1	acc.	F1
Author	96.80%	92.70%	99.90%	99.40%	97.17%	93.98%
Booktitle	94.40%	0.85%	97.70%	93.70%		
Date	99.70%	96.90%	99.80%	98.90%	99.92%	98.96%
Editor	98.80%	70.80%	99.50%	87.70%		
Institution	98.50%	72.30%	99.70%	94.00%		
Journal	96.60%	67.70%	99.10%	91.30%	93.58%	83.27%
Location	99.10%	81.80%	99.30%	87.20%		
Note	99.20%	50.90%	99.70%	80.80%		
Pages	98.10%	72.90%	99.90%	98.60%	99.21%	92.09%
Publisher	99.40%	79.20%	99.40%	76.10%		
Tech	98.80%	74.90%	99.40%	86.70%		
Title	92.20%	87.20%	98.90%	98.30%	94.17%	90.13%
Volume	98.60%	75.80%	99.90%	97.80%	99.21%	84.62%

3.3.3 Comparison with ParaCite

In this experiment, we compared BibPro with ParaCite [16] using EVAL3 as the performance measurement on the D3 dataset. The results are detailed in Table 5. Since the source code for ParaCite is available on the Internet, we can use the D3 dataset, which was compiled by our automatic programs to compare ParaCite's performance with that of BibPro. Because ParaCite does not automatically build templates, we use ParaCite's default template database to test the D3 dataset, which contains about 4000 records. Moreover, because ParaCite can only extract one author name per citation string, its accuracy in the author field is much lower than that of BibPro. From Table 5, we observe that, in terms of accuracy, BibPro outperforms the ParaCite system by more than 20% in all fields, except the title field, and by as much as 90% in the page field. BibPro achieves a better performance than ParaCite because the D3 dataset consists of real data, which is more complex than regular datasets. However, comparing the accuracy level of the different fields in BibPro, it is interesting to note that the average accuracy for the title and journal fields is consistently lower than it is for other fields. This is probably due to the frequent variability (the variability in punctuation e.g., "-", ".", and "?") in the title and journal fields.

Table 5 Extraction results of ParaCite and BibPro on D3 using EVAL3.

	Author	Title	Journal	Volume	Page	Issue	Month	Year
Bib Pro	93.11%	73.31%	54.23%	82.79%	95.08%	84.63%	88.99%	96.47%
Para Cite	24.02%	72.77%	29.65%		4.67%	24.57%		77.02%

3.4 Analysis

We now consider several important factors that can influence the performance of BibPro. The factors are the benchmark, score matrix, knowledge database, and template database. Our experiment results suggest that the origin of target datasets affects the performance substantially. In addition, the more regular the dataset is, the higher the level of accuracy will be. To determine what other factors reduced the accuracy level, we checked the datasets manually. Our findings are listed below.

- [**Template Creating Error**]: Since the process of transforming a citation string into its STYLE FORM depends on token matching, problems may arise when tokens with ambiguous meanings are encountered. For example, the inclusion of numbers or people's names in the title field may affect the accuracy of the author and year fields. We applied a template filter to alleviate this problem, but it still affects BibPro's performance to some degree.
- [**Extraction Alignment Error**]: Even though BLAST can find templates with a high degree of similarity to the target citation strings during the online parsing phase, errors in alignment continue to occur during the extraction process. There may be several different alignments with templates that have the same similarity score. Hence, in the trace-back stage of global alignment, there may be many paths to trace back, but it is very difficult to choose the correct path automatically.
- [**Database Completeness Problem**]: Because we use the template database as training data, the comprehensiveness of the template database and knowledge database has a strong influence on the performance of BibPro.

Since BibPro's performance depends to a large extent on the template database, our primary interest is to determine how we can automatically generate each template's feature index as precisely and efficiently as possible. However, as we use a token matching technique to recognize templates, it is difficult to create the correct feature index when citation styles are very complex. This is an interesting problem that we will consider in our future work.

We also applied two sequence alignment techniques in BibPro: BLAST and the Needleman-Wunsch algorithm. Both techniques are based on dynamic programming, so they need score matrices to evaluate the alignment results. The score matrix can be adjusted as necessary to meet different requirements. In this paper, we adjusted the score matrix to fit our experiments, as detailed in Appendix 1.

4. CONCLUSION AND FUTURE WORK

Parsing citations is a challenging problem due to the diverse nature of citation styles. In this paper, we have proposed a template-based citation parsing system called "BibPro." It not only adds new citation templates easily, but also searches for the most similar templates so that it can extract metadata from citation strings rapidly. In BibPro, we use the order of punctuation marks in a citation string as features of the string's citation style. We then transform citation strings into protein sequences and apply two sequence alignment techniques, BLAST and the Needleman-Wunsch algorithm, to find the most similar template for the online parsing process. To evaluate the performance of BibPro, we compare it with some other systems by implementing experiments with various evaluation measures and datasets. The experiment results show that BibPro performs well when good quality template databases are used for training.

There are still several challenges to address when implementing BibPro in real world applications. One challenge is that it is difficult to get accurate, large-scale training datasets to cover all kinds of citation styles. Moreover, the training data we can collect from the Internet may contain a variety of errors, such as missing values, spelling errors, inconsistent abbreviations, and extraneous tokens [9]. Another challenge is that different publication types use a variety of information fields. It is difficult to extract all the information fields from each of the publication types. Therefore, in this paper, we concentrate on the most common information (fields) for all publication types. In the future, we will try to determine how we can generate the system templates more precisely and efficiently, and thus make BibPro more practical for real-world applications.

5. REFERENCES

[1] Giles, C. L., Bollacker, K. D., and Lawrence, S. CiteSeer: an automatic citation indexing system. *Digital Libraries 98* Pittsburgh PA USA, 1998.

[2] Bollacker, K. D., Lawrence, S., and Giles, C. L. CiteSeer: an autonous Web agent for automatic retrieval and identification of interesting publications. *In Proceedings of the Second international Conference on Autonomous Agents*, 1998.

[3] Lawrence, S., Giles, C. L., and Bollacker, K. D. Autonomous citation matching. *In Proceedings of the Third Annual Conference on Autonomous Agents*, 1999.

[4] Lawrence, S., Giles, C. L., and Bollacker, K. D. Digital Libraries and Autonomous Citation Indexing. *IEEE Computer, 32*, 1999, 67-71.

[5] F. Peng, A. McCallum. Accurate information extraction from research papers using conditional random fields. *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics* (HLT-NAACL), 2004, 329-336.

[6] Hui Han, Giles, C.L., Manavoglu, E., Hongyuan Zha, Zhenyue Zhang, Fox, E.A. Automatic document metadata extraction using support vector machines. *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital libraries*, 2003, 37-48.

[7] K. Seymore, A. McCallum, R. Rosenfeld. Learning hiddenMarkov model structure for information extraction. *AAAI-99Workshop on Machine Learning for Information Extraction,* 1999, 37-42.

[8] Takasu, A. Bibliographic attribute extraction from erroneous references based on a statistical model. *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital libraries*, 2003, 49-60.

[9] Agichtein, E. and Ganti, V. Mining reference tables for automatic text segmentation. *In Proceedings of the Tenth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining* (Seattle, WA, USA, August 22 - 25, 2004). KDD '04. ACM Press, New York, NY, 2004, 20-29.

[10] I-Ane Huang, Jan-Ming Ho, Hung-Yu Kao, and Shian-Hua Lin. Extracting citation metadata from online publication lists using BLAST. *In PAKDD*, 2004, 539-548.

[11] Min-Yuh Day, Tzong-Han Tsai, Cheng-Lung Sung, Cheng-Wei Lee, Shih-Hung Wu, Chorng-Shyong Ong, Wen-Lian Hsu. A Knowledge-based Approach to Citation Extraction. *in Proceedings of the IEEE International Conference on Information Reuse and Integration (IEEE IRI 2005)*, Las Vegas, Nevada, USA, 2005, 50-55.

[12] Min-Yuh Day et al. Reference metadata extraction using a hierarchical knowledge representation framework. *Decision Support Systems,* 2006.

[13] S. F. Altschul, W. Gish, W. Miller, E. Myers and D. Lipman. A basic local alignment search tool. J. Mol. Biol., 215, 1990, 403-410.

[14] http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/similarity.html

[15] Needleman, S. B. and Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol., 48, 1970, 443-453.

[16] http://paracite.eprints.org/

[17] http://www.cs.umass.edu/~mccallum/code-data.html

6. APPENDIX 1: DEFAULT PARAMETER

BLAST and the Needleman-Wunsch algorithm both use score tables to evaluate their alignment results. Figure 16 shows the score table used for BLAST, while Figure 17 shows the score table used for the Needleman-Wunsch algorithm.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z	X	*
A	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
R	-1	3	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
N	-1	-1	8	-1	-1	-1	-1	1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
D	-1	-1	-1	3	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
C	-1	-1	-1	-1	6	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
Q	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
E	-1	-1	-1	-1	-1	-1	7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
G	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
H	-1	-1	1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
I	-1	-1	-1	-1	-1	-1	-1	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
L	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-4
K	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
M	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
F	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
P	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-4
S	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-4
T	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	1	-1	-1	-4
W	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-4
Y	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-4
V	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-4
B	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	3	-1	-1	-4
Z	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-4
X	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-4
*	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	1

Figure 16. The score table used in BLAST.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z	X	*
A	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	4	-1	3	-4
R	-1	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
N	-1	-1	6	-1	-1	-1	-1	9	-1	-1	-1	-1	9	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-4
D	-1	-1	-1	3	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
C	-1	-1	-1	-1	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
Q	-1	-1	-1	-1	-1	2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
E	-1	-1	-1	-1	-1	-1	7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
G	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
H	-1	-1	9	-1	-1	-1	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
I	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
L	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	3	-1	3	-4
K	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
M	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
F	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-4
P	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-4
S	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-4
T	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	5	-1	-1	-1	6	-1	3	-4
W	-1	-1	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	8	-1	-1	-1	-1	-1	-4
Y	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-1	-4
V	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-1	-1	-4
B	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	-1	6	-1	-1	-1	1	-1	-1	-4
Z	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	-1	-4
X	3	-1	-1	-1	-1	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	-1	3	-1	-1	-1	-1	-1	1	-4
*																								

Figure 17. The score table used in Needleman-Wunsch algorithm.