NAIST-IS-DT9761021

Doctor's Thesis

Studies on Multilingual Information Processing on the Internet

Akira Maeda

September 18, 2000

Department of Information Systems
Graduate School of Information Science
Nara Institute of Science and Technology

Doctor's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
DOCTOR of ENGINEERING

Akira Maeda

Thesis committee: Shunsuke Uemura, Professor
Yuji Matsumoto, Professor
Minoru Ito, Professor
Masatoshi Yoshikawa, Associate Professor

Studies on Multilingual Information Processing on the Internet *

Akira Maeda

Abstract

With the increasing popularity of the Internet in various part of the world, the languages used for Web documents are expanded from English to various languages. However, there are many unsolved problems in order to realize an information system which can handle such multilingual documents in a unified manner.

From the user's point of view, three most fundamental text processing functions for the general use of the World Wide Web are display, input, and retrieval of the text. However, for languages such as Japanese, Chinese, and Korean, character fonts and input methods that are necessary for displaying and inputting texts, are not always installed on the client side.

From the system's point of view, one of the most troublesome problems is that, many Web documents do not have meta information of the character coding system and the language used for the document itself, although character coding systems used for Web documents vary according to the language. It may result in troubles such as incorrect display on Web browsers, and inaccurate indexing on Web search engines. Also, other text processing applications such as categorization, summarization, and machine translation are dependent on identifying the language of the text to be processed.

Moreover, there might be some cases where the user wants to retrieve documents in unfamiliar languages, especially for cases where information written in languages other than the user's native language is rich. The needs for retrieving such information must not be small. Consequently, research on cross-language information retrieval

*Doctor's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT9761021, September 18, 2000.

(CLIR), which is a technique to retrieve documents written in one language using a query written in another language, is being paid much attention. However, it is difficult to achieve adequate retrieval effectiveness for Web documents in diverse languages and domains.

The goal of this thesis is to provide some solutions to these problems. Specifically, we focus on techniques such as:

1. display and input functions for multilingual text which does not depend on installed fonts and input methods on the client side,
2. an algorithm for the automatic identification of the languages and the character coding systems of Web documents based on a combination of both statistics and heuristics,
3. a cross-language information retrieval technique, which is suitable for Web documents in diverse domains, based on the word co-occurrence information obtained from a Web search engine.

By integrating these three techniques, we realized a system which supports access to documents written in languages other than the user's native language. This system provides some solutions to the problems in multilingual information processing that are specific to the Internet.

Keywords:

internationalization, multilingual document, text processing, character coding system, cross-language information retrieval, WWW

Acknowledgements

I am grateful to my advisor, Professor Shunsuke Uemura for his guidance, support, and encouragement throughout the doctoral course.

I would like to thank Associate Professor Masatoshi Yoshikawa for his guidance and numerous suggestions with this research.

I would also like to thank other members of my thesis committee, Professor Yuji Matsumoto and Professor Minoru Ito, for their valuable comments on this thesis.

I am grateful to Professor Koichi Tabata, Professor Shigeo Sugimoto, and Associate Professor Tetsuo Sakaguchi at University of Library and Information Science, and Assistant Professor Takehisa Fujita at Kyoritsu Women's University, for their advice and guidance through the master's course, particularly on the development of MHTML.

I would also like to thank Dr. Toshiyuki Amagasa, Dr. Kenji Hatano, and all members of Uemura Laboratory at Nara Institute of Science and Technology for discussions and help.

Finally, I would like to express my sincere gratitude to my parents for their support and encouragement throughout my student years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The increasing volume of information available globally through the Internet places high demands on information systems that can handle multilingual documents in a unified manner. Also, the languages used for Web documents are expanded from English to various languages. However, there are many unsolved problems in order to realize an information system which can handle such multilingual documents in a unified manner.

From the user's point of view, three most fundamental text processing functions for the general use of the World Wide Web are display, input, and retrieval of the text. However, for languages such as Japanese, Chinese, and Korean, character fonts and input methods that are necessary for displaying and inputting texts, are not always installed on the client side.

From the system's point of view, one of the most troublesome problems is that, many Web documents do not have meta information of the character coding system and the language used for the document itself, although character coding systems used for Web documents vary according to the language. It may result in troubles such as incorrect display on Web browsers, and inaccurate indexing on Web search engines. Also, other text processing applications such as categorization, summarization, and machine translation are dependent on knowing the language of the text to be processed.

Moreover, there might be some cases where the user wants to retrieve documents in unfamiliar languages, especially for cases where information written in a language

other than the user's native language is rich. The needs for retrieving such information must not be small. Consequently, research on cross-language information retrieval (CLIR), which is a technique to retrieve documents written in one language using a query written in another language, are being paid much attention. However, it is difficult to achieve adequate retrieval effectiveness for Web documents in diverse languages and domains.

In this thesis, we propose three individual techniques to solve these problems. These techniques are distinct from each other, but are used as components of a complete system that supports end-user access to multilingual information on the Web.

1.2 Focus of this Research

The goal of this thesis is to provide some solutions to the problems described above. Specifically, we focus on the following three techniques:

1. display and input functions for multilingual text which does not depend on installed fonts and input methods on the client side, which will be described in Chapter 3,

2. an algorithm for the automatic identification of the languages and the character coding systems of Web documents based on a combination of both statistics and heuristics, which will be described in Chapter 4,

3. a cross-language information retrieval technique, which is suitable for Web documents in diverse domains, based on the word co-occurrence information obtained from a Web search engine, which will be described in Chapter 5.

By integrating these three techniques, we realized a system which supports access to documents written in languages other than the user's native language. This system provides some solutions to the problems in multilingual information processing that are specific to the Internet.

1.3 Organization of the Thesis

This thesis is organized as follows; in Chapter 2, some basic information and current issues related to multilingual information processing on the Internet are introduced. In Chapter 3, display and input functions of multilingual text which does not depend on installed fonts and input methods on the client side is proposed. In Chapter 4, an algorithm for automatic identification of the languages and the character coding systems of Web documents is proposed. In Chapter 5, a cross-language information retrieval technique which is suitable for Web documents in diverse domains is proposed. In Chapter 6, a system called *Multilingual Knowledge Discovery System*, which integrates three techniques proposed in previous chapters, is introduced. Finally, Chapter 7 discusses the contributions of this study, and concludes the thesis.

Chapter 2

Introduction to Multilingual Information Processing on the Internet

2.1 Introduction

In this chapter, we introduce some basic information and current issues that are related to multilingual information processing on the Internet, with particular emphasis on the Web.

2.2 Layers of Multilingual Information Processing on the Web

Figure 2.1 shows the layers of multilingual information processing on the Web.

The 1st layer is *character coding system*, which defines the characters sets and their encodings to be used in the upper layers. It can be further divided into two components; *character encoding scheme* and *character set*. These components will be described in detail in the next section.

The 2nd layer is *communication protocol*, which defines how to transmit documents through a communication network, typically the Internet. HTTP (HyperText Transfer Protocol)[27] is an Internet protocol for communication between user agents (e.g. Web browsers) and Web servers. It has some features related to multilingual information processing, such as indicating the character encoding scheme of a page

and indicating the language(s) of the specific bounds of a text, etc. MIME (Multipurpose Internet Mail Extensions)[28] is primarily defined for electronic mail messages. However, some features, especially the Content-Type header, are also used in HTTP. The charset attribute of the Content-Type header, which will be described later in this chapter, is one of the most important features for multilingual information processing on the Internet and the Web.

The 3rd layer is *text format*, which defines the structure of a document. HTML (HyperText Markup Language)[91] is a fundamental text format for the Web. As described later in this chapter, it involves many features that are related to multilingual information processing.

The 4th layer is *user interface*, which is typically a Web browser. Although a Web browser is an application in the sense of operating systems, it provides a user interface for Web applications that run on a browser. It also involves many features related to multilingual information processing, such as display and input.

The 5th layer on the top is *Web application*, which runs on a Web browser. Typical Web applications include search engines, digital libraries, electronic commerce sites, etc. Since the Web itself is multilingual, every Web applications that manage Web documents, such Web search engines, must handle multilingual documents to some extent.

2.3 Character Coding Systems

2.3.1 Definition of Terms

In this section, we define terms related to character code standards, which are the most fundamental part in text processing on computers. Unfortunately, the terminology in this field are somewhat confused and sometimes mistakenly used, even in standards such as MIME (Multipurpose Internet Mail Extensions) *charset* parameter[28, 17]. Moreover, the definitions of those terms are not fully standardized, and therefore slightly different terms are used among standards to represent the same concept. The terms used in this thesis are mostly come from the definitions of ISO (International Organization for Standardization), but some terms (e.g. character encoding scheme) does not have the corresponding term in ISO standards. In such a case, we use terms that

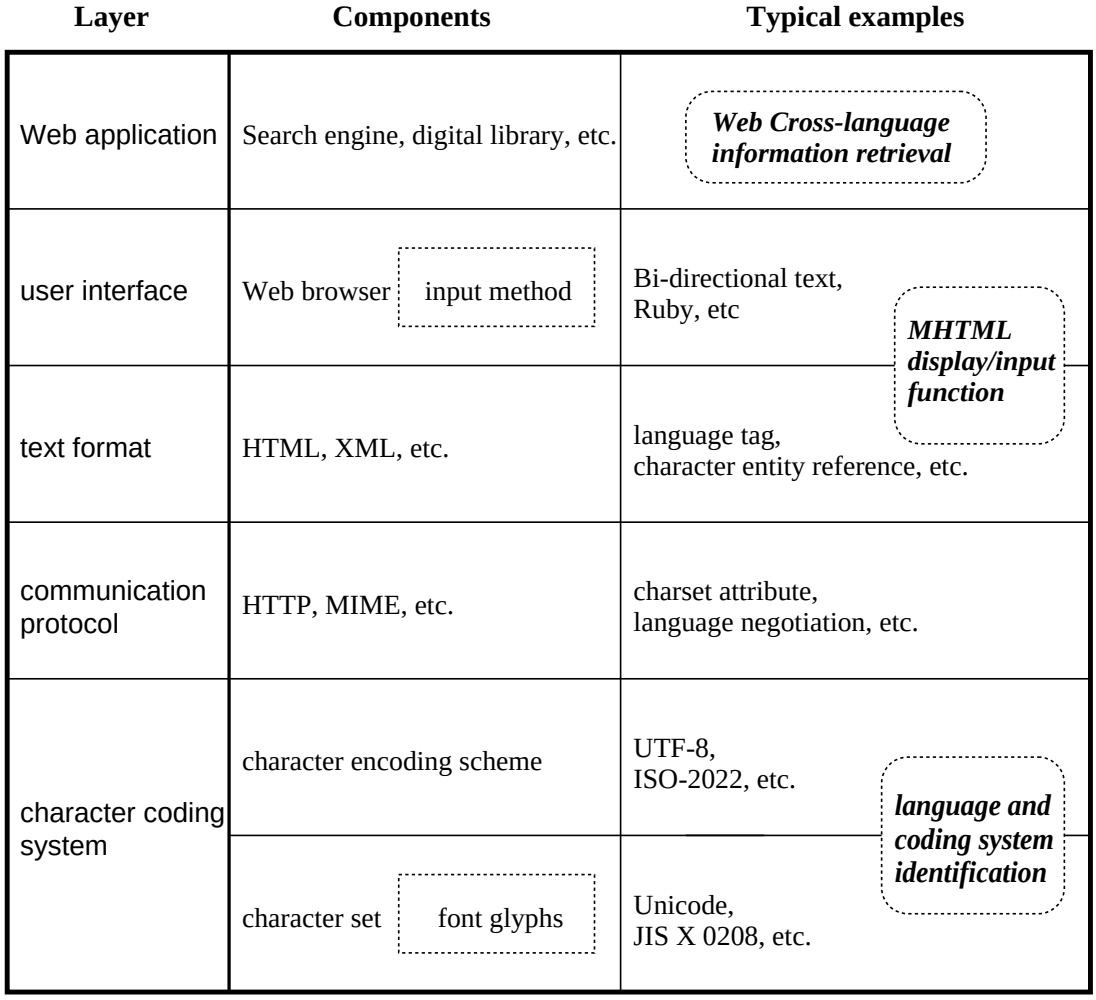


Figure 2.1. Layers of multilingual information processing on the Web.

are defined in RFC 2130[102] and RFC 2277[4]. RFC (Request For Comments) documents are published by IETF (Internet Engineering Task Force), which define various standards and protocols that are used on the Internet.

The following definitions of terms are used in the thesis:

Character set is a collection of elements (characters) used to represent textual information.

Graphic character is a character typically associated with a visible display representation, and is not primarily associated with a control or formatting function.

Coded character set is a character set in which each character is assigned a numeric code value.

Character encoding scheme is a mapping from one or more character set definitions to the actual bits used to represent the data. Character encoding scheme is also simply called *encoding scheme* or *encoding*.

Character coding system that we deal with in this chapter and Chapter 4, is a combination of one or more coded character sets and an encoding scheme for those. In this thesis, we call it *coding system* for short.

Character encoding schemes can be classified into the following three categories:

Modal encoding is a character encoding scheme that requires escape sequences or other special characters for switching between character sets or different versions of the same character set.

Non-modal encoding, on the other hand, is a character encoding scheme that makes use of the numeric values of bytes (typically the eighth bit) to switch between character sets.

Fixed-length encoding is a character encoding scheme that uses the same number of bytes to represent all the characters.

2.3.2 European Languages

The most fundamental character set for today's computers is ASCII (American national Standard Code for Information Interchange)[5], or ISO 646 [43] IRV (International Reference Version). It defines 94 code points for Latin alphabets and symbols, and 34 code points for control codes. It is used as the basis for almost all of the character coding system standards in the world, including Unicode, which will be described later.

ASCII is also referred to as a character encoding scheme, in which the code points are simply used as the codes. The code range of ASCII is 0-127, thus it is called a 7-bit coding system.

For European languages that use accented letters, such as French and German, etc., a character set called ISO 8859-1[49] is defined. It is an extension of ASCII, is a sense that it uses the same code points in the code range of 0-127. It uses the code range of 128-255 for accented letters and some additional symbols. Thus, it is an 8-bit coding system.

2.3.3 Asian Languages

Japanese

For Japanese, usually ASCII (for one byte alphabets and symbols) or JIS X 0201[54] (for one byte alphabets, symbols, and katakana's) and JIS X 0208[55] (for two bytes Kanji's), and occasionally JIS X 0212[53] (Japanese supplementary character set), are used for coded character sets.

The most popular encoding scheme for Japanese is Shift_JIS[56][1] originally developed by Microsoft. It is a variable-length non-modal 8-bit encoding. Although it has many drawbacks such as some overlaps in code ranges between ASCII and JIS character sets, which may require a special treatment in text processing, it is widely used in many popular operating systems (e.g. Microsoft Windows, Apple MacOS, and some Unix implementations) and applications.

EUC-JP[87] is a variation of EUC (Extended Unix Code) that supports Japanese. EUC conforms to ISO-2022[46] standard code extension technique. It is a variable-length non-modal 8-bit encoding. It is mainly used as the internal encoding scheme for Unix environments configured to support Japanese. To be exact, the specific encoding scheme that is widely used in Unix environments is called *packed format* of EUC, and another format called *complete two-byte format*, which is fixed-length, is rarely used.

ISO-2022-JP is defined in RFC 1468[80]. It is a variable-length and modal encoding, in which escape sequences or other special characters are used to switch between different character sets. It is a 7-bit encoding that is suitable for electronic mail and net-

[1] In this thesis, we use the preferred MIME names that are registered in IANA (Internet Assigned Numbers Authority), for the name of a coding system.

work news systems. Although its primary purpose is for electronic mail and network news, it is also used for some Web documents written in Japanese.

Chinese

GB2312 is an 8-bit encoding scheme for simplified Chinese that handles ASCII and GB 2312-80[90] character sets. Despite its name, it is a variation of EUC, and is a variable-length non-modal encoding.

Big5 is an 8-bit encoding scheme for traditional Chinese that handles ASCII and Big Five[42] character sets. It is a variable-length non-modal encoding. Although it is not a national standard, it has become a de facto standard in Taiwan.

ISO-2022-CN is defined in RFC 1922[36]. It specifies how Chinese text is to be encoded for electronic mail messages. It is a 7-bit variable-length modal encoding, which handles both simplified and traditional Chinese.

Korean

EUC-KR[65] is a variation of EUC that supports Korean. It is an 8-bit variable-length non-modal encoding.

ISO-2022-KR is defined in RFC 1557[15]. It specifies how Korean text is to be encoded for electronic mail messages. It is a 7-bit variable-length modal encoding.

Arabic

Two character sets are widely used for Arabic. One is Windows-1256 and the other is ISO-8859-6[48]. Both contain ASCII characters and Arabic alphabets in a single-byte 8-bit range. Like European character sets, these are also referred to as character encoding schemes, in which the code points are simply used as the codes. Each character set contains slightly different character repertoire, thus not compatible.

2.3.4 Multilingual

Unicode[100] is a multilingual character coding system that includes most of the major scripts of the world. It is a subset of ISO 10646-1[45] and is equivalent to Basic Multilingual Plane (BMP) of ISO 10646-1. Unicode specifies both a coded character

set called UCS (Universal Character Set) and encoding schemes called UTF (UCS Transformation Formats). Several encoding schemes such as UTF-7[33], UTF-8[105], and UTF-16[47] are used according to the purpose.

ISO-2022-JP-2[86] is defined in RFC 1554, and is a multilingual extension of ISO-2022-JP that handles simplified Chinese (GB2312), Korean (KSC5601), Japanese supplementary character set (JIS X 0212), some European languages (ISO 8859-1 and ISO 8859-7) in addition to the coding systems supported in ISO-2022-JP. One of the applications that support this encoding is a multilingual text editor called MULE[82].

Although some alternative multilingual character coding systems have been proposed such as TRON code[95], Multicode[79], and EPICS[88], these are not widely used.

2.4 Operating Systems

2.4.1 L10N vs. I18N

Localization (often abbreviated as l10n) refers to the process of adapting software to one specific language or culture. *Internationalization* (often abbreviated as i18n) refers to the process of preparing software so that it can be used by more than one language or culture.

The locale model is one method to internationalize operating systems, and applications that run on it, and has been implemented on Unix[44]. It predefines attributes that are language- (or locale-) specific, such as the date format, time format, currency format, numeric formatting, collation ordering, diagnostic messages, etc. Only one locale can be specified for an application. Therefore, the user must explicitly switch the locale in order to use languages that are not defined in the current locale.

The multilingual model is an alternative to the locale model, in which the user does not have to switch between locales to use multiple languages. However, there are some cases when it is impossible to correctly process (e.g. rendering characters, sorting strings, etc.) without knowing the target locale or language.

Almost all of Unix variants are based on the locale model. However, implementation of the model is not strictly standardized, so the usage and the functionalities are slightly different between systems. For example, there are many variations for speci-

fying the encoding scheme to be used in the locale name (e.g. to specify the territory as Japan, the language as Japanese, and the encoding scheme as EUC-JP, the following different names are used: “ja”, “japanese”, “ja_JP.EUC”, “ja_JP.ujis”, “ja_JP.eucJP”, etc.).

2.4.2 Internationalized Operating Systems

Kataoka et al.[59, 58] proposes a multilingual input/output system that has been implemented on X window system version 11 release 5 (X11R5). This system is built on top of an operating system, i.e. Unix, and is independent from the locale models furnished by the operating system, such as the POSIX locale model[44]. This system is one of the typical realizations of the multilingual model introduced in the previous section.

TRON is an operating system that is based on the multilingual model[95]. This system stratifies multilingual handling functions into four layers, i.e. *language*, *group*, *script*, and *font* layers, in order to make application programs language-independent. The TRON code system, which is used as the internal coding system of TRON, has a capability to assign language-specifier codes to the specific portion of a text. It enables the separation of language-dependent and language-independent algorithms for text processing.

2.5 Multilinguality on the Web

2.5.1 Standards and Protocols

Although early protocols and standards related to the Web, such as HTTP and HTML, assumed only European languages to be used for documents, they have already been internationalized in some degree.

Initially, HTML was restricted to the ISO-8859-1 character set which is appropriate only for Western European languages. HTML4.0[91] now specifies ISO 10646 UCS (Universal Character Set) as the document character set. It allows the use of various coding systems which covers most existing character coding systems used on the Internet. Besides, it defines *lang* attribute for specifying the language of the specific

bounds of a text, and some other issues on i18n are taken into account (e.g. bidirectional text handling, character references etc.). On the other hand, at the coding system level, *language tag* is defined in Unicode 3.0[100] for the same purpose as HTML *lang* attribute.

Similar to HTML, XML (eXtensible Markup Language)[12] specifies ISO 10646 UCS as the document character set. It allows various coding systems to be used. The coding system ("encoding" in XML context) used in an XML document can be indicated within the XML declaration, which should be placed at the first line of any XML document, by using *encoding* attribute. Also, XML defines *xml:lang* attribute for specifying the language of the specific bounds of a text according to the language tags defined in RFC 1766[3].

HTTP1.1[27] defines a mechanism to indicate the coding system of a document sent from the server to the client by using the *charset* parameter of the *Content-Type* header. It can be used for any text-type content, including HTML, XML, and plain text. Also, it defines the language negotiation algorithm in which a user can specify the preferred language(s) from the multiple language versions of a document.

Unfortunately, these i18n features are not fully implemented on current WWW servers and browsers. The most serious problem is the disuse of *charset* parameter in current WWW documents, which causes incorrect display on browsers or incomplete indexing on search engines.

Recently, popular Web browsers are becoming to support more languages and coding systems for the display function. Netscape Navigator version 4 supports more than 22 coding systems. Internet Explorer version 5 supports more than 34 coding systems, including Arabic and Hebrew, which are bidirectional in writing direction. Both browsers support Unicode (UTF-8). However, the user has to install the appropriate fonts to display a document written in the language for which fonts are not installed.

2.5.2 Cross-Language Information Retrieval

Recently, as an approach to easily access multilingual information, researches on cross-language information retrieval (CLIR) are becoming to be paid much attention[63]. CLIR is a technique to retrieve documents written in one language using a query written in another language. It is effective in cases such as: 1) document collection itself is multilingual, 2) the user can read the language of document collection, but hard to

formulate a query. Various approaches to CLIR will be further discussed in Chapter 5.

2.5.3 Text Input/Output

Text input is a crucial issue especially for languages like Chinese, Japanese, and Korean (CJK), because CJK have a large number of characters and thus a special input method is needed in order to input characters of these languages from a keyboard. Text output is also a crucial issue for most languages other than European languages, because the fonts for foreign languages are usually not installed in computers.

Yong et al.[106] proposes a technique to input characters on a Web browser without using the input method installed on the client. Their approach is to download the input method and the fonts for a particular language as a Java applet. Consequently, it might take long time to download the whole applet for languages that use many characters or that require a complex input method, such as Japanese, Chinese, and Korean.

2.6 Summary

In this chapter, we briefly introduced some basic information and current issues related to multilingual information processing on the Internet. The current situation is much better than the early days of the Internet and the Web, but there are still many problems to be solved in order to handle multilingual documents in a unified manner, such as a lack of installed fonts and input methods for some languages, an identification of coding systems and languages, and a language-barrier to access documents written in foreign languages. In the following sections, some solutions to those problems will be proposed.

Chapter 3

MHTML: Display and Input Functions for Multilingual Web Documents

3.1 Introduction

In order to realize true international information sharing on the Internet, it is important to be able to read and retrieve documents from every part of the world, in the same manner. Recently, WWW browsers that support Unicode are becoming popular, and operating systems are becoming to support Unicode as the internal character code. Consequently, it is much easier today to handle multilingual documents than before.

However, for languages such as Japanese, Chinese, Korean, and some minor languages, character fonts and input methods that are necessary for displaying and inputting texts, are not always installed on the client side. In ordinary personal computer environments, usually only fonts for the native language in addition to English are installed by default. Therefore, in order to display documents in other languages, the user has to install additional fonts for that language. This process is rather a hard task for casual users of personal computers and the Internet. Moreover, even if a multilingual coding system such as Unicode and the browsers that support it become popular, it is not realistic to prepare fonts for all languages supported by Unicode in every client, especially for small clients with a limited storage, such as PDAs (Personal Digital Assistants) and mobile phones.

The simplest and the easiest solution to this problem is to realize a browser that does not require fonts on the client side. That is, display of a text will be possible

regardless of the installed fonts on the client, by transmitting character glyphs instead of character codes. In this approach, it is inevitable that the number of bytes to be transmitted will increase, but on the other hand, it can avoid the problem of installing and storage of fonts. Incidentally, it can be used to display characters that are not included in existing character sets. As practical methods to implement such an approach utilizing an existing Web browser, we can think of following methods:

1. To convert the source document into a set of page images (Figure 3.1),
2. To replace each character or string in the source document into an inline image (Figure 3.2),
3. To add fonts for characters that appeared in the source document, and use that fonts to display the text (Figure 3.3).

The method 1 is a low-cost and practical method for digitizing existing paper documents, and used in many existing digital library systems. Hyperlinks of HTML can be implemented by using clickable maps. The method 2 is already implemented as a conversion service of existing Web documents, i.e. CII (Character to Inline Image) library of *DeleGate*[96][1] proxy server and *Shodouka*[2].

This method has an advantage that it can utilize the HTML layout engine of Web browsers. The method 3 is the one we propose. It converts the source HTML document into a format called MHTML in which fonts for appeared characters are added, and it is displayed in a Java applet which runs on a Web browser.

We have developed a browsing system which enables viewing multilingual HTML documents using this technique[72, 93, 98, 68, 70, 69], and it is extended to implement the text input function[99]. In this chapter, we introduce this technique in detail, and show some applications, such as an OPAC[30], a collection of Japanese folk tales[19, 20], and an SGML-based text retrieval system[94].

[1] http://www.delegate.org/

[2] http://www.shodouka.com/

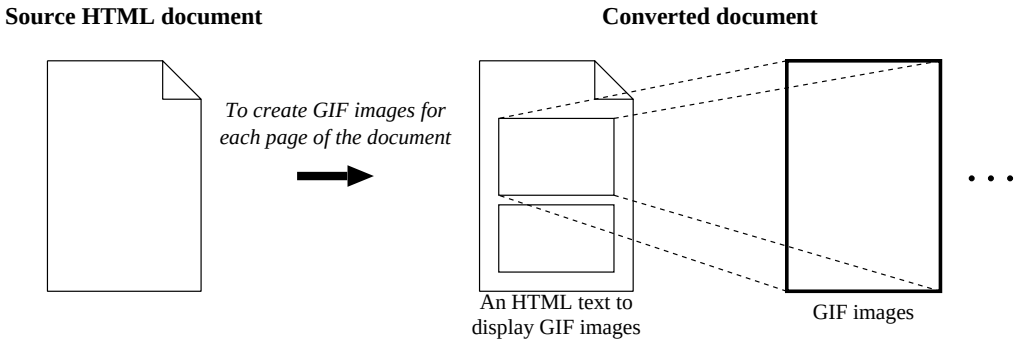


Figure 3.1. The method of converting the source document into a set of page images.

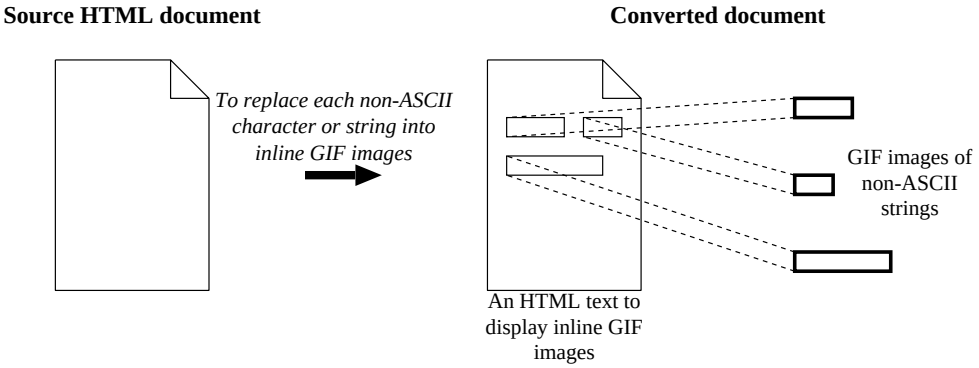


Figure 3.2. The method of replacing characters or strings into inline images.

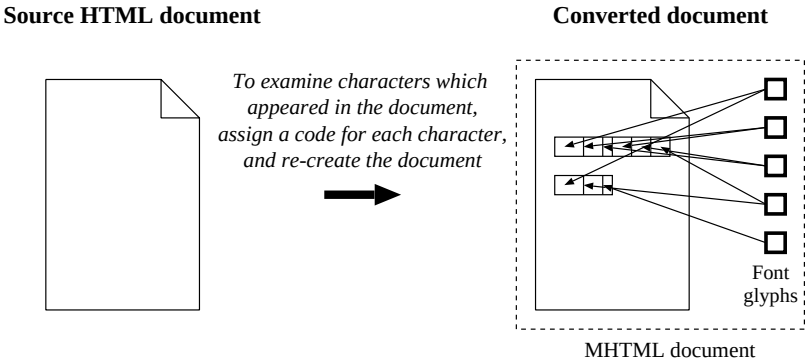


Figure 3.3. The method of adding fonts for characters appeared in the source document.

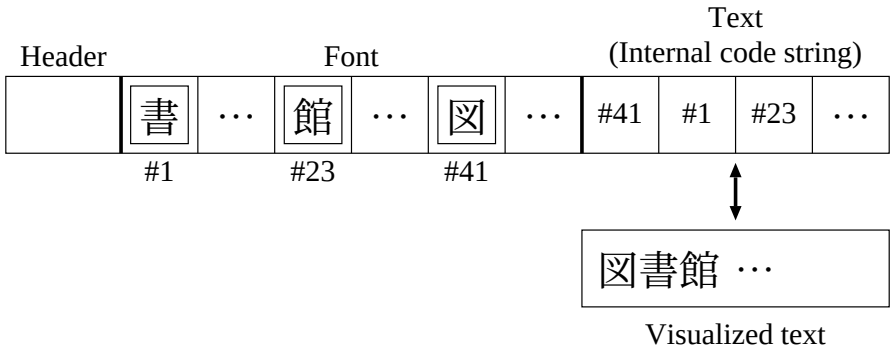


Figure 3.4. Outline of the MHTML document.

3.2 MHTML (Multilingual-HTML) Document Format

3.2.1 Overview of MHTML

Figure 3.4 shows the outline of the display mechanism of MHTML. An MHTML document contains a source HTML text and a minimum set of font glyphs. The character codes of the source text, except for HTML tags, are internalized to the document itself, so that the codes are effective only in the document.

Figure 3.5 shows the mechanism of displaying a document using MHTML. First, WWW browser on the client loads a Java applet called MHTML class from MHTML server, and the applet is invoked on the browser. Then, two parameters, the URL of the source HTML document (*URL*) and the coding system identifier which indicates the coding system used for the document (*coding system ID*), are sent to the MHTML server (1). The MHTML server fetches the source HTML document indicated by *URL* parameter (2), and converts it into MHTML using *coding system ID* parameter. Finally, the MHTML document is returned to the browser (3), and the document is displayed on the MHTML class applet.

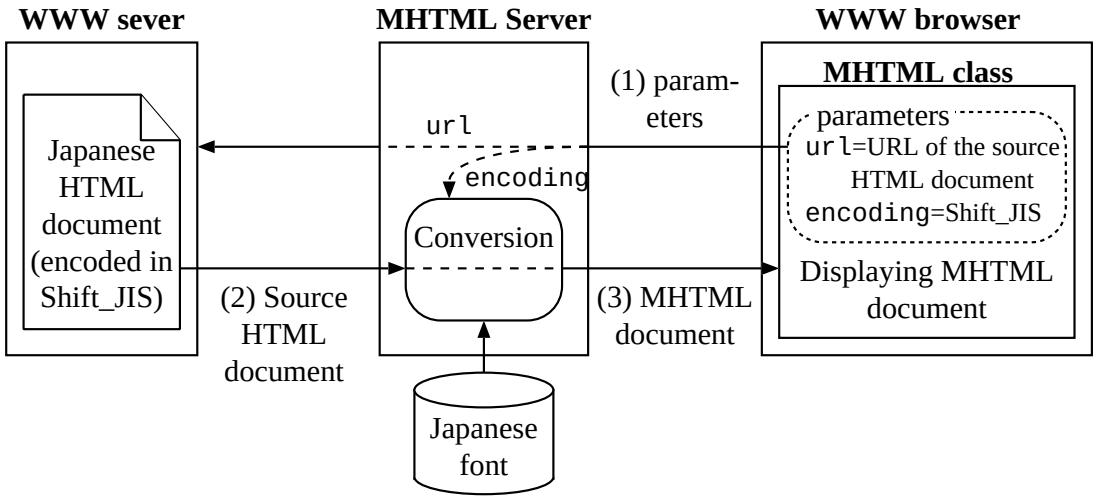


Figure 3.5. The mechanism of displaying a document using MHTML.

3.2.2 Comparing with Other Methods

Required Transmission Size

To clarify the advantage of our method in terms of size, we conducted a comparative experiment for required transmission size of our method and alternative methods described in Section 3.1. Specifically, we compared the ratios between the transmitted bytes and the source document size, for four methods: a set of page images, inline images of characters or strings (CII: Character(s) to Inline Image), and MHTML.

For the method of page images, the transmitted bytes were calculated by summing the sizes of GIF images of each page captured from a Web browser. For CII method, the transmitted bytes were calculated by summing all inline images converted by DeleGate CII library and the HTML document to display those inline images. In the case that the same character or string appeared more than once, it is counted only once, because a browser usually caches it at the first fetch. For CII, two methods were examined, i.e. one character per inline image (CII/C), and at most 30 continuous characters per inline image (CII/S).

Figure 3.6 shows the result of the comparative experiment for 30 Japanese academic articles published in HTML. In the figure, X-axis indicates the source document size, and Y-axis indicates the ratio between the transmission size of the converted doc-

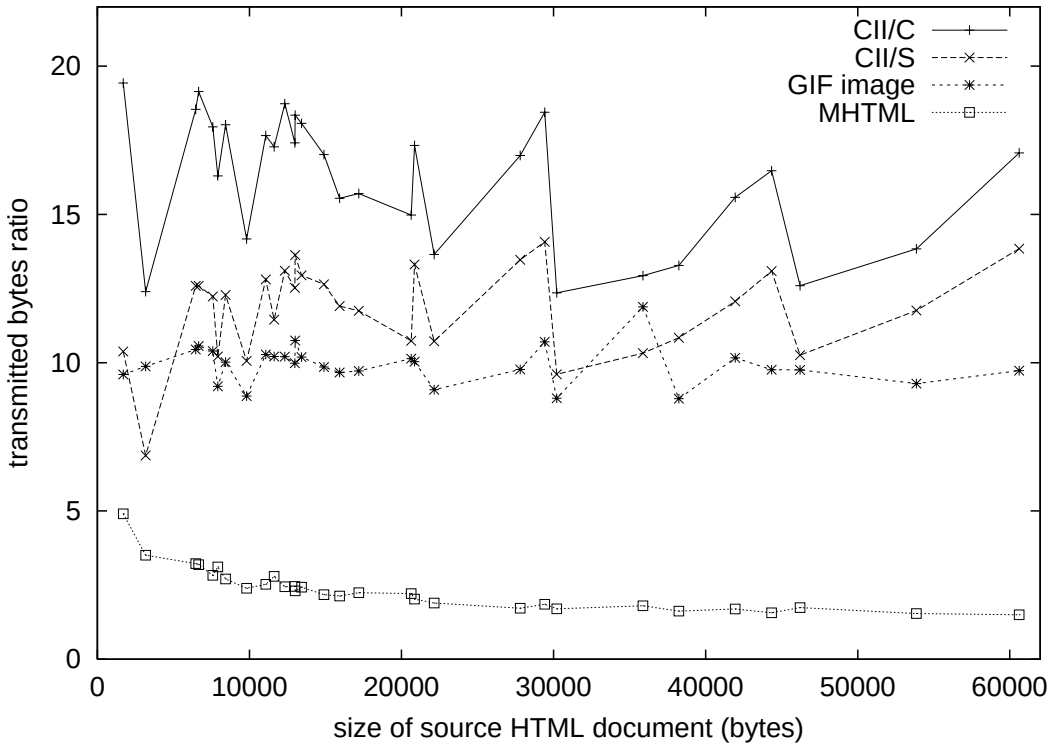


Figure 3.6. Comparison of the transmitted bytes ratios for each method.

ument and its source. From the figure, we can observe that GIF image and CII methods require roughly 10-20 times larger transmission size than its source, whereas MHTML method requires only twice large for documents larger than 10 kilobytes.

Overhead of Network Connections

Another crucial factor that affects the time to complete displaying a document on the client is the number of HTTP connections to transmit all required files. In CII method, it requires TCP (Transmission Control Protocol) connections for inline images of every characters or strings in a document. It causes an extreme overhead for slower network connections such as a dial-up connection to the Internet, thus it may take very long time to complete rendering the whole document.

On the other hand, MHTML method requires only one HTTP connection for a document, thus it is even more efficient than other methods in terms of the network

connection.

Comparing with Web Fonts

Web Fonts[11] is an alternative method to add fonts to an HTML document. This method has a functionality to specify the location of a font in a HTML document, and supported browser dynamically downloads the specified font when displaying the document. Although its primary objective is to provide a rich presentation of HTML documents using various font shapes, it can be used as an alternative method to display multilingual documents without using fonts on the client.

However, Web Fonts requires a supported Web browser on the client, and the author of a document has to add the font information to the document beforehand. In addition, Web Fonts does not have a functionality to transmit only required fonts for a document, thus the whole font for a character set has to be transmitted. It is not efficient for large character sets such as Japanese, Chinese, and Korean. On the other hand, MHTML only requires a Java enabled browser on the client, and the author does not have to modify the source document.

Also, the input of characters is not considered in Web Fonts. In MHTML, it is possible to input characters without requiring fonts on the client. Moreover, it is easy to handle *gaiji* in MHTML as described in the next section. Gaiji refers to characters that are not included in existing character sets. Since none of the existing character set standards for CJK (Chinese, Japanese, and Korean) languages covers all the characters used, it is important for applications that support these languages to be able to display gaiji characters.

3.2.3 Mixture of Multilingual Text and Handling of Gaiji

As described in Section 3.2.1, an internal coding system, which is independent from the encoding scheme and the language used in the source HTML document, is used for the character coding system of the text part. As the result, it is possible for MHTML to display a document written in multiple languages by creating an HTML document using a multilingual encoding scheme such as ISO-2022-JP-2, and preparing the corresponding fonts on the MHTML server.

Furthermore, it is very easy for MHTML to display gaiji, which is particularly

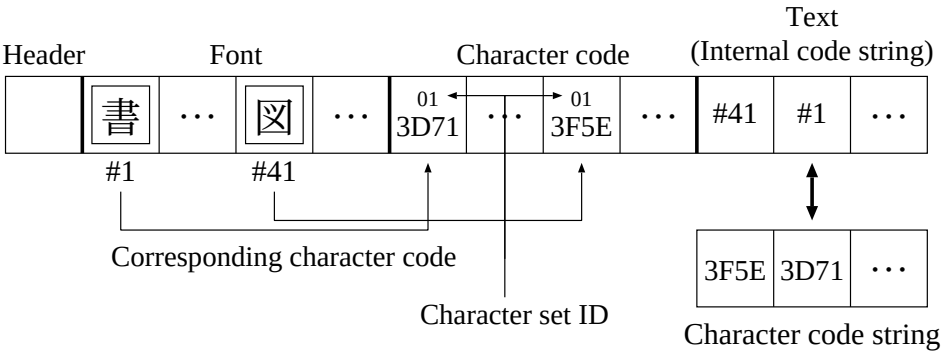


Figure 3.7. Outline of an extended MHTML document.

important for some applications such as OPACs for Asian languages. It only requires preparing a font that contains gaiji and registering it as a new character set in the MHTML server.

3.2.4 Extended MHTML Document Format

An extended version of the MHTML document format is defined in order to realize the text input function, which is required for HTML forms, etc. In the extended MHTML document format, an identifier of the character encoding scheme and the character code are added for each character in the document, in addition to the components of the basic MHTML document format shown in Figure 3.4. Figure 3.7 shows the outline of an extended MHTML document.

A source character code can be reproduced from an extended MHTML document by replacing every character in the internalized text with its corresponding source character code. The character encoding identifier is required to make the re-converted text conform to ISO-2022-JP-2. Conversion to Unicode is also possible.

We developed a Japanese text input system which does not require fonts on the client using this format. The information of the character code is added to the basic MHTML format, because it is needed for the server, for example a search engine, to interpret the submitted string. The detailed architecture of the text input system will be described in Section 3.3.6.

3.3 Implementation of the Browsing System

3.3.1 MHTML Document Format

Figure 3.9 shows the MHTML document format (version 0.3). An MHTML document consists of three components; a header, a font part, and a text part.

Header

The header contains the version number, the number of fonts classified by their size, the byte offset to the text part, and information of each font. All of the numeric values are unsigned integer in network byte order. Actual version number is obtained by dividing the value in ”version number” by 10 (e.g. the value ”3” stands for version 0.3). The fonts classified by their size are sorted according firstly to their width, secondly to their height, by ascending order.

Font Part

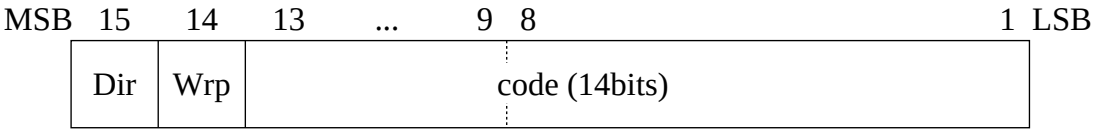
The font part contains the font glyphs appeared in the text. Font glyphs are classified by their size and placed in order of appearance in the text part. If the width of a font is not multiple of 8, padding bits are inserted in order to fit into 1 byte boundary.

Text Part

Text part is composed by ASCII and internal coding system. HTML tags are represented in ASCII, and other characters are represented in the internal coding system. Text part always starts with ASCII. The internal coding system is unsigned 16 bit integer in network byte order. Figure 3.8 shows the code structure of the internal coding system. The code starts from 11, and is assigned in the order of appearance of the character.

The 15th and the 14th bits are used for attributes of the character. The 15th bit indicates the directionality of the character and the 14th bit indicates the line-wrapping behavior of the character.

The control codes are defined in Table 3.1. The codes from 0003 to 000A is reserved for future use. The 15th (Dir) and the 14th (Wrp) bits are zero for control codes.



Dir: 0 ... Written from left to right
1 ... Written from right to left

Wrp: 0 ... Can wrap a line at a white space (e.g. Latin scripts)
1 ... Can wrap a line between any character (e.g. CJK scripts)

Figure 3.8. The code structure of the internal coding system in MHTML.

Table 3.1. The control codes for the MHTML internal coding system.

Code	Function
0000	Switch between ASCII and the internal coding system
0001	Carriage return
0002	Space character

3.3.2 Extended MHTML Document Format

Figure 3.10 shows the extended MHTML document format (version 0.2). An extended MHTML document consists of four components; a header, a font part, a code part, and a text part. The components except for the code part has the same structure as the basic MHTML document format described in the previous section.

In the code part, a pair of the character set ID and the character code are stored for each font, in the order of appearance in the font part.

3.3.3 MHTML Server

The MHTML server has two primary functions. One is the function to fetch the source HTML document from a Web server, and to convert it into MHTML and send it to the browser. The other is to send the MHTML class, which is required to display MHTML documents, to the browser. Both functions use HTTP protocol version 1.0[10] for network connections.

The source HTML document in the language-dependent format (e.g. in Japanese,

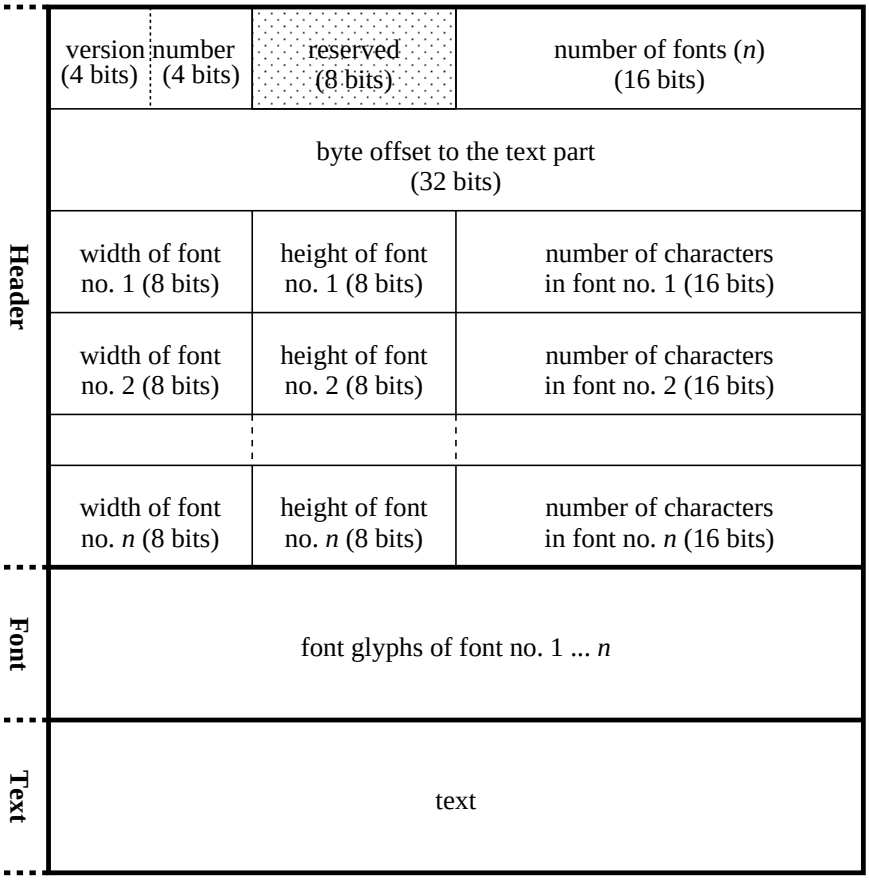


Figure 3.9. MHTML document format.

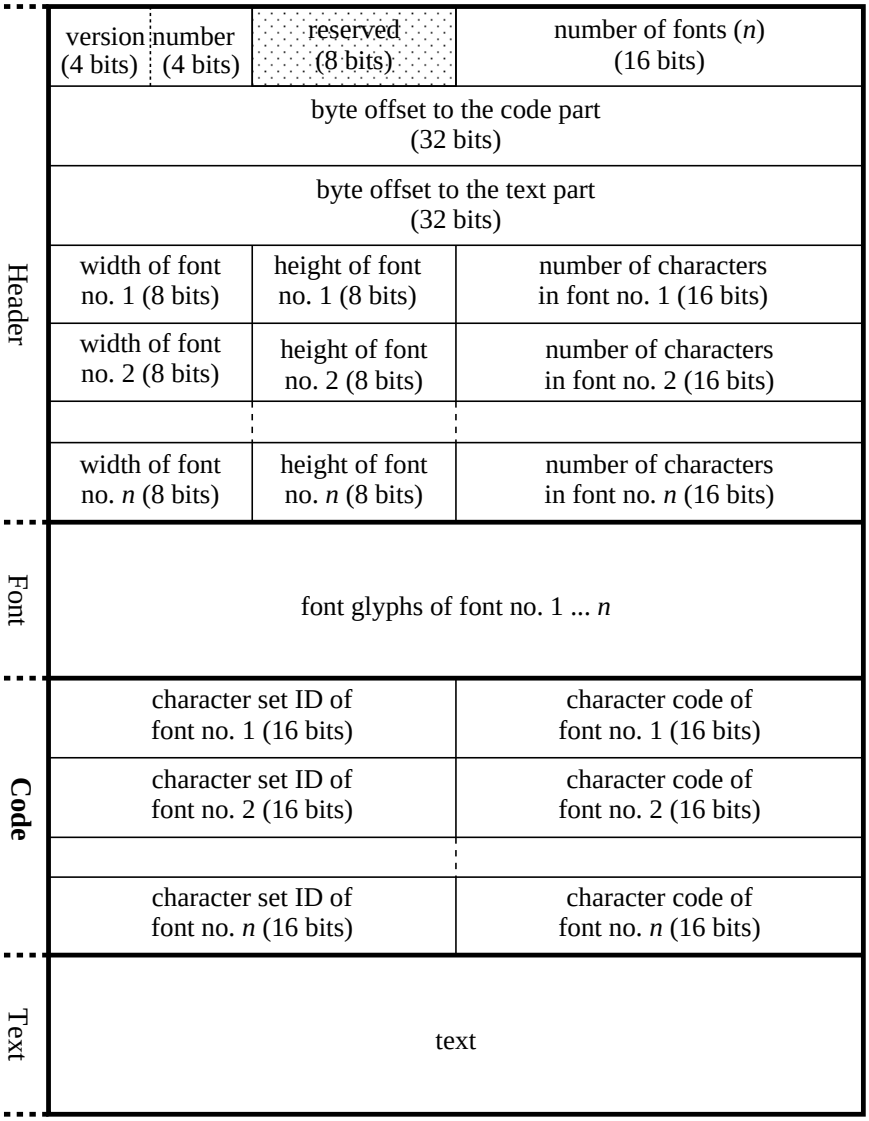


Figure 3.10. Extended MHTML document format.

Table 3.2. Parameters of the MHTML Panel class.

Parameter	Value
url	URL of the source HTML document
encoding	Encoding ID of the source HTML document

Shift_JIS or EUC-JP) is first converted into ISO-2022-JP-2 encoding scheme[86], before converting into MHTML. This process can generalize the MHTML conversion function to be independent from the encoding scheme, and this makes it easy to add the supported encoding scheme or language by simply adding a module to convert from language-dependent encoding scheme into ISO-2022-JP-2. In addition, this makes it possible to display multiple languages in one document, by using ISO-2022-JP-2 as the encoding scheme of the source HTML document.

For the format of fonts, PCF (Portable Compiled Format) for X-Window system can be used. The MHTML server (version 1.4.0) supports Japanese, Chinese (both simplified and traditional), Korean, Thai, Russian, Arabic, and some Eastern-European languages.

The MHTML server is written in C, and it is tested to be compiled and to work on major UNIX systems. The source code of the MHTML server is distributed as a free software.

3.3.4 MHTML Class

As described earlier, MHTML class is realized as a Java applet. The class files are loaded from an MHTML server into the Web browser on the client, and it allocates an arbitrary size of scrollable area on the browser, and then it displays an MHTML document. Multiple instances of MHTML classes can be embedded in one page.

The client first loads an HTML page to load MHTML class. This HTML page contains parameters shown in 3.2 within an APPLET tag. These parameters are sent to MHTML server through MHTML class.

Then, the MHTML server fetches the HTML document specified by “url” parameter from the Web server, converts it from the encoding specified by “encoding” parameter into ISO-2022-JP-2, then converts it into MHTML, and returns it to the MHTML class on the client’s browser. Finally, the document is displayed on the MHTML class.

Table 3.3. Request types supported by MHTML server and content of the response.

Request type	Meaning of the request	Parameter(s)	Response
INITIAL	Initial request from a user	none	An HTML document to invoke MHTML Class as an applet.
APPLET	Request for a class or an image file	File name	A class file, an icon image, etc.
MHTML	Request for an MHTML document	URL, encoding	An MHTML document

The MHTML class also supports navigation through hyperlinks. In this case, “url” and “encoding” parameters of the link are embedded in the URL for requesting the document and sent to the MHTML server.

Figure 3.3 shows the request types supported by MHTML server, required parameters for the request, and the content of the response.

MHTML class supports basic HTML tags. Inlined images are supported by replacing them into hyperlinks designating the images. This is because the layout algorithm will be complicated if images are embedded in a document, and it may take time to load images. Figure 3.4 shows HTML tags supported by MHTML class.

3.3.5 Support for Bidirectional Text

The MHTML technique has been extended to support Arabic script[35].

Arabic, one of the Semitic languages, is written from right to left, while Arabic digits and Latin characters are shaped from left to right[8]. Arabic is spoken by millions of people worldwide, and is being increasingly used on the Internet[2]. Arabic characters are stored in logical order different from the visual order. The logical order corresponds to the reading/typing order. This means that the internal storing structure does not correspond to the visual structure which depends on the contextual form of the Arabic language. The conversion from the logical into visual order is the main issue in rendering Arabic, and Hebrew that is also written right to left[21].

An Arabic character might take on four different glyphs depending on its position in the word, i.e. at beginning, in the middle, surrounded and standalone. However, popular Arabic coding systems, i.e. ISO-8859-6 and Windows-1256, do not assign

Table 3.4. HTML tags supported by MHTML Class.

Tag	Meaning
TITLE	Title of a document
H1, H2, H3, H4, H5, H6	Headings
P	Paragraphs
PRE	Preformatted text
ADDRESS	Contact information
BLOCKQUOTE	Block-level quotation
UL	Unordered list
OL	Ordered list
LI	List item
DL	Definition list
DT	Term in definition list
DD	Description in definition list
EM	Emphasis
STRONG	Stronger emphasis
B*	Bold
I*	Italic
A	Anchor
BR	Forcing a line break
HR	Horizontal rule
IMG	Inline images
CENTER*	Centering
FONT*	Altering font size and color

$*$ indicates elements that are deprecated or obsolete in HTML 4.01 specification[91], but these are still widely used in current Web documents. MHTML Class supports those elements for backward compatibility.

different codes for those glyphs, thus the rendering engine have to employ some contextual analysis in order to handle those different glyphs according to their positions. In addition, Arabic is a cursive language, i.e. characters are linked together. Rules exist to define how to link characters, but the font still need to be adapted to this form. Furthermore, there are some mandatory ligatures. It is also the responsibility of a rendering engine to correctly handle those ligatures.

Figure 3.11 shows an example of displaying a bidirectional Arabic text.

3.3.6 Text Input System

Text input function, which is required in HTML forms, is realized by the *Text Input System (TIS)*. It supports *romanji-kanji conversion* for clients which do not have fonts and input methods of Japanese, by communication between the client and TIS. Figure 3.12 shows the overview of TIS.

A Java applet called TextInput class first receives user's input in romanji (a transliteration of Japanese characters which is represented in Latin alphabet) and sends it to the Text Input server. Figure 3.13 shows the image of TextInput class applet.

Text Input server converts inputted string from romanji to kana (a syllabic script used by the Japanese writing system) and converts from kana to kanji by the conversion server, and returns the conversion candidates to the TextInput class. The user selects the appropriate one from conversion candidates displayed on TextInput class, and goes to the next phrase. The user's desired string will be fixed by repeating this process. The character code string will be sent to the Web server when the user presses the "submit" button. The extended MHTML document format described in Section 3.2.4 is used because the conversion result must be sent to the Web server in a character code string.

As the kana-kanji conversion server, Wnn version 4.2[83] is used. Although TIS supports only Japanese at the moment, support for other languages can be added by integrating a conversion server for the language. Since Chinese and Korean use input methods similar to Japanese in which the source input text is Latin characters, it is feasible to input from a Latin keyboard. However, Thai and Arabic, for example, use special keyboard layouts to input characters of these languages[64]. In such a case, it might be better to use a virtual keyboard[9, 35] instead of using a conversion method, because it is usually difficult for a user to find appropriate key for a particular character using a Latin keyboard without a label of that character.

Figure 3.11. An example of displaying a bidirectional Arabic text.

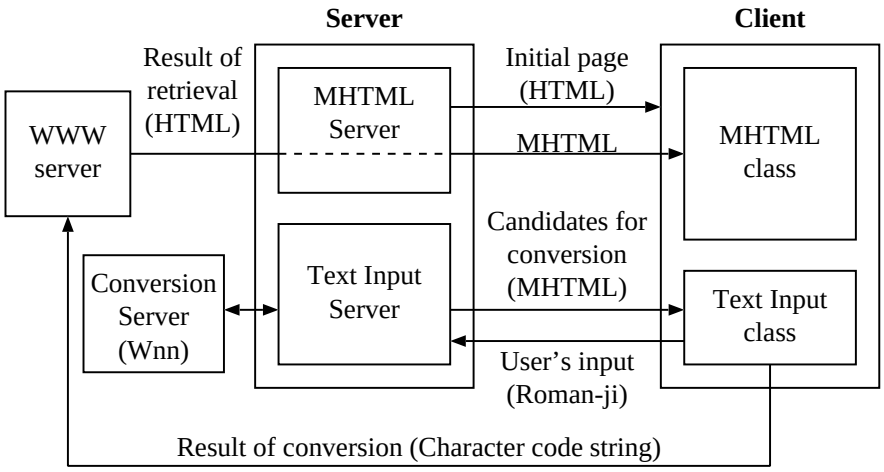


Figure 3.12. Overview of the Text Input System.

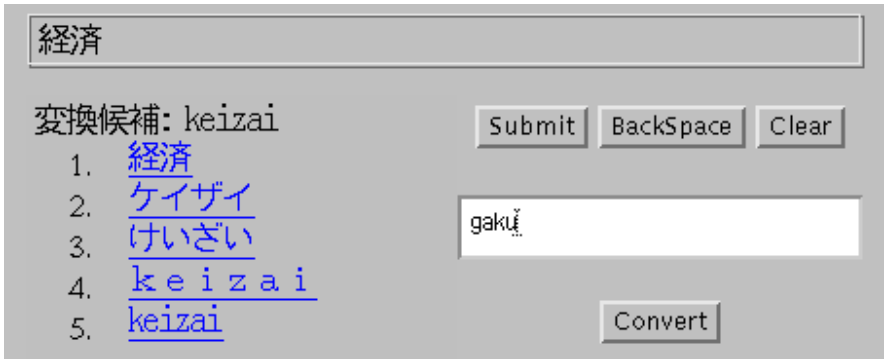


Figure 3.13. The image of the TextInput class.

3.3.7 Printing MHTML Document

Since an MHTML document is displayed on a Java applet, it cannot directly printed from the Web browser using the "print" function. In order to support the printing function in MHTML, the MHTML class implements a function to create the GIF image of the currently displayed MHTML document. It is invoked by pressing the "print" button on the applet. Since it is impossible for a Java applet to directly display the created image outside of the applet (i.e. the Web browser) for security reasons, somewhat redundant process is required to realize this function. The image created on the client is first sent to the MHTML server, and just sent back to the client without any processing, and finally displayed on the client Web browser. The user can then print this image using the built-in "print" function of the browser.

3.4 Example Applications Based on MHTML Technique

3.4.1 Browsing Service

Since 1996, we have been offering a browsing service which enables users browse Web documents without requiring fonts[73, 68]. This service has been opened to the public and freely accessible[3].

The MHTMLViewer class, which is an extension to MHTML class, is used for this service. In the MHTMLViewer class, URL input field, navigation buttons, and a pull-down menu to choose the document language, are added in addition to the core functionalities of the MHTML class. This service provides users with easy access to Web documents written in, for example Japanese, from anywhere in the world, regardless of whether the fonts are installed or not.

Figure 3.14 shows an example of viewing Korean and Thai documents using the browsing service.

[3] http://mhtml.ulis.ac.jp/

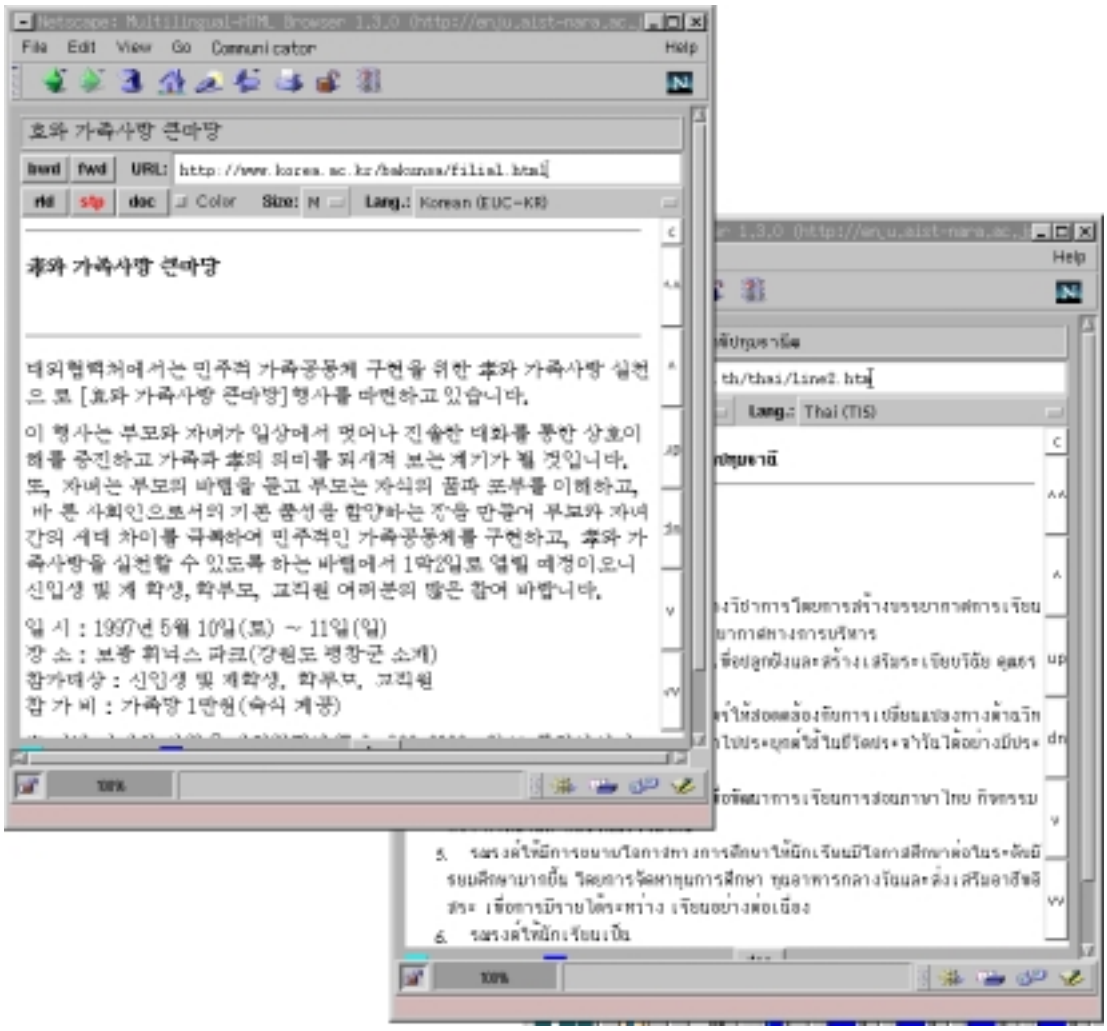


Figure 3.14. An example of viewing documents using the browsing service.

3.4.2 Japanese Old Tales Collection

We applied MHTML to develop the user interface for a multilingual electronic text collection of Japanese old folk tales[19, 20][4]. The tales were taken from Japanese old tales and rewritten by volunteer authors. Each tale is written in one or more languages, including English, French, German, Spanish, Portuguese, Italian, Bulgarian, Russian, Japanese, Chinese, and Korean.

Figure 3.15 shows an example of viewing a Japanese old tale in three languages in parallel. In this example, three MHTML class applets are embedded in a table.

Figure 3.16 shows the overview of the system. The client first loads an HTML document which contains three APPLET tag (1), and MHTML class is loaded into the browser (2). Each MHTML class sends parameters such as URL of the source text, to the MHTML server. The MHTML server fetches the source text (3), converts it into MHTML, and returns it into the MHTML class on the client browser (4).

3.4.3 OPAC System

A simple OPAC (Online Public Access Catalog) system has developed using MHTML technique[30]. Figure 3.17 shows the user interface of the OPAC using MHTML. The user interface consists of two Java applets. The MHTML class at the top displays the retrieval result, and the Text Input class at the bottom receives and displays the word or phrase to be used as a search term. Since this is a very simplified example, the search term given is used for a full-text search in the whole OPAC database. This user interface can be extended to implement more elaborate search commands.

3.4.4 Text Retrieval System

An experimental SGML-based text retrieval system for multilingual documents has developed using MHTML[94]. It can currently store and retrieve Japanese and English (ASCII) documents, and it is extensible to any languages. Its user interface was designed using MHTML in order to provide users with ubiquitous accessibility to documents written in multiple languages.

[4] http://www.DL.ulis.ac.jp/oldtales/

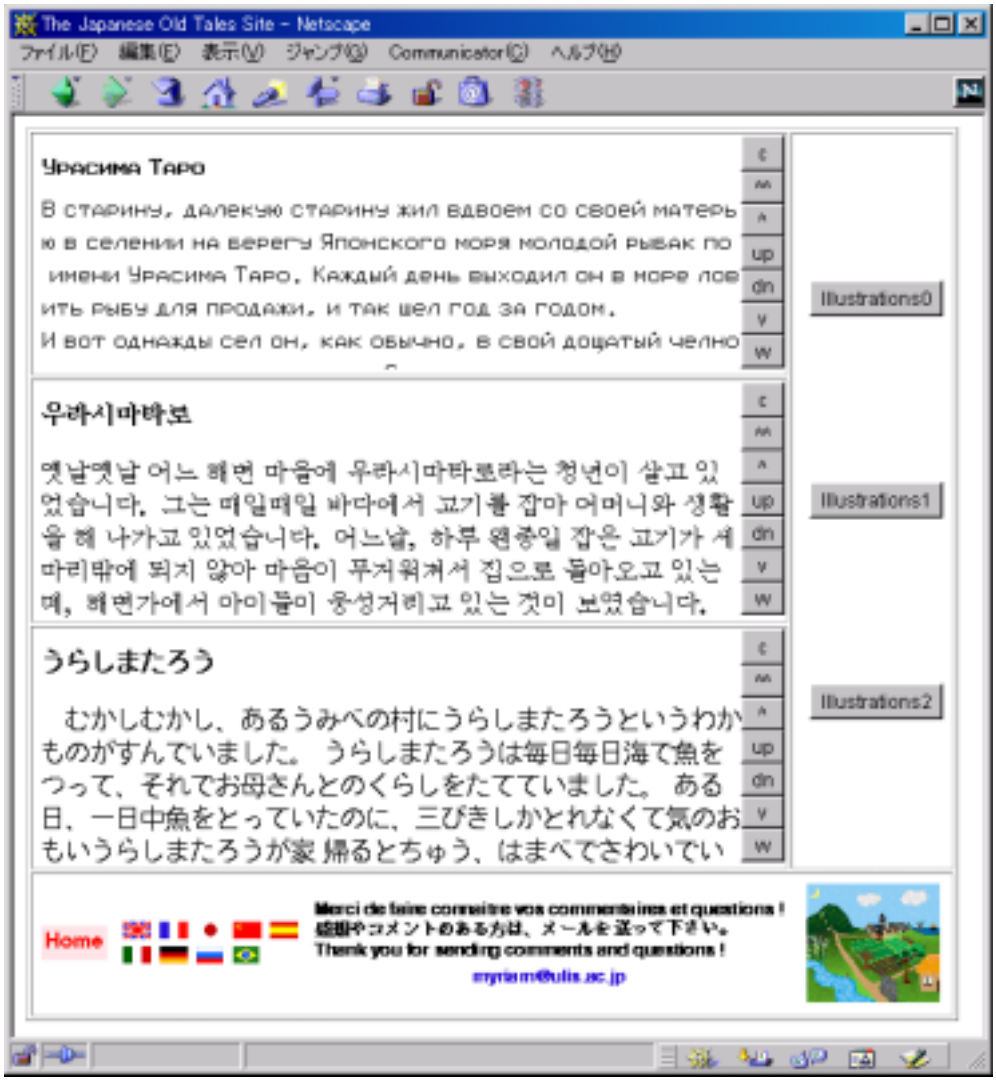


Figure 3.15. An example of displaying a Japanese old tale in the collection.

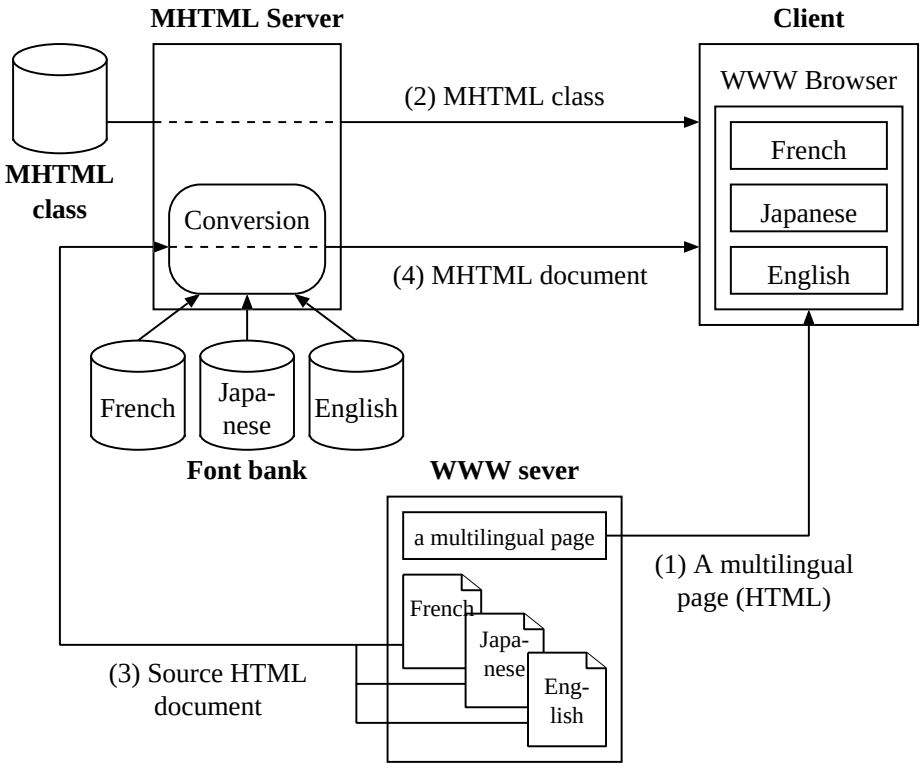


Figure 3.16. Overview of the Japanese old tales collection system.

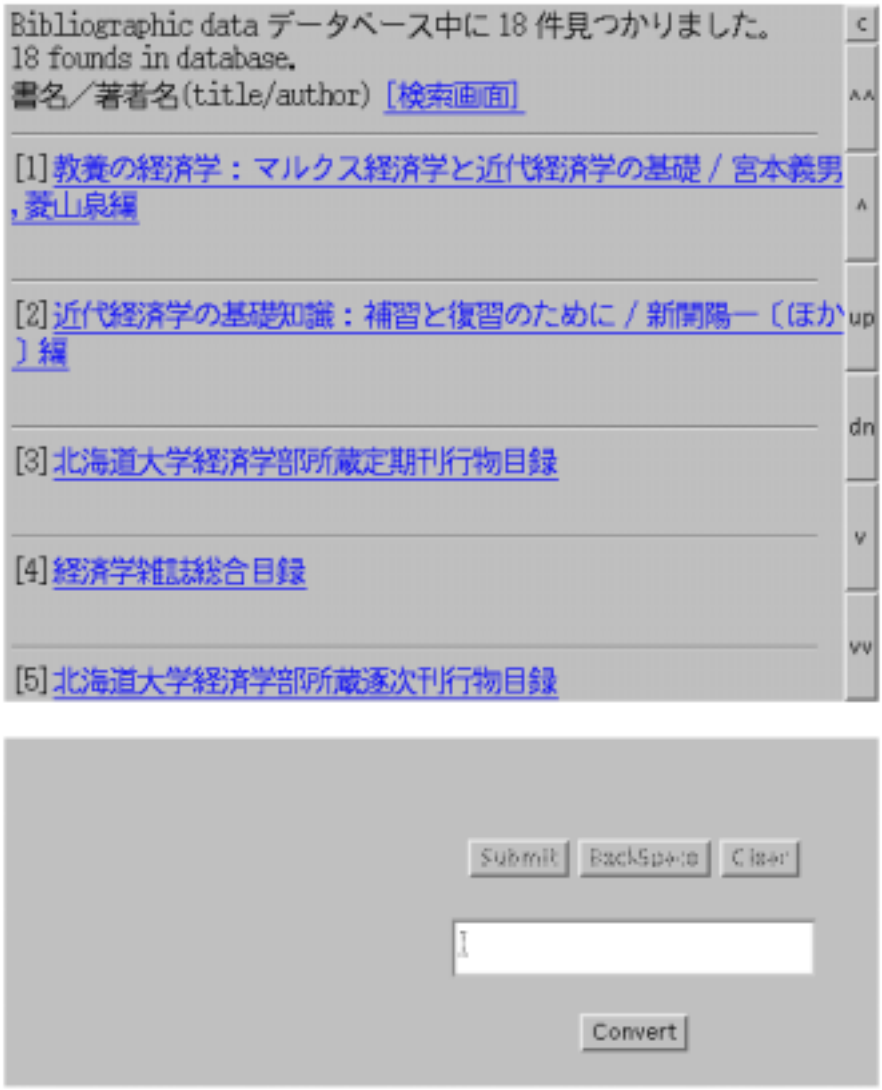


Figure 3.17. OPAC interface image.

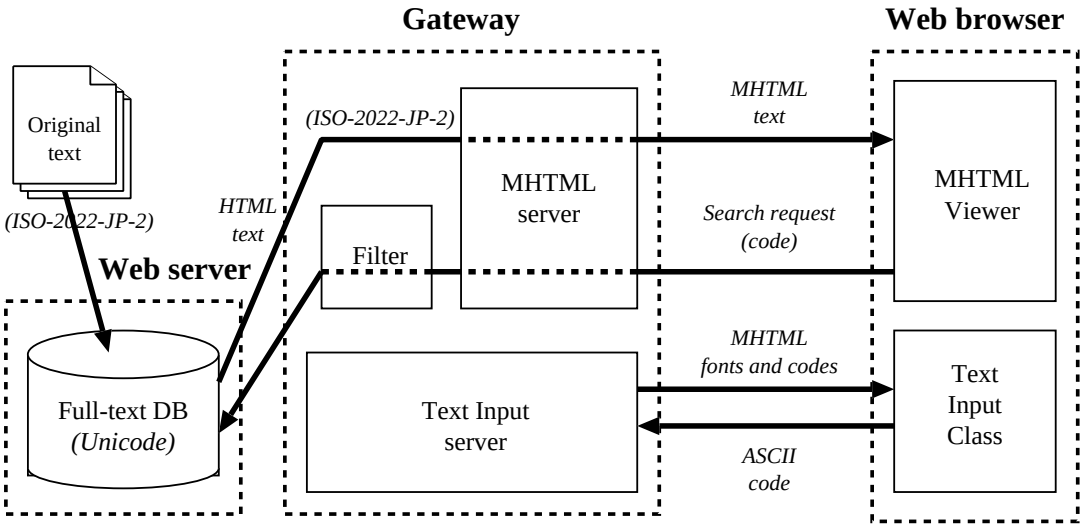


Figure 3.18. Overview of the text retrieval system for multilingual documents.

Figure 3.18 shows the overview of the system. It receives documents encoded in various encoding scheme and converts it into the ISO-2022-JP-2. The text is converted into Unicode to create the index, and the text is also stored as is. As described in Chapter 2.3.4, ISO-2022-JP-2 defines the escape sequence to switch between multiple character sets. This feature is convenient for use with an existing document encoded in various encoding scheme in the internationalized environment. However, this encoding scheme makes the text retrieval function complex, because of the statefulness. Since the flat character space given in Unicode makes the text retrieval function simple, Unicode is employed as the basic encoding scheme to build the index. The index is created based on *n*-gram, and its user interface is created using extended MHTML. Thus, this system provides a framework for retrieving texts encoded in a multiple encoding scheme (i.e., multilingual documents) with a universal user interface for retrieval.

3.5 Summary

In this chapter, we proposed a system for providing multilingual HTML documents without requiring to install fonts on the client. This system might be especially useful for documents written in minor languages, for old literature which may include charac-

ters that are not defined in any standard character set, and for newspaper articles which may require gaiji's for some proper nouns.

Despite the name of MHTML, it is entirely independent from HTML. Therefore, it can also be used to encapsulate any text formats that are based on plain text, such as electronic mail and XML (eXtensible Markup Language). For example, it might be useful for a Japanese user to be able to read and write electronic mails in Japanese from a public Web terminal (e.g. in a public library) anywhere in the world.

The techniques employed in this system are rather simple, but more efficient than other approaches such as Web Fonts, CII, and page image conversion. Also, this technique is extended to the text input function, which is important for an interactive use of the Web.

Chapter 4

Automatic Identification of Coding Systems and Languages of Web Documents

4.1 Introduction

With the growth of the Internet and WWW in recent years, documents written in various languages are being provided. Although 80% of current Web pages are written in English, it is estimated that over a half of Web documents will be non-English in 2003. In proportion to this growth, character coding systems used for Web documents also keep on increasing.

However, Web documents often lack information on the language and coding system they use. A mechanism to indicate those information to Web documents is already developed, but it is not very popular at this moment. In such a case that those information are not indicated, recipients of the document have to presume the coding system and language used. And if the presumption is wrong, it might result in troubles such as incorrect display on Web browsers. Besides, Web search engines create an index for collected documents, and lack of those information might cause inaccurate indexing. Even in search engines that support only one language, if the language were not identified for collected documents, it is most likely that documents in other languages will also be indexed, and it might cause a significant loss in precision of retrieved documents. Furthermore, in addition to search engines, other document processing

applications such as categorization, summarization, and machine translation are also dependent on knowing the document language. Of course, coding system has to be identified before identifying the document language.

Although the use of multilingual coding systems such as ISO/IEC 10646-1[45] and Unicode[100] will hopefully avoid such problems, it will take a long time for existing enormous amount of documents to be replaced by such coding systems. Therefore, at least in the present circumstances, languages and coding systems have to be identified prior to processing, for Web documents to be correctly processed.

In this chapter, we propose a method which automatically and efficiently identifies both languages and coding systems of Web documents by employing a simple statistical technique and heuristics together.

4.1.1 Current Status and Issues on WWW

As mentioned in Chapter 2, various languages and coding systems are used for documents on WWW.

For specifying the coding system information to an HTML document, *charset* parameter in *Content-Type* header of HTTP[27], and the same header and parameter in META tag of HTML[91] are defined. However, usually these information are not specified. In a simple analysis of several thousands of Web documents cached in a proxy server, only about 30% of documents indicate the coding system information.

On the other hand, for specifying the language information to a document, *lang* attribute in HTML 4.0[91], and *language tag* in Unicode 3.0[100] are defined. However, usually these are not used because of lack of supported applications.

Therefore, automatic identification of languages and coding systems can be regarded as an indispensable technique for WWW information systems[75].

4.2 Related Work

4.2.1 Language Identification

Automatic language identification of text documents is well researched mainly for European languages. Employed techniques include statistical-based ones such as *n*-

gram[14, 1] and hidden Markov model[25], and heuristic-based ones such as counting frequencies of characters or words that are peculiar to the language[32].

For example, in the method of Cavnar et al.[14] which is based on *n*-gram of characters, they use the similarity of the frequency distribution of 1 to 5-grams. In their experiment, over 98% correct rate was achieved for 8 European languages included in ISO-8859-1. The procedure of their language identification method is as follows:

1. For training data of each language to be identified, calculate all *n*-gram ($n = 1 \ldots 5$) frequencies and sort them by reverse order of frequency (the resulting list is called a *profile*),
2. Create a profile for the document to be identified,
3. Calculate the *out-of-place* distance (the sum of the differences in ranks for each *n*-gram between both profiles) between the document profile and each language profile, and the language with the smallest distance is picked as the identification result.

Although the target languages for their experiment were only European, this method can be applied to languages which have no clear segmentations between words. Also it can be applied for identification of coding systems in which the number of bytes used for one character is variable, such as CJK. Because their method is based on *n*-grams of bytes, it does not depend on word or character segmentations. However, this method is less efficient because it needs to extract, sort, and compare all *n*-grams in the target document.

4.2.2 Coding System Identification

On the other hand, for automatic identification of coding systems, the algorithms are well established for some languages such as Japanese, and are already incorporated into applications such as Web browsers and code conversion filters. These algorithms check differences of code ranges that do not overlap for each coding system (e.g. Shift_JIS, EUC-JP, and ISO-2022-JP for Japanese), and are accurate enough even for short texts. However, such algorithms are not applicable for coding systems that have the same or similar code ranges, such as language versions of EUC (Extended Unix Code).

As a method to support multiple languages, Kikui[61] proposes a method which uses statistical language model. His method first extracts East-Asian language parts in the target document using decoder for each East-Asian language. Then, it calculates the likelihood score of each decoded character, which is obtained from the training data, and the coding system which has the highest average score is selected as the result for the East-Asian language part. The same method is applied for each 4-gram at the tail of a word in the remaining one-byte part. In his experiment, over 95% correct rate was achieved for 15 languages and 11 coding systems. However, this method needs to decode the target document assuming every supported languages and coding systems. Therefore, the addition of the language and coding system to support might lead a significant drop in performance.

In this chapter, we propose an identification method, which is more efficient than previous methods, and less dependent on the length of the document to be identified, primary for use with the indexing of collected documents on Web search engines which support multiple languages[74, 75].

4.3 Target Languages and Coding Systems

Our proposed method supports 12 languages and 10 coding systems as shown in Table 4.1. For ISO-8859-1 coding system, it supports 9 languages, namely, English, German, French, Italian, Spanish, Portuguese, Danish, Norwegian, and Swedish. Although ISO-8859-1 supports 14 languages, we only support 9 languages because of availability of training data for other languages. Besides, Chinese coding systems GB2312 and Big5 handle different coded character sets, we regard these as different languages.

In the table, “Unit” indicates the number of bits actually used for one octet, and “Character range” indicates the code range within which graphic characters are assigned.

Although the target coding systems except for ASCII and ISO-8859-1 are all variable length multi-byte coding systems, our algorithm treats all coding systems as one byte code, regardless of the segments of multi-byte characters, for simplicity and efficiency.

Table 4.1. Target languages/coding systems for identification.

Coding System	Language	Unit	Character range
ASCII	English	7	33–126
ISO-2022-JP	Japanese	7	33–126
ISO-2022-CN	Chinese	7	33–126
ISO-2022-KR	Korean	7	33–126
Shift_JIS	Japanese	8	33–252
EUC-JP	Japanese	8	33–126, 142–254
GB2312	Chinese (simplified)	8	33–126, 161–254
Big5	Chinese (traditional)	8	33–126, 161–254
EUC-KR	Korean	8	33–126, 142–254
ISO-8859-1	European languages	8	33–126, 161–254

4.4 Training Data

4.4.1 7-Bit Coding Systems

Because ASCII and ISO-2022 variants are 7-bit coding systems, those can be distinguished from other 8-bit coding systems by checking if the document contains a byte in which MSB (Most Significant Bit) is 1. Moreover, documents written in ISO-2022 variants necessarily contain some escape sequences that is used to designate (and to invoke) a coded character set to be used after it. Therefore, subsets of ISO-2022 (-JP, -CN, and -KR) can be distinguished by checking those escape sequences.

Consequently, for all 7-bit coding systems mentioned, it is possible to infallibly identify those coding systems without using any training data.

4.4.2 8-Bit Coding Systems

8-bit coding systems are identified by analyzing the distributions of character codes for every one byte or every consecutive two bytes.

For the training data for analysis, we used the directory pages of country/language versions of Yahoo! directory service. The languages and coding systems used in each country/language version are shown in Table 4.2.

Table 4.2. Country/language version of Yahoo! used for training data.

Site	Language	Coding system
Yahoo! Japan	Japanese	EUC-JP
Yahoo! China, Yahoo! Chinese	Chinese (simplified)	GB2312
Yahoo! Taiwan, Yahoo! Chinese	Chinese (traditional)	Big5
Yahoo! Korea	Korean	EUC-KR
Yahoo!	English	ISO-8859-1
Yahoo! Deutschland	German	ISO-8859-1
Yahoo! France	French	ISO-8859-1
Yahoo! Italia	Italian	ISO-8859-1
Yahoo! España, Yahoo! en Español	Spanish	ISO-8859-1
Yahoo! Brasil	Portuguese	ISO-8859-1
Yahoo! Danmark	Danish	ISO-8859-1
Yahoo! Norge	Norwegian	ISO-8859-1
Yahoo! Sverige	Swedish	ISO-8859-1

We extracted only the descriptions of sites from all directory pages of each country/languages version, in order to avoid fixed descriptions that are identical among all pages. From the extracted sentences, randomly selected 100 kilobytes were used for the training data. For the training data of Japanese Shift_JIS, sentences converted from EUC-JP using a coding system converter were used.

One byte code distributions of the training data for some 8-bit coding systems are shown in Figure 4.1. In these figures, the horizontal axis is the one byte code values (from 33 to 254), and the vertical axis is the occurrence frequencies for each code values.

Some characteristics in distributions of code occurrence frequencies for each coding system can be observed from these figures. For example, in Shift_JIS and EUC-JP, the first bytes of hiragana and katakana, that are frequently appeared in Japanese text, converge into a particular code range (i.e. 164-165 for EUC-JP), and are relatively high values. Similarly, in EUC-JP, code ranges for the first bytes of Greek and Cyrillic, and unused characters (166-174) are relatively low values compared to that of other two-byte codes (161-254).

However, no significant characteristics were observed among GB2312, Big5, and EUC-KR, and among languages of ISO-8859-1.

4.5 Automatic Identification Algorithm

4.5.1 Parameter Method

This method is applied for Shift_JIS and EUC-JP, in which some significant characteristics were observed, and all 7-bit coding systems. In the method, firstly the frequencies of code values (0-255) for each one byte in the document are calculated. Then, identification algorithm, which is derived heuristically from the statistical characteristics of the frequency distributions of training data, is applied. The algorithm is shown in Figure 4.2.

The input of this algorithm is the frequency distribution of the document to be identified, and the output is the name of the identified coding system. $freq(n)$ is the frequency of the code value n, and $avg\_freq(m \ldots n)$ is the average frequency for the range from the code value m to n.

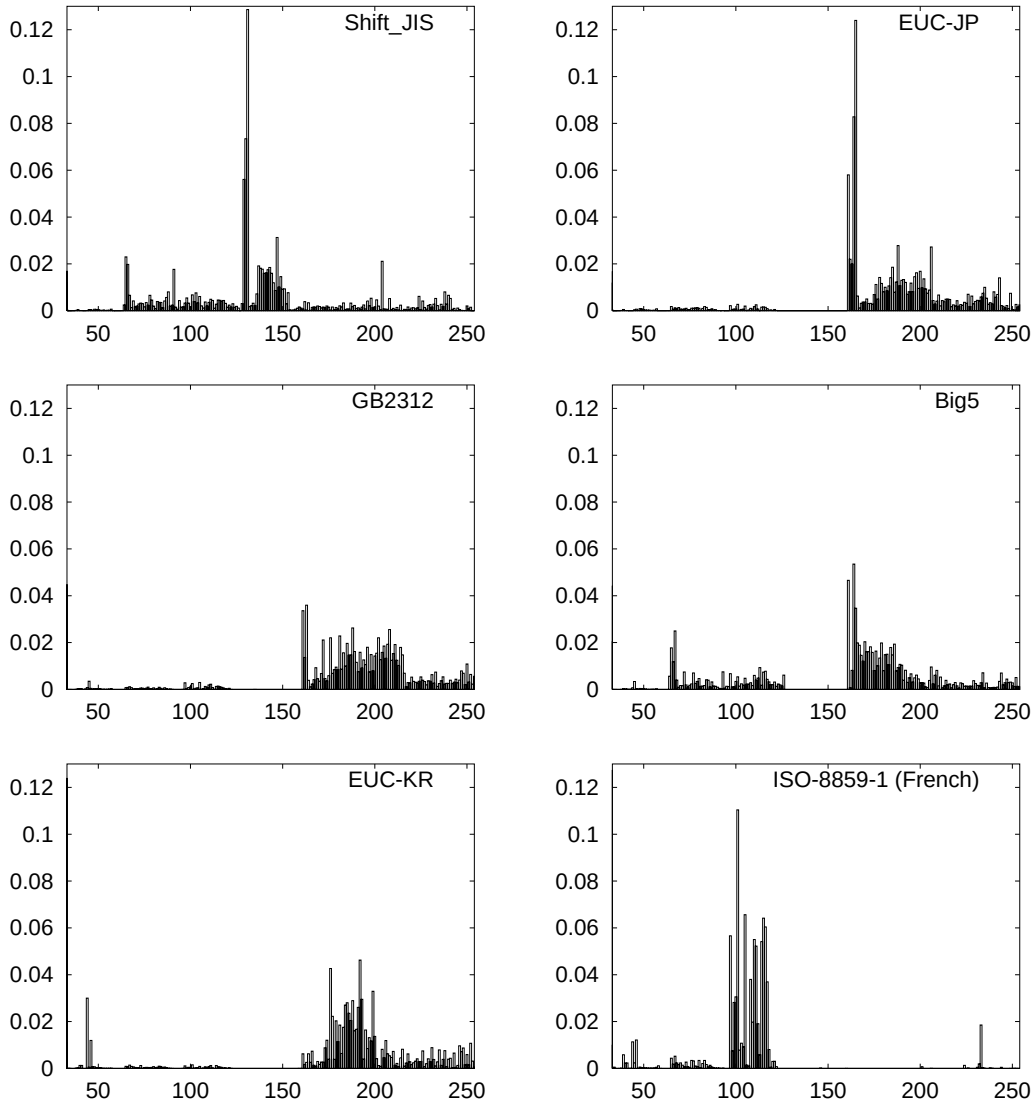


Figure 4.1. One byte code distribution of 8-bit coding systems.

if $avg\_freq(128 \ldots 255) = 0$
    if $freq(ESC) > 0$
        if $appears(ESC, 36, 64)$ or $appears(ESC, 36, 66)$ or
            $appears(ESC, 40, 74)$ or $appears(ESC, 36, 40, 68)$
            return "`ISO-2022-JP`"
        else if $appears(ESC, 36, 41, 65)$ or $appears(ESC, 36, 41, 71)$ or
            $appears(ESC, 36, 42, 72)$
            return "`ISO-2022-CN`"
        else if $appears(ESC, 36, 41, 67)$
            return "`ISO-2022-KR`"
        else
            return "`ISO-2022 (others)`"
        end if
    else
        return "`Unknown`"
    end if
else if $avg\_freq(128 \ldots 159) \neq 0$ and $(freq(130) > FK$ or $freq(131) > FK)$
    return "`Shift_JIS`"
else if $avg\_freq(166 \ldots 174) < FS$ and $(freq(164) > FK$ or $freq(165) > FK)$
    return "`EUC-JP`"
else
    return "`Unknown`"
end if

Figure 4.2. Coding system identification algorithm of the parameter method.

ESC is the special character (code 27) that indicates the beginning of an escape sequence for ISO-2022, and $appears(code\_seq)$ returns true if the specified code sequence *code_seq* is appeared in the document.

For the identification of Shift_JIS and EUC-JP, significant characteristics observed in the one byte code distributions are used. Specifically, we assume that the frequency of the code value of the first byte of hiragana or katakana is always the highest for documents written in Shift_JIS or EUC-JP. Therefore, we can set the threshold value to be the minimum frequency of the code that has the highest frequency in each document, among the training data for these coding systems. Similarly, we can also use relatively less appeared code value, such as the first byte of Greek and Cyrillic characters in EUC-JP. In this case, the maximum average frequency of the code range of these characters in each document in the training data is used. In addition, since Shift_JIS uses a code range that is not used in other coding systems (128-159), it can also be used for the identification.

The thresholds are defined as follows:

FK is the minimum value of the frequency of the most frequently appeared code value (i.e. the first byte of hiragana and katakana) for each document in the training data of Shift_JIS and EUC-JP.

$$FK = \min_{n}(max\_freq(0 \ldots 255)) \tag{4.1}$$

where n is the number of documents in the training data, and $max\_freq(m \ldots n)$ is the maximum frequency for the range from the code value m to n.

FS is the maximum value of the average frequency of relatively less appeared code value (i.e. the first byte of Greek and Cyrillic, and unused characters) for each document in the training data of EUC-JP.

$$FS = \max_{n}(avg\_freq(166 \ldots 174)) \tag{4.2}$$

The values of these parameters determined from the training data described in section 4.4.2 were $FK = 0.06, FS = 0.005$.

ASCII is not identified here despite that it is a 7-bit coding system, because it cannot be distinguished from ISO-8859-1 documents that do not contain any accented alphabets.

4.5.2 Vector-Distance Method

Using One Byte Unit (Unigram)

On the other hand, if we regard the frequencies of code values as a vector, we can consider an alternative method for identification by comparing their distances. In the method, in the same way as the parameter method, firstly the frequencies of code values (0-255) for each one byte in the document are calculated. Then, that vector is compared with vectors of each language and coding systems that are calculated from the training data in advance, in terms of their distance.

For 7-bit coding systems, ASCII and ISO-2022 are distinguished by checking the presence of a escape sequence, as described in section 4.4.1. For 8-bit coding systems, vector-distances between the target document and each languages and coding systems are calculated, and the language and coding system which have the minimum distance is selected as the result.

The vector-distance D_c between the document to be identified and the training data of the coding system *c*, is calculated by the following formula based on *cosine distance*:

$$\cos D_c = \frac{\sum_i freq_c(i)\, freq_d(i)}{\sqrt{\sum_i freq_c(i)^2}\sqrt{\sum_i freq_d(i)^2}} \quad (4.3)$$
$$(i = 65 \ldots 90, 97 \ldots 122, 128 \ldots 255)$$

where $freq_c(i)$ is the occurrence frequency of the code value *i* for the training data of the coding system *c*, and $freq_d(i)$ is the occurrence frequency of the code value *i* for the document to be identified.

The vector-distances D_c between the document to be identified and each training data for the target coding systems are calculated, and the coding system that are the closest to the document is taken as the result of the identification. In order to avoid the negative effect of characters that are not useful for identification such as symbols, numerals, and control characters, the code ranges 0-64, 91-96, and 123-127 are not counted. Figure 4.3 shows the code ranges actually used for calculation of the vector distance.

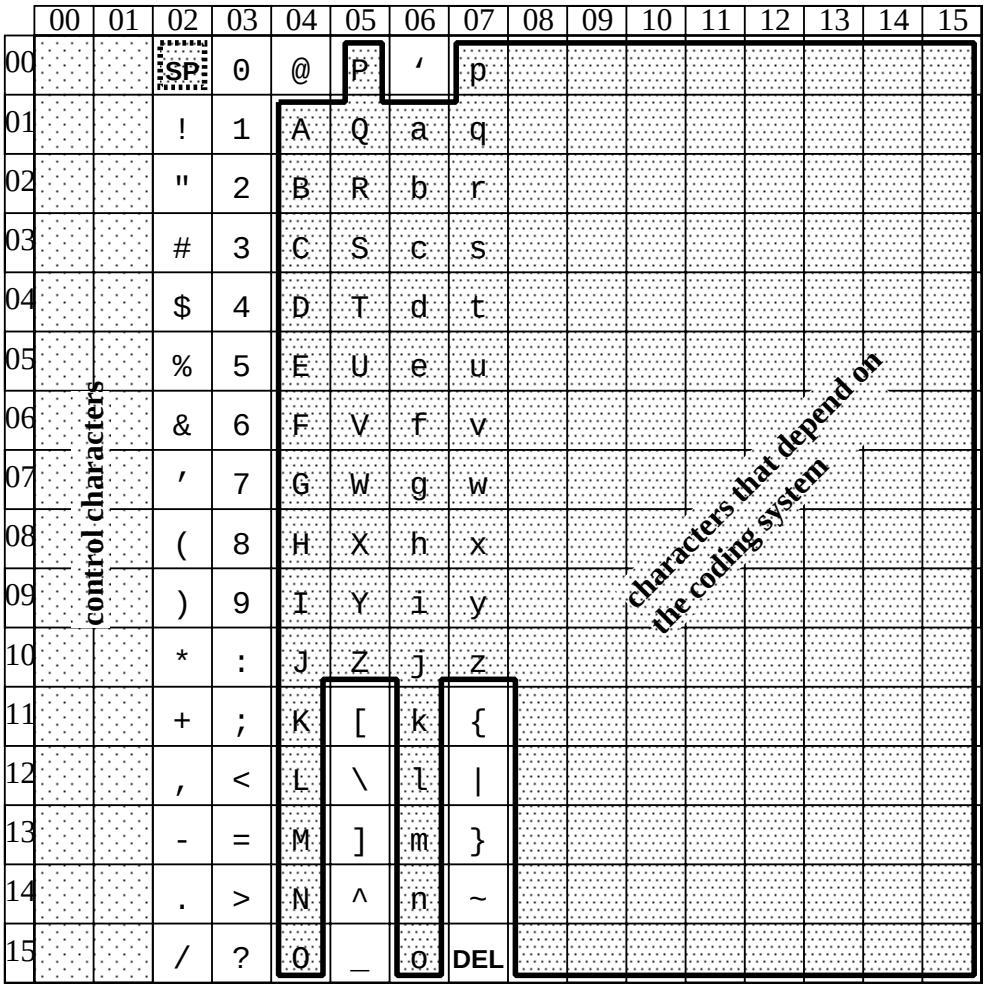


Figure 4.3. The code range actually used for calculation of the vector distance.

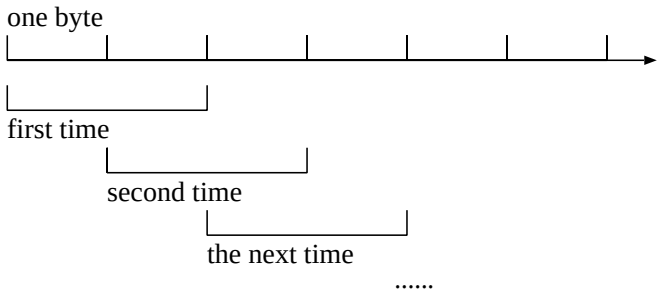


Figure 4.4. Statistics using consecutive two bytes.

Using Consecutive Two Bytes Unit (Bigram)

If we take the connection of consecutive characters into account, we can consider a method using consecutive two bytes as a unit instead of one byte unit. In this case, as shown in Figure 4.4, consecutive two bytes that slide simply one byte at a time from the beginning, regardless of the number of bytes used for one character. This method might catch the characteristics of connections of characters, that are dissimilar among languages.

In case of using consecutive two bytes unit, the vector-distance D'_c between the document to be identified and the training data of the coding system c, is calculated by the following formula:

$$\cos D'_c = \frac{\sum_i \sum_j freq_c(i,j)\, freq_d(i,j)}{\sqrt{\sum_i \sum_j freq_c(i,j)^2}\sqrt{\sum_i \sum_j freq_d(i,j)^2}} \qquad (4.4)$$
$$(i, j = 32, 65 \ldots 90, 97 \ldots 122, 128 \ldots 255)$$

where $freq_c(i,j)$ is the occurrence frequency of the code value $i \times 256 + j$ for the training data of the coding system c, and $freq_d(i,j)$ is the occurrence frequency of the code value $i \times 256 + j$ for the document to be identified.

Similar to that of one byte unit, non-useful ASCII characters such as symbols, numerals, and control characters are not counted. However, in the case of two bytes unit, it might be useful to count the characters at the beginning and the end of a word, especially for European languages, we take *space* character (code 32) into account.

Table 4.3. Correct rate of identification using the parameter method.

Coding system (language)	Correct rate (%)
ISO-2022-JP (Japanese)	100.0
ISO-2022-CN (Chinese)	100.0
ISO-2022-KR (Korean)	100.0
Shift_JIS (Japanese)	100.0
EUC-JP (Japanese)	100.0
Avg.	100.0

4.6 Evaluation

We conducted the experiment of the proposed methods in terms of accuracy. For the test data, we used the same document collection as described in section 4.4.2 (i.e. Yahoo! directory pages), but distinct set of documents to that of the training data. We used 539-1,000 documents that are longer than 300 bytes (1,255 bytes on average) for each coding system and language. The documents in Shift_JIS and ISO-2022-JP in Japanese, ISO-2022-CN in Chinese, ISO-2022-KR in Korean, were created from EUC-JP, GB2312 and Big5, and EUC-KR, respectively, and used as the test data. Sun Ultra 1 (UltraSPARC 167MHz, Solaris2.5.1) were used for the experiment, and the identification program is implemented in C language.

4.6.1 Applying Parameter Method

The execution time of the parameter method was about 0.002 seconds per document. Correct rates of identification using the parameter method are shown in Table 4.3.

Identification using the parameter method achieved 100% for all target coding systems.

4.6.2 Applying Vector-Distance Method

The execution time of the vector-distance method was, about 0.003 seconds per document for 1 byte unit, and about 0.1 seconds per document for consecutive two bytes.

Table 4.4. Correct rate of identification using the vector-distance method and Cavnar et al.'s method.

Coding system (language)	Correct rate (%)		
	1 byte	2 bytes	Cavnar
Shift_JIS (Japanese)	99.8	100.0	100.0
EUC-JP (Japanese)	99.2	100.0	100.0
GB2312 (Chinese)	100.0	100.0	100.0
Big5 (Chinese)	100.0	100.0	100.0
EUC-KR (Korean)	100.0	100.0	100.0
ISO-8859-1 (English)	90.9	98.9	99.0
ISO-8859-1 (German)	99.4	100.0	99.8
ISO-8859-1 (French)	95.1	99.9	99.4
ISO-8859-1 (Italian)	98.2	100.0	99.8
ISO-8859-1 (Spanish)	90.7	99.9	99.8
ISO-8859-1 (Portuguese)	92.8	100.0	100.0
ISO-8859-1 (Danish)	69.8	92.6	91.4
ISO-8859-1 (Norwegian)	70.1	91.5	88.6
ISO-8859-1 (Swedish)	94.9	99.7	99.4
Avg.	92.9	98.7	98.4

Correct rates of identification using the vector-distance method are shown in Table 4.4.

The confusion matrixes for the results of one-byte and two-bytes vector-distance methods are shown in Table 4.5 and Table 4.6, respectively. In the tables, the rows correspond to the identification results, and the columns correspond to the correct answers.

4.6.3 Comparing with Cavnar et al.'s Method

In addition to the proposed methods, Cavnar et al.'s method described in Section 4.2.1 was experimented using the same test data. The GNU dbm (gdbm) library[31] was used for counting *n*-grams, and qsort() built-in function was used for sorting *n*-grams.

When applying this method, the number of the top most *n*-grams to be used for

Table 4.5. Confusion matrix of the identification results of the one-byte vector-distance method.

	SJ	**EJ**	**GB**	**B5**	**EK**	**en**	**de**	**fr**	**it**	**es**	**pt**	**da**	**no**	**sv**
SJ	998													
EJ		992	5	3										
GB			767											
B5				812										
EK					1000									
en						909	8	38	5	15	17	5		3
de						3	994	1				2		
fr						21	3	951	4	13	8			
it						8		3	982	7				
es						6		11	7	907	69			
pt						5		7	4	30	593			
da						8	5	1				506	183	22
no							3		1			131	378	26
sv						8	3	1			1	15	12	743

CJK: SJ=Shift_JIS, EJ=EUC-JP, GB=GB2312, B5=Big5, EK=EUC-KR
ISO-8859-1: en=English, de=German, fr=French, it=Italian, es=Spanish, pt=Portuguese, da=Danish, no=Norwegian, sv=Swedish

Table 4.6. Confusion matrix of the identification results of the two-bytes vector-distance method.

	SJ	**EJ**	**GB**	**B5**	**EK**	**en**	**de**	**fr**	**it**	**es**	**pt**	**da**	**no**	**sv**
SJ	1000													
EJ		1000												
GB			767											
B5				812										
EK					1000									
en						989				5	2	3		1
de							1000							
fr						1		999						
it									1000					
es										999	1			
pt											639			
da						1	1					671	7	
no												45	493	1
sv												1	1	781

CJK: SJ=Shift_JIS, EJ=EUC-JP, GB=GB2312, B5=Big5, EK=EUC-KR
ISO-8859-1: en=English, de=German, fr=French, it=Italian, es=Spanish, pt=Portuguese, da=Danish, no=Norwegian, sv=Swedish

calculating the similarity affects the performance. In this experiment, we used top 400 *n*-grams, which achieved the best performance in their experiments[14].

The execution time of Cavnar et al.'s method was about 1.4 seconds per document. Correct rate of identification using the Cavnar et al.'s method are shown in the column "Cavnar" in Table 4.4.

The average correct rate of Cavnar et al.'s method was slightly lower than that of two bytes vector-distance method. In regard to the efficiency, Cavnar et al.'s method requires to prepare a data structure such as a hash table in order to store the occurrence frequencies of *n*-grams, whereas vector-distance method only requires to prepare an array to store the occurrence frequencies of character codes. In addition, Cavnar et al.'s method requires to sort *n*-grams in the order of occurrence frequency. Furthermore, for the calculation of distance, Cavnar et al.'s method requires to look up the rank in the training data for each *n*-gram in the target document.

The average number of distinct *n*-grams per document was 1,443 for target documents used in experiments. However, the actual length of Web documents is about 7 kilobytes on average according to a recent statistics[66]. It is much longer than the average length of documents used in the experiments which was about 1 kilobytes. Therefore, it may take longer time for counting and sorting *n*-grams for the actual Web documents. In the experiment, the execution time of Cavnar et al.'s method was about 14 times longer than that of two bytes vector-distance method. Although the performance of Cavnar et al.'s method may possibly be improved to some extent by optimization, the proposed method may still be more efficient. Consequently, the proposed method has advantages over Cavnar et al.'s method in terms of both effectiveness and efficiency.

4.6.4 Discussion of Identification Errors

From the table 4.4, one byte vector-distance method achieved 92.9% on average, but only achieved about 70% for Danish and Norwegian of ISO-8859-1. Mostly those two languages were mis-identified vice versa; about 25% of Danish were mis-identified as Norwegian, and about 24% of Norwegian were mis-identified as Danish. It is probably because the languages of Danish, Norwegian, and Swedish are very similar in terms of linguistics, and it might be the cause of the mis-identification between Danish and Norwegian. On the other hand, Swedish gained much better results than those two

languages, probably due to the difference in the usage of accented letters.

For two bytes vector-distance method, correct rates are improved for almost all of the languages, and achieved 98.7% correct rate in average. Even for Danish and Norwegian, in which the performances were poor for one byte vector-distance method, the correct rates are improved over 20% for both languages.

4.6.5 Combining Parameter and Vector-Distance Methods

From the analysis described above, it may be possible to identify both 7-bit and 8-bit coding systems effectively and efficiently, by combining the parameter method and the two bytes vector-distance method. The procedure of this method is as follows:

1. First, the parameter method is applied for the target document, and if the result was not "Unknown" (in the algorithm in Figure 4.2), the identified coding system and language is selected as the result,

2. For documents that could not be identified by the parameter method (those identified as "Unknown"), the two bytes vector-distance method is applied, and the identified coding system and language is selected as the result.

The execution time of this method was about 0.09 seconds per document. Correct rates of identification using this method are shown in Table 4.7.

Considering the possibility of misidentification in the parameter method, correct rate might be improved by applying the parameter and vector-distance methods in parallel and comparing both results. However, because the parameter method achieved 100% correct rate in this experiment, and it can identify 50 times faster for supported coding systems, we used a sequential combination of two methods as described above.

4.6.6 Text Length vs. Correct Rate

We investigated the minimum text length by which the language and the coding system can be correctly identified in our method. Note that identification of 7-bit coding systems such as ASCII and ISO-2022 variants does not depend on the text length, so those were not investigated here.

Table 4.8 shows the correct rates changing the text length n used for identification from 50 to 1,000 bytes. In the table, "CJK" indicates the average correct rates for five

Table 4.7. Correct rate of identification using combination of the parameter and the vector-distance methods.

Coding system (language)	**Correct rate (%)**
ISO-2022-JP (Japanese)	100.0
ISO-2022-CN (Chinese)	100.0
ISO-2022-KR (Korean)	100.0
Shift_JIS (Japanese)	100.0
EUC-JP (Japanese)	100.0
GB2312 (Chinese)	100.0
Big5 (Chinese)	100.0
EUC-KR (Korean)	100.0
ISO-8859-1 (English)	98.9
ISO-8859-1 (German)	100.0
ISO-8859-1 (French)	99.9
ISO-8859-1 (Italian)	100.0
ISO-8859-1 (Spanish)	99.9
ISO-8859-1 (Portuguese)	100.0
ISO-8859-1 (Danish)	92.6
ISO-8859-1 (Norwegian)	91.5
ISO-8859-1 (Swedish)	99.7
Avg.	99.0

Table 4.8. Correct rate at different text length for identification.

# bytes	Correct rate (%)		
	CJK	8859-1	avg.
50	98.0	76.3	84.0
100	99.8	90.0	93.5
200	100.0	95.6	97.2
300	100.0	97.0	98.1
400	100.0	97.8	98.6
500	100.0	98.0	98.7
600	100.0	98.2	98.8
700	100.0	98.3	98.9
800	100.0	98.5	99.0
900	100.0	98.5	99.0
1,000	100.0	98.8	99.3

coding systems (Shift_JIS, EUC-JP, GB2312, Big5, and EUC-KR) of Japanese, Chinese, and Korean, and "8859-1" indicates the average correct rates for each language in ISO-8859-1 coding system.

Figure 4.5 shows the transition of correct rate at different text length for identification. The horizontal axis is the text length n used for identification, and the vertical axis is the correct rate of identification for that length.

From the result of this experiment, we can observe that about 100 bytes are almost enough for five coding systems of Asian languages, but for European languages included in ISO-8859-1, at least 500 bytes are required in order to achieve 98% correct rate. Note that 99.8% average correct rate was achieved by excluding Danish and Norwegian, which are prone to be mis-identified.

4.7 Summary

In this chapter, we proposed a automatic identification method of coding systems and languages of multilingual Web documents which employs the statistics and its analysis.

In order to verify the effectiveness of the proposed method, we conducted exper-

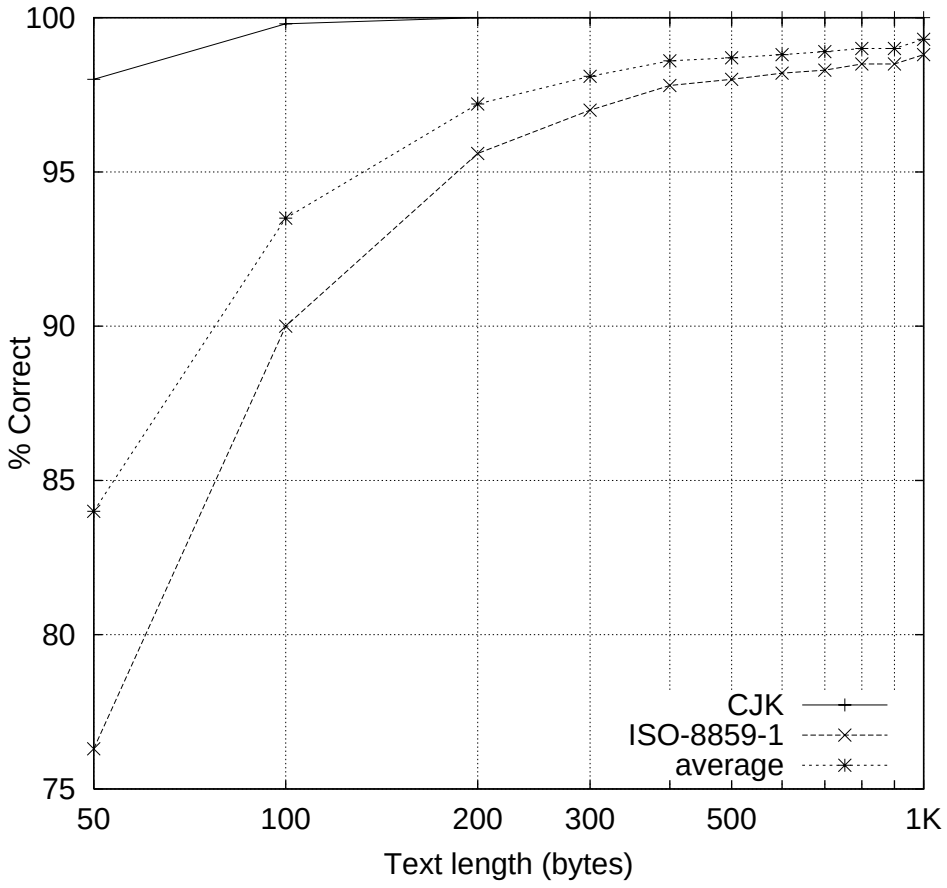


Figure 4.5. Transition of correct rate at different text length for identification.

iments for 12 languages and 10 coding systems, and achieved over 98% correct rate in average. Especially for Asian languages that use multiple bytes to represent one character, we showed that it is possible to identify those languages and coding systems effectively and efficiently, simply using one byte unit and without discriminating the boundaries of characters. For European languages, we showed that it can achieve 98% average correct rate for 9 languages included in ISO-8859-1, by using the consecutive two bytes unit, which takes the connection of consecutive characters into account. Furthermore, we investigated the relation between text length and correct rate, and showed that our method can identify nearly perfect by using only first 300 bytes of text, excluding Danish and Norwegian.

Also, our method has an advantage that it can easily be extended to other languages and coding systems other than the target languages used in this chapter, because it does not depend on the byte length of one character, which varies depending on the coding system.

One of the future work for this research is an extension to other languages and coding systems. For example, several different coding systems are used to represent Web documents in Arabic and Russian, so automatic identification technique for these coding systems is desired. Also, the language identification of documents in which multiple languages are used, which is expected to be increase in proportion to the increasing popularity of Unicode, has to be investigated in the future.

Another possible future work might be an exploration into the automatic creation of discrimination rules for the parameter method. The parameter method has an advantage that it is much more efficient than other methods, but has a disadvantage that the rule has to be manually created by observing the code distribution and finding a significant characteristic for a particular coding system. Therefore, a method to automatically create such rules might be a promising future work.

Chapter 5

Query Term Disambiguation for Web Cross-Language Information Retrieval

5.1 Introduction

With the increasing popularity of the Internet in various parts of the world, languages used for Web documents are expanded from English to others. Some Web search engines such as AltaVista and Lycos can handle multiple languages in addition to English and can specify the target language of the documents to be retrieved. At the same time, many search engines exist that handle Web documents written in a particular language other than English. However, these search engines are essentially a collection of monolingual search engines from the user's perspective. Nevertheless, there might be some cases where the user wishes to retrieve documents in unfamiliar languages. Needs for retrieving such information must be large. For example, when Japanese is used as a query language, target collection will represent only a very small portion of the whole Web documents, which consist of several hundreds millions of pages. Also, there might be cases, depending on the user's demand, where information written in a language other than the user's native language is rich. For example, for the economic trend of a particular country, there should be extremely rich information in the language related to that country.

To fulfill such needs, researches on Cross-Language Information Retrieval (CLIR), a technique to retrieve documents written in a certain language using a query written in another language, have been active in recent years. CLIR is also referred to as cross-

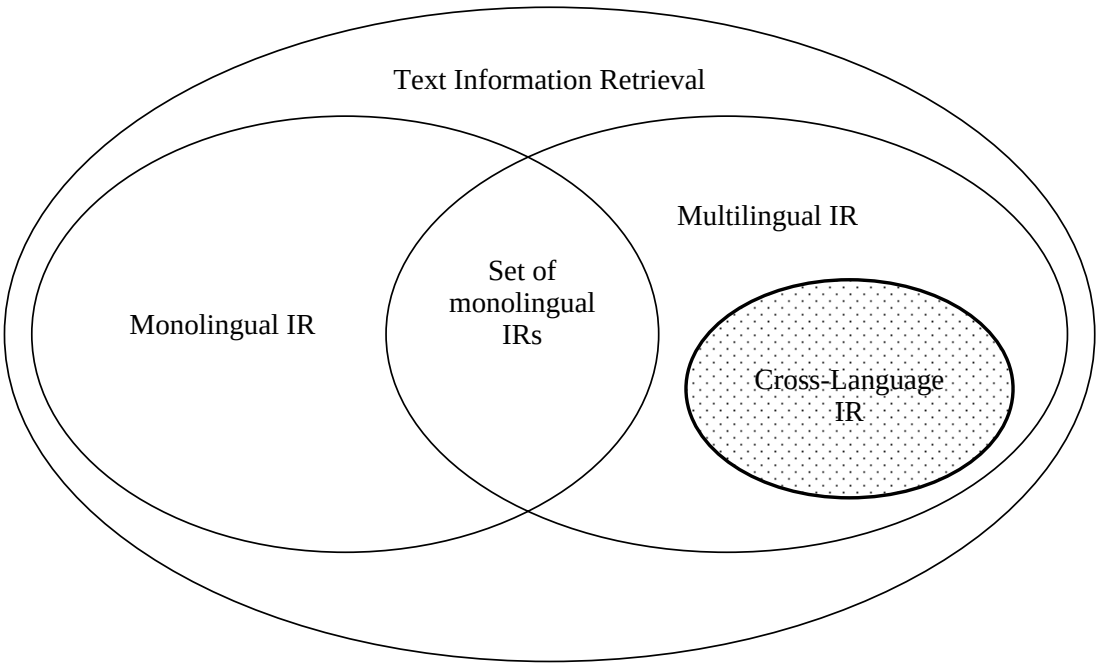


Figure 5.1. Scope of multilingual and cross-language information retrieval in the context of text information retrieval.

lingual IR, cross-linguistic IR, or translingual IR. Although the term *multilingual IR* is often used to indicate CLIR, it is a broader term and includes a system that handles multiple languages but does not cross languages (e.g. AltaVista and Lycos), as shown in Figure 5.1.

An obvious solution to CLIR is to translate all Web documents into the query language in advance. However, considering the enormous amount of Web documents, this approach is unrealistic. Therefore, it is feasible to translate the retrieved documents into the specified language. Recently, relatively inexpensive machine translation softwares are becoming more available. Some of them can translate and display a Web document on a Web browser on-the-fly. Therefore, some users cannot use languages other than their own native language in this case, increasing solutions in CLIR can be considered as useful.

Therefore, there is a problem in the query formulation in the target language. In order to satisfy such needs on usual monolingual retrieval system, the user has to man-

ually translate the query by using a dictionary. This process not only imposes a burden to the user but also might choose incorrect translations for the query, especially for languages that are unfamiliar to the user. Accordingly, the approach in which the user formulates a query in his/her native language and the system translates it, should be desirable. One of the major technical problems to be solved in CLIR concerns the translation of short queries of one or few words, appropriately. Possible translation candidates might be numerous in such cases and resolving such ambiguities becomes a hard task.

In this chapter, we propose a novel approach for CLIR system targeting Web documents, which uses a natural language resource that is extracted from a Web search engine as a corpus, and resolves the ambiguities caused by the dictionary-based query translation approach, by using a co-occurrence information[76, 71]. We have evaluated the effectiveness of this method by experiments. By using this method, we do not have to worry about obtaining expensive language resources, which is one of the inconveniences of existing CLIR approaches. Easy extension to other languages, as well as the achievement of a reasonable retrieval effectiveness, are among the advantages of our approach. We also conducted a comparative evaluation of four co-occurrence measures; mutual information, modified Dice coefficient, log likelihood ratio, and χ^2 test.

5.2 Related Work

Approaches to CLIR can be classified into three categories; document translation, query translation, and the use of an inter-lingual representation that is language independent.

The approach based on translation of target documents has the advantage of utilizing existing machine translation systems, in which more context information can be used for disambiguation. Thus, in general, it achieves a better retrieval effectiveness than those based on query translation. However, since it is impractical to translate a huge document collection beforehand and it is difficult to extend this method to new languages, this approach is not suitable for multilingual, large-scale, and frequently-updated collection of WWW.

The second approach transfers both documents and queries into inter-lingual rep-

resentation, such as bilingual thesaurus classes[97] or a language-independent vector space[23, 104]. The latter approach requires a training phase using a bilingual (parallel or comparable) corpus as a training data.

The major problem in the approach based on the translation and the disambiguation of queries is that the queries submitted from ordinary users of Web search engines tend to be very short (approximately two words on average[51]) and usually consist of just an enumeration of keywords (i.e. no context). However, this approach has an advantage that the translated queries can simply be fed into existing monolingual search engines. In this approach, a source language query is first translated into the target language using a bilingual dictionary, and then the translated query is disambiguated. Our method falls into this category. One of the crucial problems of dictionary-based translation is the lack of headwords (e.g. compound words, coined words, loan words, etc.). Fujii et al.[29] proposes the methods for the translation of compound words and transliteration of phonograms (i.e. katakana in Japanese) using bigram statistics.

For the disambiguation method, various approaches have been proposed[84, 34], such as using the first term listed in the dictionary[85], using relevance feedback[6, 7], and using a parallel corpus[22] or a comparable corpus[103]. However, such bilingual corpora are usually not readily available. Nie et al.[81] proposes a method to automatically gather parallel texts from the Web and use them for query term selection. However, for language pairs other than English-French in their case, the amount of parallel documents on the Web might not always be enough. Therefore, query term disambiguation method that does not depend on expensive language resources such as a parallel corpus, is of practical use.

One of the practical approaches for disambiguation is the use of the target language monolingual corpus. This approach has been adopted in word sense disambiguation task in the field of machine translation. Dagan and Itai[18] use the combination of syntactic rules and statistics on the target language corpus for target word selection in machine translation. Their method is not directly applicable to CLIR application because of the lack of syntax in queries submitted from ordinary users, as mentioned above. Therefore, only statistics on the target language corpus can be used for the disambiguation task of CLIR. For such a type of methods, *mutual information*, which is calculated from co-occurrence frequency of terms in a monolingual corpus, has been employed in several researches[75, 67, 50]. These approaches are all based on the hy-

pothesis that correct translations of query terms should co-occur in the target language corpus, and incorrect translations should not co-occur.

Lin et al.[67] uses mutual information for the disambiguation of translated queries in Japanese-English CLIR task. However, their method selects the only translation-candidate that has the highest score for actual retrieval. In this case, there is a possibility of selecting inappropriate translation. Jang et al.[50] uses mutual information for both disambiguation and query term weighting in Korean-English CLIR task. In their experiments, the method achieved 85% of monolingual retrieval case.

According to Lin et al. and Jang et al. experiments, the window size of the co-occurrence was set to 3 and 6 words, respectively. In proportion the narrower the window size, the less effect of unrelated terms. However, term pairs that do not co-occur will increase even if the corpus is relatively large. Furthermore, mutual information has an undesired characteristic, which is the assignment of unexpectedly high values to rarely occurred terms[7].

Although our method[75, 26] also uses mutual information, in order to avoid the possibility of selecting only inappropriate translations, we take all translation candidates that exceed a certain threshold value. In addition, in order to avoid the undesired effect of rarely occurred terms, we set another threshold value for the minimum occurrence of a term in a corpus. Moreover, by utilizing a Web search engine, we make it possible to use it as a huge corpus of various domains for the disambiguation without collecting enormous amount of Web documents.

5.3 Query Translation

Figure 5.2 shows the flow of query translation for the proposed query term disambiguation method.

A query in user's native language is first translated into the target language using a bilingual dictionary. The obtained translation candidates are disambiguated using term co-occurrence statistics and then passed to the search engine.

A query submitted by a user is first segmented into words using a morphological analyzer. Then, each word is translated into the target language using a machine-readable dictionary. In this phase, the longest matched term in the dictionary is used as the translation term. For example, an English query "digital library" can be segmented

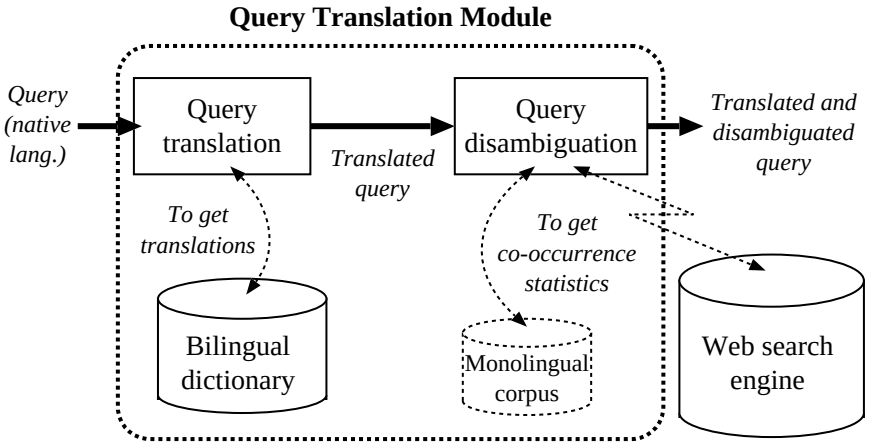


Figure 5.2. Flow of Query Translation.

into “digital” and “library”, but if the phrase “digital library” was found in the dictionary, the translation(s) of that phrase will be used instead. For the case of the longest match overlaps, (e.g. “distributed network environment” matches both “distributed network” and “network environment”), both phrases will be used.

Translation candidates obtained from the dictionary are then disambiguated using the method described in the next section, which is based on 2 or *n* words co-occurrence frequency information obtained from the target language corpus.

5.4 Query Term Disambiguation using a Web Search Engine

Since Web search engines gather an enormous volume of documents that cover extensive domains, they might be very useful as natural language resources. For instance, the number of retrieved documents by searching some terms combined by AND operators, can be regarded as a co-occurrence frequency of those terms in a Web document corpus. Ikeno et al.[41] investigates the possibility to apply it for selecting appropriate translated words for machine translation. We apply this method to the query disambiguation in CLIR.

Table 5.1. 2-by-2 table for calculating two words co-occurrence tendency.

	w_2	$\neg w_2$
w_1	n_{11}	n_{12}
$\neg w_1$	n_{21}	n_{22}

5.4.1 Different Measures of Co-occurrence

Here we define several different measures of co-occurrence (we call them *co-occurrence tendency*). First, for two words w_1 and w_2 to calculate co-occurrence tendency, we define $n_{ij}(i, j = 1,2)$ in a 2-by-2 table shown in Table 5.1. In the table, n_{11} indicates the number of times two words w_1 and w_2 co-occur within a text window, n_{12} indicates the number of times w_1 occurs, but w_2 does not occur within a text window, and so on.

$n_{i.}$, $n_{.j}$, and N are defined as follows:

$$\begin{aligned} n_{i.} &= n_{i1} + n_{i2} \\ n_{.j} &= n_{1j} + n_{2j} \\ N &= \sum_{i,j} n_{ij} \end{aligned}$$

That is, $n_{i.}$ indicates the number of times w_1 occurs ($i = 1$) or does not occur ($i = 2$) regardless of the occurrence of w_2, and N indicates the total number of co-occurrence windows in the corpus.

Generally, the window size of co-occurrence is a fixed number of words, but mainly for the limitation of utilizing search engines, we use one document as the window of co-occurrence. Actually, for example $w_1 \wedge \neg w_2$ (n_{12} in Table 5.1) can be obtained by submitting a query "w_1 AND NOT w_2" (in syntax of AltaVista). In this case, N is the total number of documents in the search engine. For example, for AltaVista, the total number of documents can be obtained by submitting a query of "*" symbol that matches arbitrary strings.

We consider four co-occurrence measures; mutual information, modified Dice coefficient, log likelihood ratio, and χ^2 test. Each measure will be described in the following sections.

Mutual Information

Mutual Information (MI) is one of the metrics that can be used for calculating the significance of word co-occurrence associations[16]. The mutual information MI between two events x, y is defined as follows:

$$MI(x,y) = \log_2 \frac{p(x,y)}{p(x)p(y)} \tag{5.1}$$

where $p(x,y)$ is the joint probability that two events x and y co-occur, and $p(x)$ and $p(y)$ are the probabilities that the event x or y occur independently.

MI can be applied to the words in the documents and can be used to calculate the correlation between those words. Two words co-occurrence tendency COT_{MI2} between words w_1 and w_2 in a corpus is defined as follows:

$$COT_{MI2}(w_1, w_2) = \log_2 \frac{\frac{n_{11}}{N}}{\frac{n_{\cdot 1}}{N}\frac{n_{1\cdot}}{N}} \tag{5.2}$$

Note that we use one document as the window of co-occurrence instead of fixed number of words.

Usually, co-occurrences are measured between two words mainly because of computational and the storage cost. However, when using a Web search engine, it is not necessary to calculate the frequencies for every pair of words in advance. We can use co-occurrence frequencies among any n words.

n words co-occurrence tendency COT_{MIn} among words $w_1, w_2 \ldots w_n$ is defined, as an extension of COT_{MI2}, as follows:

$$COT_{MIn}(w_1, w_2 \ldots w_n) = \frac{1}{n-1} \log_2 \frac{\frac{f(w_1, w_2 \ldots w_n)}{N}}{\frac{f(w_1)}{N}\frac{f(w_2)}{N}\cdots\frac{f(w_n)}{N}} \tag{5.3}$$

where N is the total number of documents in the search engine, $f(w)$ is the number of retrieved documents for the word w, and $f(w_1, w_2, \ldots w_n)$ is the number of retrieved documents for the words $w_1, w_2, \ldots w_n$ combined using AND operators. Note that MI is essentially a measure between two events, so this is an *ad hoc* extension only for the purpose of calculating n words co-occurrence tendency.

Table 5.2. An example of two words co-occurrence tendency values of Japanese word pairs.

w_1	w_2	$COT_2(w_1, w_2)$
情報 (information)	検索 (retrieval)	1.46
情報 (information)	ニュース (news)	1.41
情報 (information)	図書館 (library)	1.23
情報 (information)	負傷 (injury)	0.77
情報 (information)	団子 (dumpling)	0.25
情報 (information)	陶磁器 (pottery)	0.16

Table 5.3. An example of two words co-occurrence tendency values of English word pairs.

w_1	w_2	$COT_{MI2}(w_1, w_2)$
database	multimedia	3.37
database	transaction	2.41
database	relational	2.01
database	chair	-2.96
database	soul	-3.43
database	iron	-4.62

Table 5.3 gives some examples of co-occurrence tendency values for English words extracted from randomly sampled 14 thousand English Web documents.

Modified Dice Coefficient

Dice coefficient is rather a heuristic measure and it is also commonly used for calculating the word co-occurrence tendency[101]. The Dice coefficient between two words w_1 and w_2 is defined as follows:

$$Dice(w_1, w_2) = \frac{2n_{11}}{n_{1\cdot} + n_{\cdot 1}} \tag{5.4}$$

Kitamura et al.[62] proposes a modification of Dice coefficient, which improves the accuracy of co-occurrence tendency by adding a weight based on co-occurrence frequency. In this chapter, we call it *modified Dice coefficient*. Two words co-occurrence tendency COT_{DICE} between words w_1 and w_2 in a corpus, is defined as follows:

$$COT_{DICE}(w_1, w_2) = \log_2 n_{11} \frac{2n_{11}}{n_{1\cdot} + n_{\cdot 1}} \tag{5.5}$$

Log Likelihood Ratio

Likelihood ratio test is a kind of hypothesis testing. It can also be used for calculating *COT*, and it is said to be more accurate than MI, especially for rare occurred terms[24]. Two words co-occurrence tendency COT_{LLR} between words w_1 and w_2 in a corpus, is defined as follows:

$$COT_{LLR}(w_1, w_2) = 2 \sum_{i,j} n_{ij} \left(\log_2 \frac{n_{ij}}{N} - \log_2 \frac{n_{i\cdot} n_{\cdot j}}{N} \right) + \Delta \tag{5.6}$$

Although the value of the correction term Δ does not affect the actual selection of translations, we use $\Delta = 2$ on the basis of AIC (Akaike Information Criterion).

Chi-Square Test

χ^2 test is also a hypothesis testing and is used for testing the dependence of two variables whose probabilities are approximately χ^2 distributed. Since χ^2 is a continuous distribution, some correction is needed for small counts. Yate's correction is applied if any of n_{ij} is smaller than 5. Two words co-occurrence tendency COT_{CHI} between words w_1 and w_2 in a corpus, is defined as follows:

$$COT_{CHI}(w_1, w_2) = \begin{cases} \dfrac{N(|n_{11}n_{22} - n_{12}n_{21}| - \frac{N}{2})^2}{n_{1\cdot}n_{2\cdot}n_{\cdot 1}n_{\cdot 2}} & \text{if} \min(n_{ij}) < 5, \\ \dfrac{N(n_{11}n_{22} - n_{12}n_{21})^2}{n_{1\cdot}n_{2\cdot}n_{\cdot 1}n_{\cdot 2}} & \text{otherwise} \end{cases} \tag{5.7}$$

Note that log likelihood ratio and χ^2 test cannot easily be extended into *n* words, because the frequencies of non-occurrence (e.g. $\neg w_1$ etc.) have to be calculated.

Table 5.4. Comparison of various measures for co-occurrence tendency of Japanese words “大気 (air)” and “汚染 (pollution)”.

	COT_{MI2}	COT_{DICE}
1	consideration contamination	**air pollution**
2	consideration pollution	consideration pollution
3	**air pollution**	atmosphere pollution
4	atmosphere pollution	consideration contamination
5	air contamination	air contamination
6	atmosphere contamination	atmosphere contamination
7	generosity pollution	generosity pollution
8	generosity contamination	generosity contamination
	COT_{LLR}	COT_{CHI}
1	**air pollution**	**air pollution**
2	consideration pollution	consideration pollution
3	air contamination	air contamination
4	consideration contamination	consideration contamination
5	atmosphere pollution	atmosphere pollution
6	atmosphere contamination	atmosphere contamination
7	generosity pollution	generosity pollution
8	generosity contamination	generosity contamination

5.4.2 Comparing COT Measures

Table 5.4 shows the comparison of various measures for the co-occurrence tendency of Japanese words “大気 (air)” and “汚染 (pollution)”. In this example, the pair “air” and “pollution” is the most appropriate translation. It is ranked top for COT_{DICE}, COT_{LLR}, COT_{CHI}, but third for COT_{MI2}. In [77] and some other research on co-occurrence measures[24, 40], it is claimed that mutual information is less accurate than other co-occurrence measures. This example might support those claims.

bank

money

trade

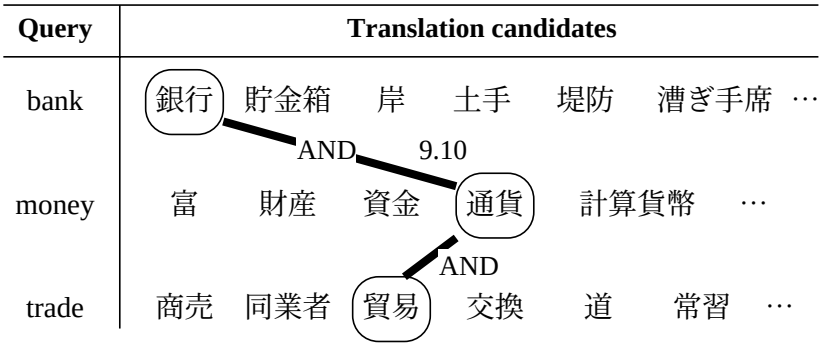


Figure 5.4. An example of a disambiguation using *n* words *COT* (COT_{MIn}).

We eliminate the terms which rarely occur, in order to avoid the undesired phenomenon of MI described before. For example, in Figure 5.3, if we do not eliminate rarely occurred terms, unrelated term sets "漕ぎ手席 (seat of an oarsman or a rower)", "富 (wealth or fortune)" and "常習 (habitual for customary)" is mistakenly given the highest average COT_{MI2} score.

On the other hand, when using *n* words co-occurrence tendency, the terms actually used for the target language query are determined as follows:

1. Obtain the number of retrieved documents for each term in the query from the Web search engine,
2. Obtain numbers of retrieved documents for all possible combinations of translation candidates whose occurrence frequency for each term exceed the threshold value T_{freq}, from the Web search engine (using AND operators),
3. The term sets whose *COT* exceed the threshold value T_{COT} are selected as the target language query.

Figure 5.4 shows an example of a disambiguation using COT_{MIn} for the same query as the previous example. Also in this case, the same term set, "銀行", "通貨", and "貿易" is given the highest COT_{MIn}. All term sets which exceed T_{COT} are combined with OR operators, and are used as a Japanese query.

This method may cost much time for querying the search engine, especially for queries with many possible translation-candidate pairs. However, it can be greatly reduced by submitting multiple requests for a query in parallel.

We can consider using a proximity operator instead of an AND operator. The proximity operator matches documents containing specified terms within a specific window of words, regarding or regardless of order. For example, AltaVista supports the proximity operator called "NEAR" which retrieves specified terms within 10 words regardless of order. In this case, to be exact, N is the total number of object windows, which cannot be calculated exactly using a search engine. However, since it does not affect the ranking of translation candidates, and since the absolute value is not important in this case, we used the total number of documents for N in our experiments.

5.5 Evaluation

We have conducted experiments to evaluate the effectiveness of the Japanese-English CLIR using the proposed query translation method.

The threshold value T_{COT} was set as, $\max(COT) - 4.0$, $\max(COT) \times 0.7$, $\max(COT) \times 0.9$, $\max(COT) \times 0.8$ for COT_{MIn}, COT_{LLR}, COT_{DICE}, and COT_{CHI}, respectively ($\max(COT)$ is the maximum COT for a query). The threshold value for word occurrence frequency T_{freq} was set to $1/10,000$ of the total number of English documents in the search engine, hence $N/10,000$. Those values were empirically determined based on the best results in the preliminary experiments. Specifically, we tested $\max(COT) - 0.0 \ldots 6.0$ for COT_{MIn}, and $\max(COT) \times 0.1 \ldots 0.9$ for COT_{LLR}, COT_{DICE}, and COT_{CHI}.

5.5.1 Test Data

For the test data, we used NACSIS Test Collection 1 (NTCIR-1)[57] (Research Purpose Use). It contains about 330,000 technical documents, which are summaries of papers presented at conferences hosted by 65 Japanese academic societies, and we used E-Collection, which contains about 190,000 English summaries. Although our system is essentially targeting Web documents, we used NTCIR collection because there is no other test collection suitable for evaluating Japanese-English CLIR of Web documents at present. In fact, relevance judgment of Web documents is a difficult task because of their diversity in format, domain, length, language, etc. Furthermore, it is also very difficult to assess recall in a large collection of Web documents, since it is totally unaffordable to judge all documents against a large number of topics[37]. However,

recall is less important in Web IR than in traditional IR, because people usually do not want to retrieve all relevant documents, which will sometimes be a very large number.

We used 39 Japanese search topics for evaluation. The TITLE fields of 39 topics in NTCIR-1 and their English translations are shown in Table 5.5.

We used only TITLE field, which is a very short description of the topic, for queries. It contains 1-7 words (2.7 words on average) and it resembles to the queries often submitted by an end-user of Web search engines in terms of length.

TITLE fields are segmented into words using ChaSen[78] Japanese morphological analyzer, and only nouns and unknown terms were used as a query.

5.5.2 Language Resources used for the Experiments

Bilingual Dictionary

For query translation, we merged three dictionaries, *Japanese-English Bilingual Dictionary* and *Technical Terms Dictionary (Information Processing)* of EDR Electronic Dictionary Version 1.5[52], and EDICT[1], which is a freeware Japanese-English dictionary. The total vocabulary of the merged dictionaries was 366,041. However, it was not sufficient for translating some of the queries used for the experiments. In order to avoid the effect of the quality of the dictionary, we added translations for 18 words that appeared in the queries. For example, we added “extraction” for one of the translations of “抽出”, and “collision” for “干渉”, etc. The reason for adding translations is simply to avoid the loss in accuracy of the experimental results. If we do not add the missing translations, some queries will not retrieve any documents, thus dropping the accuracy and the reliability of the experimental results. Additionally, the absolute performance is not important in this experiment since there is no comparable experimental result that used only TITLE field in NTCIR experiments.

Note that we used only translation candidates that consist of one term or one phrase and did not use translation candidates that consist of descriptional sentences, because they are not suitable for the proposed disambiguation method.

[1] http://www.csse.monash.edu.au/~jwb/edict.html

Table 5.5. TITLE fields in NTCIR-1 topics and their English translations.

Topic ID	TITLE field	English translation
0031	データ 品質 制御	data quality control
0032	ネットワーク コラボレーション	network collaboration
0033	メディア 同期	media synchronization
0035	電子 図書館	electronic library
0036	グループ ウェア	groupware
0037	バッファ 制御	buffer management
0038	TCP/IP 通信 スループット	TCP/IP communication throughput
0039	WWW トラフィック	WWW traffic
0040	高精細 画像 生成	high definition image generation
0041	超 高精細 画像 医療	super high definition image medical
0042	3 次元 動画像 通信	three dimension motion image communication
0043	動画像 圧縮 センサ	motion image compression sensor
0044	画像 電子 透かし	image digital watermark
0047	マルチキャスト 実装	multicast implementation
0048	ホーム エリア ネットワーク	home area network
0049	ネットワーク トポロジー	network topology
0050	人工知能 将棋	artificial intelligence shogi
0051	次世代 インターネット	next generation Internet
0053	電波 人体 影響	radio wave human body effect
0054	光ファイバ 通信速度	optical fiber communication throughput
0055	構造化 文書 検索 全文 データベース 索引	structured document retrieval fulltext database index
0056	人工物 ライフサイクル 情報 知識 共有	artificial life cycle information knowledge sharing
0057	創造的 思考 モデル化 支援	creative thinking modeling support
0058	Zipf 法則	Zipf law
0059	シソーラス 自動 構築	thesaurus automatic construction
0061	階層関係 自動 抽出	hierarchical relationship automatic extraction
0062	生涯学習 ボランティア	lifelong learning volunteer
0063	多国語 文字	multilingual character
0064	生涯学習 施設 学校	lifelong learning facility school
0066	大学 図書館 建築 空間	university library architecture space
0067	自然災害 学校 施設	natural disaster school facility
0068	デジタル 著作物	digital writing
0070	高齢者 行動 特性	aged person behavior characteristic
0071	著作権 課金 体系	copyright charge system
0072	博物館 資料	museum material
0076	多面体 干渉 検出	polyhedron collision detection
0077	点字 翻訳	Braille translation
0080	神経 再生	nerve regeneration
0083	骨 形成 分子 メカニズム	bone formulation molecule mechanism

Monolingual Corpus

We used a Web search engine as a monolingual corpus, as described in Section 5.4. We chose AltaVista as the Web search engine for the query disambiguation for the following reasons:

1. It is possible to specify the target language for the retrieval,
2. It is possible to obtain the total number of documents, which is required to calculate *COT*,
3. It supports the proximity operator (NEAR) in addition to the AND operator,
4. It has comparatively large index as a Web search engine (about 130 million English documents on March 2000).

Generally, Web search engines periodically update their index, and the populations of the target collection may fluctuate as time goes on. In the experiments, in order to avoid the effect of such fluctuation, co-occurrence frequencies for a query were obtained continuously within a short period of time and those values were never reused. The number of times querying the search engine for 39 queries was 107.5 on average, but 33.7 on average for queries containing no more than 3 terms (33 queries out of 39).

Furthermore, we also experimented with NTCIR-1 E-Collection, which is identical to the target collection, as the monolingual corpus. In this case, the AND operator was used for calculating co-occurrence tendencies.

5.5.3 Retrieval System

For the retrieval system, we used *Namazu* retrieval system[2] (version 2.0.1). It is a freeware full-text retrieval system based on a Boolean model and supports basic functions such as a composition of Boolean operators, ranking of the results, and phrase retrieval.

5.5.4 Evaluation Measures

For the evaluation of the retrieval effectiveness, we used standard measures of information retrieval, i.e. *recall* and *precision*[92].

[2]http://www.namazu.org/

Consider that a set R is all relevant documents for a given query, and $|R|$ is the number of documents in this set. Assume that the retrieval result of a given IR system (an answer set) for the query is set A, and $|A|$ is the number of documents in this set. Let $|Ra|$ be the number of documents in the intersection of the sets R and A. Using these variables, the recall and precision are defined as follows:

Recall is the fraction of the relevant documents R which has been retrieved, thus,

$$Recall = \frac{|Ra|}{|R|} \tag{5.8}$$

Precision is the fraction of the retrieved documents A which is relevant, thus,

$$Precision = \frac{|Ra|}{|A|} \tag{5.9}$$

In addition, we used a single value measure called *non-interpolated average precision*. It is the average of the precisions of all relevant documents for a query, and shows the balance of recall and precision.

5.5.5 Experiment 1 (based on MI)

Experimental Results

Experimental results for n term co-occurrence tendency based on MI are shown in Table 5.6 and Figure 5.5.

In the table, NODIS is the result of no disambiguation, which means using all possible translation candidates obtained from the dictionary. In this case, all translation candidates of each term were combined with AND operators, and all translation candidates were combined with OR operators.

NOSTR is the result of using all possible translation candidates without structuring (i.e. all translation-candidate terms were simply combined with OR operators).

ONE is the result of using only one translation-candidate that has the top COT_{MI2}, by using NTCIR collection as a corpus.

NOTHR is the result of setting the threshold value for word occurrence frequency T_{freq} to be zero, by using NTCIR collection as a corpus.

AND and NEAR are the results of using the proposed disambiguation method described in section 5.4, which uses the Web search engine as a corpus. If there is no

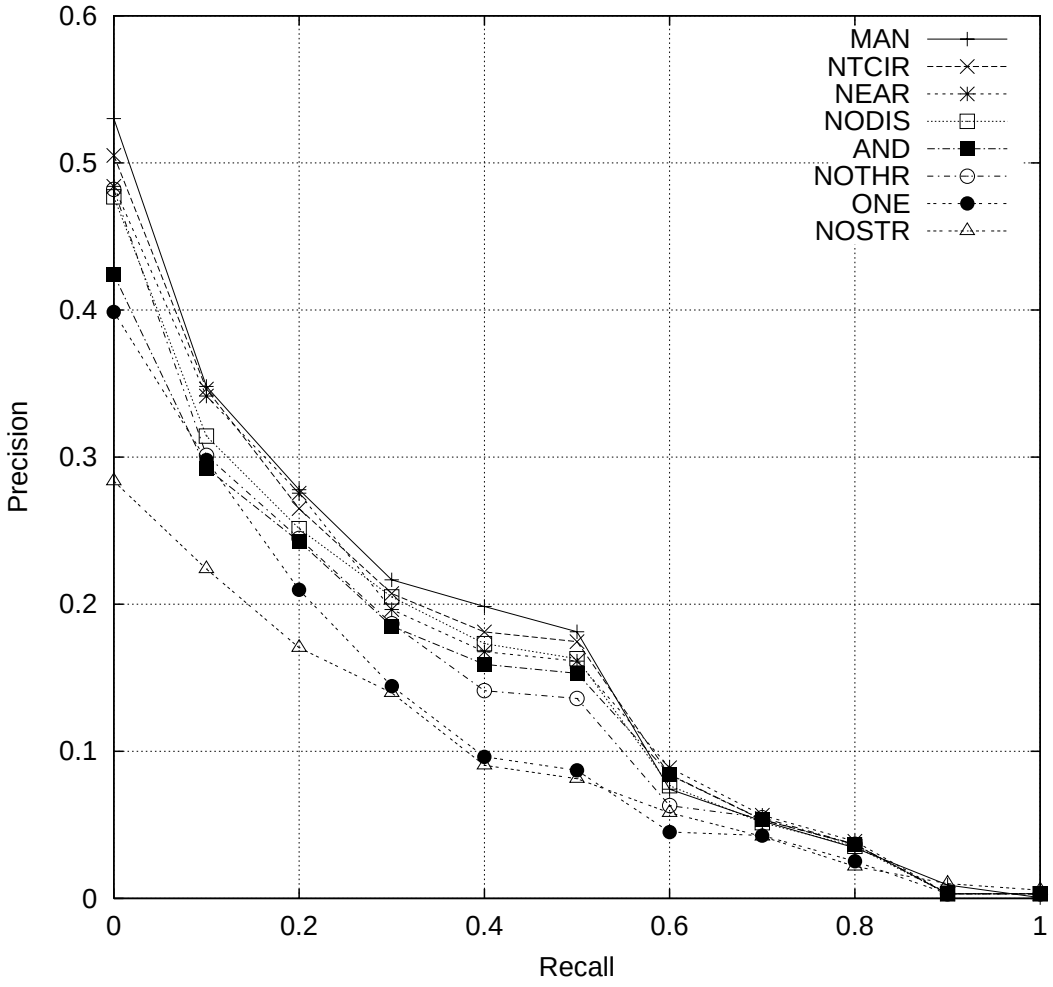


Figure 5.5. Recall-precision curves of Experiment 1.

Table 5.6. Results of the experiment using NACSIS Test Collection 1 (Experiment 1).

	Precision							
	NODIS	**NOSTR**	**ONE**	**NOTHR**	**AND**	**NEAR**	**NTCIR**	**MAN**
0.00	0.4769	0.2836	0.3987	0.4820	0.4241	0.4841	0.5051	0.5301
0.10	0.3143	0.2239	0.2980	0.3011	0.2925	0.3415	0.3463	0.3480
0.20	0.2513	0.1704	0.2097	0.2447	0.2427	0.2755	0.2650	0.2778
0.30	0.2050	0.1397	0.1443	0.1868	0.1851	0.1964	0.2072	0.2166
0.40	0.1732	0.0905	0.0963	0.1412	0.1591	0.1677	0.1812	0.1984
0.50	0.1629	0.0814	0.0871	0.1359	0.1530	0.1613	0.1745	0.1813
0.60	0.0766	0.0582	0.0451	0.0630	0.0843	0.0888	0.0840	0.0742
0.70	0.0514	0.0421	0.0428	0.0549	0.0536	0.0565	0.0534	0.0526
0.80	0.0354	0.0217	0.0253	0.0368	0.0368	0.0388	0.0366	0.0344
0.90	0.0031	0.0100	0.0031	0.0031	0.0031	0.0033	0.0031	0.0091
1.00	0.0031	0.0056	0.0031	0.0031	0.0031	0.0033	0.0031	0.0005
Avg.Prec.	0.1438	0.0896	0.1084	0.1362	0.1371	0.1513	0.1544	0.1564

translation-candidate that retrieves documents by using a search engine, all translation candidates were used.

NTCIR is the result of using the proposed disambiguation method, but by using NTCIR collection as a corpus instead of a Web search engine.

MAN is the result of English queries manually translated from the original Japanese queries.

In the table, the column 0.00-1.00 indicates the precisions at each recall level, and Avg. Prec. indicates the average precision.

For relevance judgments included in NTCIR collection, we used not only full relevance (A judgment), but also partial relevance (B judgment).

As an actual example of a disambiguation, the result of NEAR method for the query that consists of two words “神経 (nerve)” and “再生 (regeneration)” (topic ID 0080) is shown in Figure 5.6 and Table 5.7. In Table 5.7, top 7 pairs exceed the threshold value. In this case, the 1st, 3rd, and 4th pairs seem to be appropriate translations, but all 7 pairs combined by OR operators are used for the actual query.

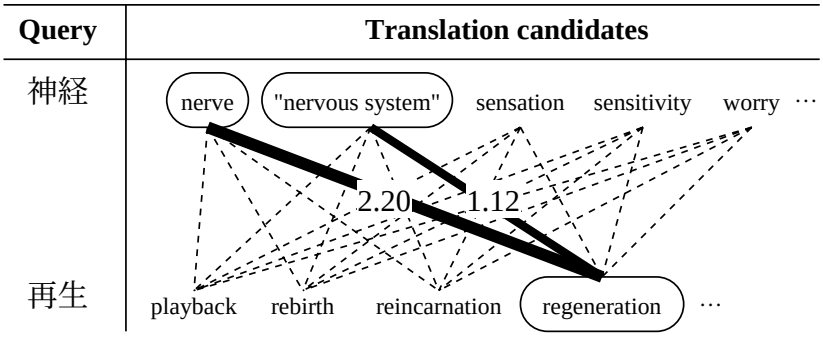


Figure 5.6. An example of disambiguation for a query used in experiments.

Table 5.7. An example of co-occurrence tendency values for a query used in experiments (top 15 pairs).

Rank	Translation candidate pair		COT_{MI2}
	“神経”	**“再生”**	
1	**nerve**	**regeneration**	2.20
2	nerve	regrowth	1.82
3	**“nervous system”**	**regeneration**	1.12
4	**nerves**	**regeneration**	0.54
5	nerves	regrowth	0.43
6	“nervous system”	regrowth	-0.17
7	sensation	regrowth	-1.52
8	sensation	reincarnation	-2.33
9	“nervous system”	resuscitation	-2.65
10	sensitivity	playback	-2.95
11	sensitivity	regeneration	-3.07
12	nerve	resuscitation	-3.07
13	sensation	regeneration	-3.09
14	worry	read	-3.27
15	sensation	rebirth	-3.91

Discussion of the Results

In terms of average precisions compared with NODIS, the proposed disambiguation method improved 1.0 point for NTCIR and 0.8 point for NEAR, but decreased 0.7 point for AND.

In general, the effectiveness of using all translation candidates in a dictionary is about a half of the one using monolingual retrieval[7], but our result of NODIS greatly surpassed it and achieved 92% of manual translation (MAN). It is probably because: 1) queries consist mostly of technical terms and their ambiguities were relatively low, 2) query structuring using Boolean operators was much more effective than expected. The result of NOSTR (all translation-candidate terms are combined with OR operators) was 57% of the result of MAN and it is comparable to previous studies.

The result of NTCIR, in which the corpus is identical to the target collection, achieved 99% of MAN and the effect of disambiguation was significant. By using a corpus which is consistent with the target, appropriate translations were selected in most of the cases. However, for the results of using Web documents as a corpus, NEAR achieved 97% of the result of MAN, but AND did 88% of MAN and it was lower than NODIS. It is probably due to the scope of the co-occurrence. In AND, the scope is one document and there are more chances for errors, but in NEAR, the scope is 10 words and relatively a better co-occurrence tendency was acquired.

In this experiment, NTCIR achieved the highest performance, but our primary target is Web CLIR. It is impractical to prepare a comprehensive corpus that covers all possible domains. Therefore, the proposed method using Web documents as a monolingual corpus will be effective for Web CLIR.

On the other hand, the result of using only one translation-candidate that has the highest COT_{MI2} achieved only 69% of MAN. The reason is that the highest-ranked translation-candidate is not always appropriate. In the proposed method, by including all translation candidates that exceed the threshold value, appropriate translations were selected in most of the cases. For example, in a query which consists of “Zipf” and “法則 (law or rule)”, the highest-ranked translation-candidate was the pair (“Zipf”, “rule”), but the most appropriate translation (“Zipf”, “law”) also exceeded the threshold, and was selected as the final query. Moreover, the number of the appropriate translations is not necessarily one. For example, the Japanese term “施設” has translations of similar meaning “facility” and “institution” in English. In some context, both

of them might be appropriate. In the proposed method, both of them were selected as an effect of selecting multiple translation candidates that exceed the threshold.

In addition, the topics used in this experiment are all for technical documents. They contain many technical terms and loan words, which are usually less ambiguous by nature. Therefore, 4 queries out of 39 were translated uniquely without requiring the disambiguation.

Moreover, the result of NOTHR, in which the threshold value for word occurrence frequency T_{freq} is zero, was 1.8 point decrease in average precision compared to the result of NTCIR, in which T_{freq} is $N/10,000$. This result proves the effectiveness of eliminating rarely occurred terms when calculating co-occurrence tendencies based on mutual information.

5.5.6 Discussion Concerning the Size of Corpus

In the experiment, we used 130 million Web documents as a corpus. We investigated the effect on selecting translations when the size of corpus is smaller.

In the case of AND operator, 2,910 candidate pairs exceeded the threshold T_{COT} for 39 queries. Among these candidate pairs, 1,233 pairs (42%) retrieved more than 100 documents ($f(w_1, w_2, \ldots w_n)$ in equation 5.3), 2,099 pairs (72%) retrieved more than 10 documents, and 2,689 pairs (92%) retrieved more than 2 documents. In the case of NEAR operator, among 304 documents that exceeded the threshold, 119 pairs (39%) retrieved more than 100 documents, 217 pairs (71%) retrieved more than 10 documents, and 243 pairs (90%) retrieved more than 2 documents.

If the size of corpus was $1/2$, $1/10$, or $1/100$, roughly 10%, 30%, or 60% of candidate pairs will not co-occur and will be excluded from candidates. Note that it is just a provisional conclusion, since the candidates that exceed the threshold are not necessarily appropriate translations. However, at least for the queries used in this experiment, we can say that there is a possibility that 130 million documents were not sufficient for obtaining co-occurrence of n words.

5.5.7 Experiment 2 (comparison of COT)

We have conducted a comparative experiment for four co-occurrence tendency measures described in Section 5.4. In this experiment, NTCIR collection was used as a

Table 5.8. Results of the comparative experiment about co-occurrence tendency measures (Experiment 2).

	Precision		
	DICE	**LLR**	**CHI**
0.00	0.5051	0.4973	0.4956
0.10	0.3464	0.3504	0.3498
0.20	0.2650	0.2651	0.2658
0.30	0.2070	0.2066	0.2073
0.40	0.1820	0.1794	0.1822
0.50	0.1753	0.1722	0.1755
0.60	0.0859	0.0863	0.0862
0.70	0.0546	0.0550	0.0549
0.80	0.0366	0.0370	0.0368
0.90	0.0031	0.0031	0.0031
1.00	0.0031	0.0031	0.0031
Avg. Prec.	0.1546	0.1550	0.1549

corpus.

Experimental Results

The results of the experiment are shown in Table 5.8 and Figure 5.7. NODIS and MAN in the figure are the same as experiment 1. DICE, LLR, and CHI are the results of disambiguation using modified Dice coefficient, log likelihood ratio, and χ^2 test, respectively.

Discussion of the Results

In this experiment, differences of the average precisions among four measures were lower than 0.1% and no significant difference was observed. It is probably due to the limitation of the target collection, which is a collection of technical documents in a limited domain. In this case, even if inappropriate translations were included in the final query, they were not likely to appear in the target collection. But for Web CLIR

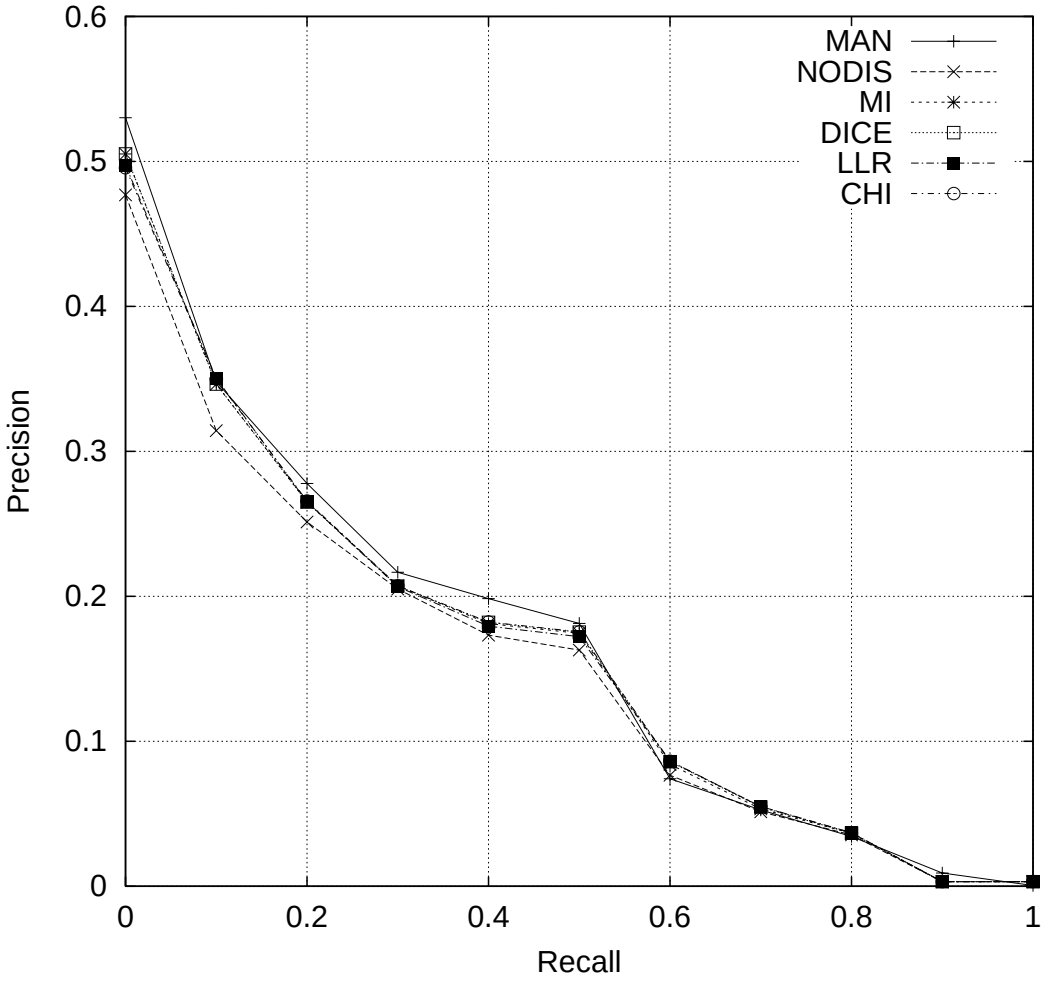


Figure 5.7. Recall-precision curves of Experiment 2.

which includes documents in various domains, difference among measures might be significant.

Since log likelihood ratio and χ^2 test require the frequency information containing non-occurrence (e.g. $w_1 \wedge \neg w_2$ etc.) and also they cannot be extended into n words, they require much more query to a search engine for calculating co-occurrence tendencies. Therefore, at least for the test collection and the retrieval system used in the experiments, we can conclude that either mutual information or modified Dice coefficient might be the best measure for co-occurrence tendency, when considering both effectiveness and efficiency.

5.6 Summary

In this chapter, we proposed a method for query term disambiguation using a Web search engine, which is readily available. The results of the experiments showed that the proposed method is effective for very short queries, which are often used by an end-user of Web search engines. We also showed that the proposed method can achieve comparable effectiveness with the manual translation, using a corpus which is consistent with the target collection.

Our method can easily be extended to other language pairs by preparing only a bilingual dictionary. Besides, disambiguated queries are simple Boolean queries and can be simply fed into an existing Web search engine.

Future work include detailed considerations on setting the threshold value and the scope of the co-occurrence, an evaluation using actual Web documents, extension to other language pairs, and a consideration on how quality and quantity of the corpus affect the retrieval effectiveness, as well as the possibility to apply some smoothing method[60].

Chapter 6

System Integration: *Multilingual Knowledge Discovery System*

6.1 Introduction

The techniques which were described in previous three sections, are used as a part of the *Multilingual Knowledge Discovery System*[75]. This system integrates three techniques described in previous chapters. The goal of this system is to manage multilingual Web documents in a unified manner, and to provide users with easy access to information written in foreign languages.

6.2 Related Work

Several systems have already been developed that supports CLIR of multilingual documents on the Web.

MULINEX[13] is a cross-language Web search engine that supports English, French, and German. It combines several existing linguistic techniques such as summarization, categorization, machine translation of summaries, and language identification. The language identification is used prior to summarization, categorization, and indexing. It employs Cavnar et al.'s method[14] described in Chapter 4 for the language identification. For the disambiguation of query terms, it does not employ any automatic disambiguation methods, but merely uses the user interaction, i.e. the user manually

select appropriate translation(s) for the target language query from the candidates obtained from a bilingual dictionary. In order to help users who do not understand the target language, it shows the translations how each translated query term translated back into the source language.

TITAN/CLMS[39] is a cross-language Web search engine that supports English, Japanese, Chinese, and Korean. It is based on TITAN[38] that supports English and Japanese. It employs Kikui's method[61] described in Chapter 4 for the language and coding system identification of the documents to be indexed. Like MULINEX, it does not employ any query term disambiguation methods. One notable feature of this system is the metasearch functionality. The system consists of several search sites that have different supported languages, e.g. site A supports English and Japanese, site B supports English and Chinese, and so on. The query requests that are not supported by the current site are routed to the site which can handle them. A similar approach has taken by Powell et al., which defines a translation request protocol for federated CLIR systems[89].

6.3 System Implementation

6.3.1 System Overview

Figure 6.1 illustrates the overview of the system. The system consists of three subsystems, i.e. *document collection* subsystem, *indexing* subsystem and *retrieval* subsystem.

6.3.2 Document Collection Subsystem

The *document collection* subsystem collects documents using a Web robot (also referred to as a Web crawler or a Web spider). An Web robot is a program that automatically traverses the Web's hypertext structure by retrieving a document, and recursively retrieving all documents that are referenced.

6.3.3 Indexing Subsystem

In the *indexing* subsystem, the collected documents are classified using *language and coding system identification* module which is implemented using the automatic identi-

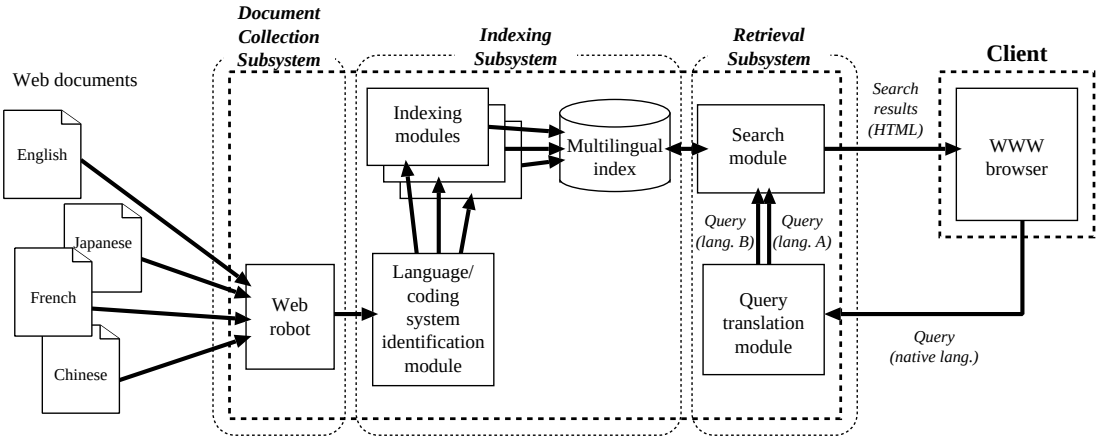


Figure 6.1. Overview of the Multilingual Knowledge Discovery System.

fication algorithm described in Chapter 4, then indexed by language dependent indexing modules, and finally stored into the index database.

6.3.4 Retrieval Subsystem

In the *retrieval* subsystem, user's query requests are translated into the language(s) the user specified, then documents are retrieved using translated query and returned to the user, using the CLIR technique described in Chapter 5.

Currently, it supports Japanese-to-English and English-to-Japanese CLIR. For query translation, we use EDICT freeware Japanese-English dictionary that was described in Chapter5. For query term disambiguation, the user can select between the AltaVista search engine and a locally maintained monolingual corpus. For co-occurrence measure for disambiguation, the user can select among four measures described in Chapter 5; mutual information, modified Dice coefficient, log likelihood ratio, and χ^2 test.

6.3.5 Text Input/Output Subsystem

Figure 6.2 illustrates the overview of the *text input/output* subsystem. This subsystem is implemented using the text display and input functions described in Chapter 3.

A query can be inputted using the MHTML Text Input System, and the search results are shown in the MHTML class. Thus the user does not have to install fonts

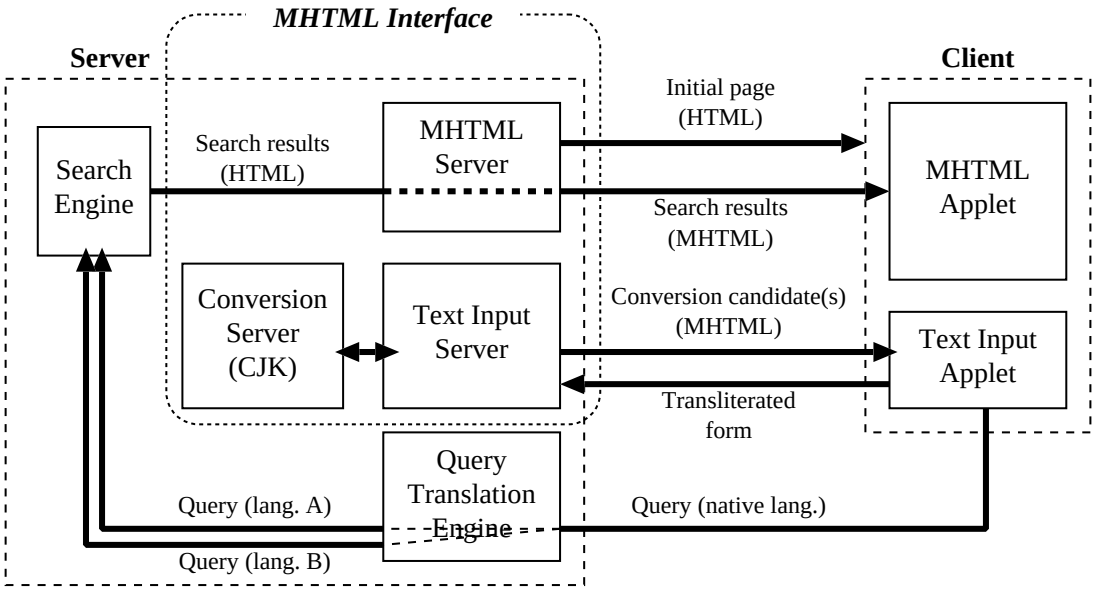


Figure 6.2. Architecture of the Text Input/Output Subsystem.

and input methods for searching Japanese documents.

Figure 6.3 shows the homepage of the prototype system.

6.4 Summary

In this chapter, the architecture of the *Multilingual Knowledge Discovery System*, which integrates the techniques described in previous three sections, was introduced and described in detail. This system is a prototype for Web-based multilingual information system which manages multilingual Web documents in a unified manner, and supports end-user access to information written in foreign languages.

This system realizes several functionalities that are not available on state-of-the-art Web search engines, such as multilingual text input/output without requiring fonts and input methods, and cross-language information retrieval of Web documents in diverse domains.

Still, there remain many unsolved problems in terms of multilingual information processing on the Internet. These problems might include a suitable indexing tech-

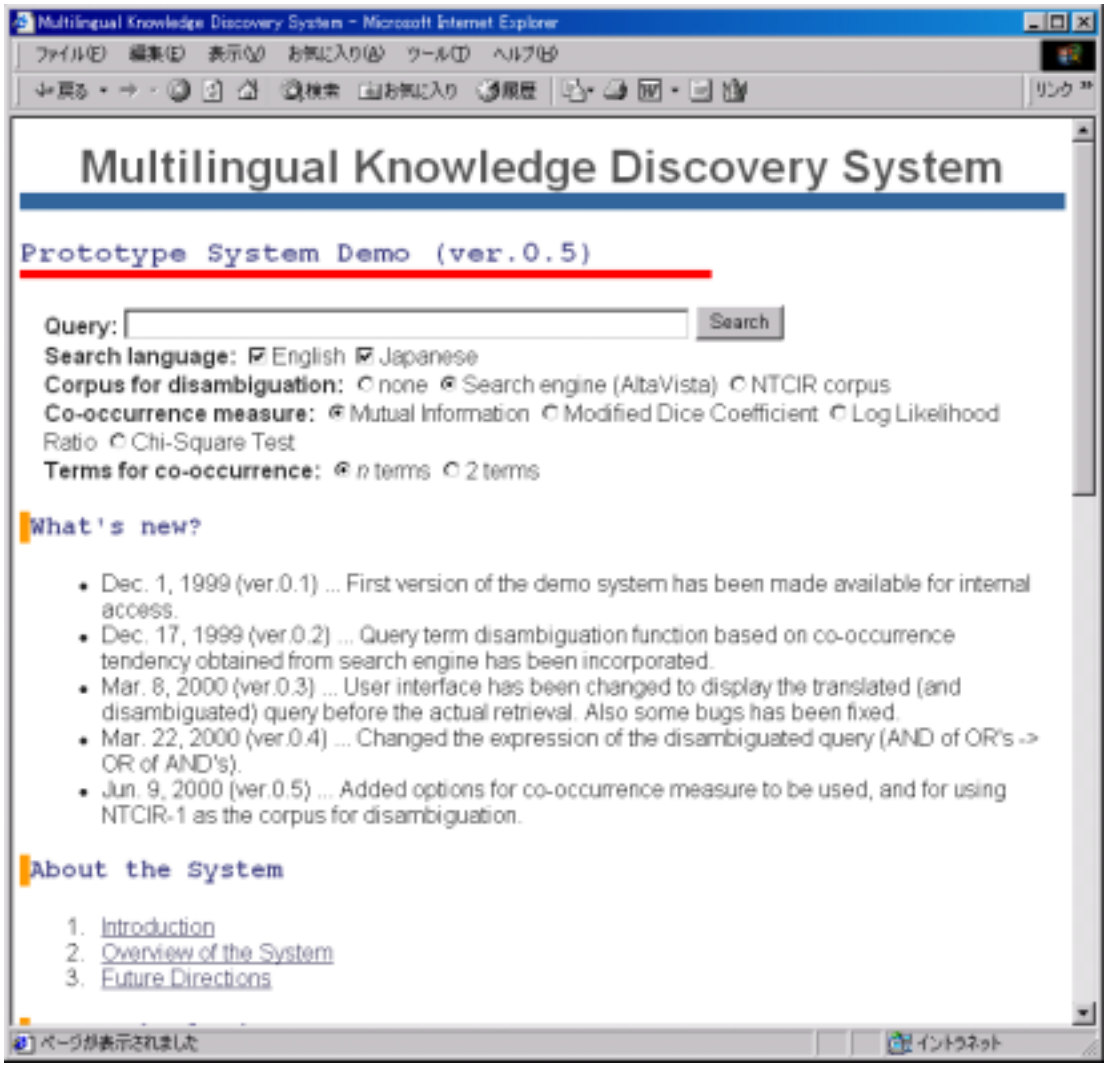


Figure 6.3. Homepage of the Multilingual Knowledge Discovery System.

nique of multilingual Web documents, a retrieval method of text including gaiji, and an implementation of input methods other than Japanese. And especially for CLIR, a consideration on cultural elements such as units, era names and color names in query translation, and a query translation through intermediary language dictionary[75] might be some of future works.

Chapter 7

Conclusions

In this thesis, we introduced practical solutions to several issues in realizing a information system which can handle multilingual documents in a unified manner. The issues we focused on are summarized as follows:

In Chapter 3, we introduced display and input functions of multilingual text which does not depend on installed fonts and input methods on the client side. The advantages of this technique are, it is easy to use by casual users of the Internet, it is inexpensive because no additional installation is required on the client side, and it is light-weight. This system might be especially useful for documents written in minor languages, for old literature which may include characters that are not defined in any standard character set, and for newspaper articles which may require gaiji's for some proper nouns.

In Chapter 4, we developed an algorithm for automatic identification of languages and character coding systems of Web documents based on combination of both statistics and heuristics. It is basic but important function in order to handle multilingual documents in practice. We conducted experiments for 12 languages and 10 coding systems, and achieved over 98% correct rate in average. For Asian languages that use multiple bytes to represent one character, we showed that it is possible to identify those languages and coding systems effectively and efficiently, simply using one byte unit and without discriminating the boundaries of characters. For European languages, we showed that it can achieve 98% average correct rate for 9 languages included in ISO-8859-1, by using the consecutive two bytes unit, which takes the connection of consecutive characters into account.

In Chapter 5, we investigated a cross-language information retrieval technique which is suitable for Web documents in diverse domains. The results of experiments showed that the proposed method is effective for very short queries which are often used by an end-user of Web search engines. We also showed that the proposed method can achieve comparable effectiveness with the manual translation, using a corpus which is consistent with the target collection. The main advantages of this technique are, since it is based on word co-occurrence information obtained from a Web search engine, we do not need to collect a large corpus of diverse domains, and it can easily be extended to other language pairs by preparing only a dictionary.

Finally, In Chapter 6, by integrating three techniques describe above, we realized a system which supports access to documents written in languages other than user's native language. This system provides some solutions to the problems in multilingual information processing that are specific to the Internet.

Future work of this research might include the extension of the overall system to other languages and coding systems, especially to multilingual coding systems such as Unicode.

References

[1] Gary Adams and Philip Resnik. A Language Identification Application Built on the Java Client/Server Platform. In *Proceedings of ACL'97/EACL'97 Workshop "From Research to Commercial Applications: Making NLP Technology Work in Practice"*, Madrid, Spain, July 1997.

[2] Badr H. Al-Badr. Using the Internet in Arabic: Problems and Solutions. In *Proceedings of INET'98*, Geneva, Switzerland, July 1998.

[3] Harald Tveit Alvestrand. Tags for the Identification of Languages. RFC 1766, March 1995.

[4] Harald Tveit Alvestrand. IETF Policy on Character Sets and Languages. RFC 2277, January 1998.

[5] American National Standards Institute. Coded character set – 7-bit American national standard code for information interchange. ANSI X3.4-1986, 1986.

[6] Lisa Ballesteros and W. Bruce Croft. Phrasal Translation and Query Expansion Techniques for Cross-Language Information Retrieval. In *Proceedings of the 20th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 84–91, Philadelphia, PA, July 1997.

[7] Lisa Ballesteros and W. Bruce Croft. Resolving Ambiguity for Cross-Language Retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pp. 64–71, Melbourne, Australia, August 1998.

[8] Joseph D. Becker. Arabic word processing. *Communications of the ACM*, Vol. 30, No. 7, pp. 600–610, July 1987.

[9] K. R. Beesley. Arabic Morphological Analysis on the Internet. In *Proceedings of the 6th International Conference and Exhibition on Multi-lingual Computing*, Cambridge, April 1998.

[10] Tim Berners-Lee, Roy T. Fielding, and Henrik Frystyk Nielsen. Hypertext Transfer Protocol – HTTP/1.0. RFC 1945, May 1996.

[11] Bert Bos, Håkon Wium Lie, Chris Lilley, and Ian Jacobs, editors. *Cascading Style Sheets, level 2: CSS2 Specification*, chapter 15. W3C Recommendation. World Wide Web Consortium, May 1998. http://www.w3.org/TR/REC-CSS2/.

[12] Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen, editors. *Extensible Markup Language (XML) 1.0*. W3C Recommendation. February 1998. http://www.w3.org/TR/REC-xml.

[13] Joanne Capstick, Abdel Kader Diagne, Gregor Erbach, Hans Uszkoreit, Anne Leisenberg, and Manfred Leisenberg. A system for supporting cross-lingual information retrieval. *Information Processing & Management*, Vol. 36, No. 2, pp. 275–289, March 2000.

[14] William B. Cavnar and John M. Trenkle. N-Gram-Based Text Categorization. In *Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval*, pp. 161–169, April 1994.

[15] Uhhyung Choi, Kilnam Chon, and Hyunje Park. Korean Character Encoding for Internet Messages. RFC 1557, December 1993.

[16] Kenneth Ward Church and Patrick Hanks. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, Vol. 16, No. 1, pp. 22–29, March 1990.

[17] Dan Connolly. *Character Set* Considered Harmful. W3C notes, The World Wide Web Consortium, May 1995. http://www.w3.org/MarkUp/html-spec/charset-harmful.

[18] Ido Dagan and Alon Itai. Word Sense Disambiguation Using a Second Language Monolingual Corpus. *Computational Linguistics*, Vol. 20, No. 4, pp. 563–596, December 1994.

[19] Myriam Dartois, Akira Maeda, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Building a Multilingual Electronic Text Collection of Folk Tales as a Set of Encapsulated Document Objects: an Approach for Casual Users to Browse Multilingual Documents on the Fly. In *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries (ECDL'97)*, Lecture Notes in Computer Science 1324, Springer-Verlag, pp. 215–231, Pisa, Italy, September 1997.

[20] Myriam Dartois, Akira Maeda, Tetsuo Sakaguchi, Takehisa Fujita, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Electronic Text Collection of Folk Tales for Casual Users Using Off-the-Shelf Browsers. *D-Lib Magazine*, October 1997. http://www.dlib.org/.

[21] Mark Davis. The Bidirectional Algorithm. Unicode Technical Report #9, November 1999.

[22] Mark Davis and William C. Ogden. QUILT: Implementing a Large-scale Cross-Language Text Retrieval System. In *Proceedings of the 20th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*, pp. 92–98, Philadelphia, PA, July 1997.

[23] Susan T. Dumais, Todd A. Letsche, Michael L. Littman, and Thomas K. Landauer. Automatic Cross-Language Retrieval Using Latent Semantic Indexing. In *Electronic Working Notes of the AAAI Symposium on Cross-Language Text and Speech Retrieval*, March 1997.

[24] Ted Dunning. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, Vol. 19, No. 1, pp. 61–74, March 1993.

[25] Ted Dunning. Statistical Identification of Language. Technical Report CRL MCCS-94-273, Computing Research Lab, New Mexico State University, March 1994.

[26] Sadat Fatiha, Akira Maeda, Masatoshi Yoshikawa, and Shunsuke Uemura. Integrating Dictionary-based and Statistical-based Approaches in Cross-Language Information Retrieval. In *IPSJ SIG Notes*, 2000-DBS-121 / 2000-FI-58, pp. 61–68, May 2000.

[27] Roy T. Fielding, James Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul J. Leach, and Tim Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, July 1999.

[28] Ned Freed and Nathaniel S. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. RFC 2045, November 1996.

[29] Atsushi Fujii and Tetsuya Ishikawa. Cross-Language Information Retrieval for Technical Documents. In *Proceedings of the Joint ACL SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 29–37, July 1999.

[30] Takehisa Fujita, Akira Maeda, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Japanese OPAC for Overseas Users. *Digital Libraries*, No. 6, pp. 32–39, February 1996. (in Japanese).

[31] Pierre Gaumond, Philip A. Nelson, and Jason Downs. *GNU dbm – A Database Manager: Manual Edition 1.5 for GNU dbm, Version 1.8.* Free Software Foundation, Cambridge, MA, May 1999.

[32] Emmanuel Giguet. Multilingual Sentence Categorization according to Language. In *Proceedings of the European Chapter of the Association for Computational Linguistics SIGDAT Workshop "From Text to Tags: Issues in Multilingual Language Analysis"*, pp. 73–76, Dublin, Ireland, March 1995.

[33] David Goldsmith and Mark Davis. A Mail-Safe Transformation Format of Unicode. RFC 2152, May 1997.

[34] Gregory Grefenstette, editor. *Cross-Language Information Retrieval*, volume 2 of *The Kluwer International Series on Information Retrieval*. Kluwer Academic Publishers, March 1998.

[35] Hachim Haddouti, Akira Maeda, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Towards Arabic Rendering Issues — MHTML Approach. In *Proceedings of the Arabic Translation and Localisation Symposium (ATLAS'99)*, Tunis, Tunisia, May 1999.

[36] Zhu Haifeng, Hu Daoyuan, Wang Zhiguan, Kao Tien-cheu, Chang Wen-chung, and Mark R. Crispin. Chinese Character Encoding for Internet Messages. RFC 1922, March 1996.

[37] David Hawking, Nick Craswell, Paul Thistlewaite, and Donna Harman. Results and Challenges in Web Search Evaluation. In *Proceedings of the 8th Inernational WWW Conference*, pp. 243–252, Toronto, Canada, May 1999.

[38] Yoshihiko Hayashi, Gen'ichiro Kikui, and Seiji Susaki. TITAN: A Cross-Linguistic Search Engine for the WWW. In *Electronic Working Notes of the AAAI Symposium on Cross-Language Text and Speech Retrieval*, March 1997.

[39] Yoshihiko Hayashi, Genichiro Kikui, Seiji Susaki, Toshiaki Iwadera, and Kazunari Watanabe. Toward Web Navigation across Language Barriers: a Big Picture and Current Trials in TITAN Project. In *Proceedings of the 8th Inernational WWW Conference*, pp. 158–159, Toronto, Canada, May 1999.

[40] Toru Hisamitsu and Yoshiki Niwa. Information Extraction from Parenthetical Expressions by Using Statistical Measures and Simple Rules. In *IPSJ SIG Notes*, NL-122-17, pp. 113–118, November 1997. (in Japanese).

[41] Atsushi Ikeno, Toshiki Murata, Sayori Shimohata, and Hideki Yamamoto. Machine Translation using the Internet Natural Language Resources. In *Proceedings of World TELECOM99+Interactive99 Forum*, Geneva, Switzerland, October 1999.

[42] Institute for Information Industry. Chinese Coded Character Set in Computer, March 1984.

[43] International Organization for Standardization. Information technology: ISO 7-bit Coded Character Set for Information Interchange. International Standard ISO/IEC 646:1991, 1991.

[44] International Organization for Standardization. Information technology – Portable Operating System Interface (POSIX) – Part 2: Shell and Utilities. International Standard ISO/IEC 9945-2:1993, 1993.

[45] International Organization for Standardization. Information Technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane. International Standard ISO/IEC 10646-1:1993, October 1993.

[46] International Organization for Standardization. Information technology – Character Code Structure and Extension Techniques. International Standard ISO/IEC 2022:1994, January 1994.

[47] International Organization for Standardization. Transformation Format for 16 planes of group 00 (UTF-16). Technical Report ISO/IEC 10646-1:1993/Amd 1:1996, October 1996.

[48] International Organization for Standardization. Information technology – 8-bit single-byte coded graphic character sets – Part 6: Latin/Arabic alphabet. International Standard ISO/IEC 8859-6:1997, August 1997.

[49] International Organization for Standardization. Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1. International Standard ISO/IEC 8859-1:1998, April 1998.

[50] Myung-Gil Jang, Sung Hyon Myaeng, and Se Young Park. Using Mutual Information to Resolve Query Translation Ambiguities and Query Term Weighting. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL'99)*, pp. 223–229, Washington D.C., June 1999.

[51] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. Real Life, Real Users, and Real Needs: a Study and Analysis of User Queries on the Web. *Information Processing & Management*, Vol. 36, No. 2, pp. 207–227, March 2000.

[52] Japan Electronic Dictionary Research Institute, Ltd. EDR Electronic Dictionary Version 1.5 Technical Guide. Technical Report TR2-007, Japan Electronic Dictionary Research Institute, Ltd., 1996.

[53] Japanese Standards Association. Code of the supplementary Japanese graphic character set for information exchange. JIS X 0212:1990, 1990. (in Japanese).

[54] Japanese Standards Association. 7-bit and 8-bit coded character sets for information exchange. JIS X 0201:1997, 1997. (in Japanese).

[55] Japanese Standards Association. 7-bit and 8-bit double byte coded KANJI sets for information interchange. JIS X 0208:1997, 1997. (in Japanese).

[56] Japanese Standards Association. 7-bit and 8-bit double byte coded KANJI sets for information interchange – Annex 1: shift code representation. JIS X 0208:1997, 1997. (in Japanese).

[57] Noriko Kando, Kazuko Kuriyama, Toshihiko Nozue, Koji Eguchi, Hiroyuki Kato, Soichiro Hidaka, and Jun Adachi. The NTCIR Workshop: the First Evaluation Workshop on Japanese Text Retrieval and Cross-lingual Information Retrieval. In *Proceedings of the 4th International Workshop on Information Retrieval with Asian Languages (IRAL'99)*, Taipei, Taiwan, November 1999.

[58] Yutaka Kataoka, Tomoko I. Kataoka, Kazutomo Uezono, and Hiroyoshi Ohara. The Essentials for Developing Multilingual Computer Environment. In *Proceedings of the 2nd Workshop on Multilinguality in Software Industry: The AI Contribution (MULSAIC'97)*, Nagoya, Japan, August 1997.

[59] Yutaka Kataoka, Masato Morisaki, Hiroshi Kuribayashi, and Hiroyoshi Ohara. A Model for Input and Output of Multilingual Text in a Windowing Environment. *ACM Transactions on Information Systems*, Vol. 10, No. 4, pp. 438–451, October 1992.

[60] Salva M. Katz. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-35, No. 3, pp. 400–401, March 1987.

[61] Genichiro Kikui. Identifying the Coding System and Language of On-line Documents Using Statistical Language Models. *Transactions of IPSJ*, Vol. 38, No. 12, pp. 2440–2448, December 1997.

[62] Mihoko Kitamura and Yuji Matsumoto. Automatic Extraction of Translation Patterns in Paralell Corpora. *Transactions of IPSJ*, Vol. 38, No. 4, pp. 727–736, April 1997. (in Japanese).

[63] Judith L. Klavans and Peter Schäuble. NSF–EU Multilingual Information Access. *Communications of the ACM*, Vol. 41, No. 4, p. 69, April 1998.

[64] Thaweesak Koanantakool. The Keyboard Layouts and Input Method of the Thai Language. In *Proceedings of the Symposim on Natural Language Pocessing in Thailand*, 1993.

[65] Korean Industrial Standard. UNIX-Hangul Environment. KS X 2901:1992, 1992. (in Korean).

[66] Steve Lawrence and C. Lee Giles. Accessibility of Information on the Web. *Nature*, Vol. 400, No. 6740, pp. 107–109, July 1999.

[67] Chuan-Jie Lin, Wen-Cheng Lin, Guo-Wei Bian, and Hsin-Hsi Chen. Description of the NTU Japanese-English Cross-Lingual Information Retrieval System Used for NTCIR Workshop. In *Proceedings of the First NTCIR Workshop on Research in Japanese Text Retrieval and Term Recognition*, pp. 145–148, Tokyo, Japan, August 1999.

[68] Akira Maeda. Multilingual Browser for WWW Documents and its Server System. Master's thesis, University of Library and Information Science, March 1997. (in Japanese).

[69] Akira Maeda, Myriam Dartois, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Viewing Multilingual Documents on Your Local Web Browser. *Communications of the ACM*, Vol. 41, No. 4, pp. 64–65, April 1998.

[70] Akira Maeda, Myriam Dartois, Jun Ohta, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. A Multilingual HTML Document Browsing System for Clients without Multilingual Fonts. *Transactions of IPSJ*, Vol. 39, No. 3, pp. 802–809, March 1998. (in Japanese).

[71] Akira Maeda, Sadat Fatiha, Masatoshi Yoshikawa, and Shunsuke Uemura. Query Term Disambiguation for Web Cross-Language Information Retrieval using a Search Engine. In *Proceedings of the 5th International Workshop on Information Retrieval with Asian Languages (IRAL2000)*, Hong Kong, China, September 2000. (to appear).

[72] Akira Maeda, Takehisa Fujita, Lee Swee Choo, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Browser for WWW without Preloaded Fonts. In *Proceedings of the International Symposium on Digital Libraries 1995 (ISDL'95)*, pp. 269–270, Tsukuba, Japan, August 1995.

[73] Akira Maeda, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Multilingual Browser for WWW Documents and its Gateway Service. In *IPSJ SIG Notes*, 96-FI-44, pp. 1–7, November 1996. (in Japanese).

[74] Akira Maeda, Qingyan Guan, and Shunsuke Uemura. Towards a Multilingual Knowledge Discovery System. In *IPSJ SIG Notes*, 99-DBS-118 / 99-FI-54, pp. 1–8, May 1999. (in Japanese).

[75] Akira Maeda and Shunsuke Uemura. Key Technologies for Multilingual Information Processing on WWW. In *Proceedings of the Fourth International Symposium on Standardization of Multilingual Information Technology (MLIT-4)*, pp. 15–25, Yangon, Myanmar, October 1999.

[76] Akira Maeda, Masatoshi Yoshikawa, and Shunsuke Uemura. A Query Disambiguation Method for Cross-Language Information Retrieval Using Web Documents. *IPSJ Transactions of Databases*, Vol. 41, No. SIG 6 (TOD 7), October 2000. (in Japanese) (to appear).

[77] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*, chapter 5, pp. 141–177. MIT Press, Cambridge, MA, May 1999.

[78] Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Hiroshi Matsuda, and Masayuki Asahara. Japanese Morphological Analysis System ChaSen version 2.0 Manual 2nd edition. Technical Report NAIST-IS-TR99013, Nara Institute of Science and Technology, December 1999.

[79] Muhammad F. Mudawwar. Multicode: A Truly Multilingual Approach to Text Encoding. *Computer*, Vol. 30, No. 4, pp. 37–43, April 1997.

[80] Jun Murai, Mark Crispin, and Erik M. van der Poel. Japanese Character Encoding for Internet Messages. RFC 1468, June 1993.

[81] Jian-Yun Nie, Michel Simard, Pierre Isabelle, and Richard Durand. Cross-Language Information Retrieval based on Parallel Texts and Automatic Mining of Parallel Texts from the Web. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pp. 74–81, Berkeley, CA, August 1999.

[82] Mikiko Nishikimi, Kenichi Handa, and Satoru Tomura. Mule: MULtilingual Enhancement to GNU Emacs. In *Proceedings of INET'93*, San Francisco, CA, August 1993.

[83] Mikiko Nishikimi, Naoto Takahashi, Satoru Tomura, Kenichi Handa, Seiji Kuwari, Shinichi Mukaigawa, and Tomoko Yoshida. *Creating a Multilingual Environment: Multilingualization Using X Windows, Wnn, Mule, and WWW Browsers*, chapter 2, pp. 43–113. Prentice Hall Japan, September 1996. (in Japanese).

[84] Douglas W. Oard. Alternative Approaches for Cross-Language Text Retrieval. In *Electronic Working Notes of the AAAI Symposium on Cross-Language Text and Speech Retrieval*, March 1997.

[85] Douglas W. Oard and Jianqiang Wang. Effects of Term Segmentation on Chinese/English Cross-Language Information Retrieval. In *Proceedings of the Symposium on String Processing and Information Retrieval (SPIRE'99)*, Cancun, Mexico, September 1999.

[86] Masataka Ohta and Ken'ichi Handa. ISO-2022-JP-2: Multilingual Extension of ISO-2022-JP. RFC 1554, December 1993.

[87] Open Software Foundation and UNIX International and UNIX System Laboratories. OSF, UI, and USL Standardize on Japanese Language Support. UI-OSF-USLP Joint Announcement, December 1991. (Press release dated December 12, 1991).

[88] Noritaka Osawa and Toshitsugu Yuba. An Efficient, Programmable and Interchangeable Code System: EPICS. *IEICE Transactions on Information and Systems*, Vol. E83-D, No. 4, pp. 797–806, April 2000.

[89] James Powell and Edward A. Fox. Multilingual Federated Searching Across Heterogeneous Collections. *D-Lib Magazine*, September 1998. http://www.dlib.org/.

[90] Technical Standards Press. Code of Chinese Graphic Character Set for Information Interchange Primary Set. GB 2312-80, 1981. (in Chinese).

[91] Dave Raggett, Arnaud Le Hors, and Ian Jacobs, editors. *HTML 4.01 Specification*. December 1999. http://www.w3.org/TR/html401/.

[92] Berthier Ribeiro-Neto and Ricardo Baeza-Yates. *Modern Information Retrieval*, chapter 3, pp. 73–97. Addison-Wesley, Reading, MA, 1999.

[93] Tetsuo Sakaguchi, Akira Maeda, Takehisa Fujita, Shigeo Sugimoto, and Koichi Tabata. A Browsing Tool for Multi-lingual Documents for Users without Multilingual Fonts. In *Proceedings of the First ACM International Conference on Digital Libraries (DL'96)*, pp. 63–71, Bethesda, MD, March 1996.

[94] Tetsuo Sakaguchi, Shigetaka Nakao, Akira Maeda, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Full-text Retrieval System for Tagged Documents. In *Proceedings of the 7th Annual Conference of Japan Society of Information and Knowledge*, pp. 49–52, May 1999. (in Japanese).

[95] Ken Sakamura. Multi-language Character Sets Handling in TAD. In *TRON Project 1987*, pp. 97–111. Springer-Verlag, 1987.

[96] Yutaka Sato. Multipurpose Protocol Mediation System: DeleGate. *Bulletin of the Electrotechnical Laboratory*, Vol. 59, No. 6, pp. 1–17, 1995. (in Japanese).

[97] Páraic Sheridan and Jean Paul Ballerini. Experiments in Multilingual Information Retrieval Using the SPIDER System. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 58–65, Zurich, Switzerland, August 1996.

[98] Shigeo Sugimoto, Akira Maeda, Tetsuo Sakaguchi, Koichi Tabata, and Takehisa Fujita. Experimental Studies on Software Tools to Enhance Accessibility to Information in Digital Libraries. *Journal of Network and Computer Applications*, Vol. 20, No. 1, pp. 25–43, January 1997.

[99] Shigeo Sugimoto, Shigetaka Nakao, Myriam Dartois, Jun Ohta, Akira Maeda, Tetsuo Sakaguchi, and Koichi Tabata. Extension of MHTML to Text Input and Text Search Functions in Multiple Languages on Off-the-shelf Browsers. In *Proceedings of the International Joint Workshop on Digital Libraries (IJWDL'98)*, pp. 75–82, Bangkok, Thailand, September 1998.

[100] The Unicode Consortium. *The Unicode Standard, Version 3.0*. Addison-Wesley, Reading, MA, 2000.

[101] C.J. van Rijsbergen. *Information Retrieval*, chapter 3. Butterworths, 2nd edition, 1979.

[102] Chris Weider, Cecilia Preston, Keld Simonsen, Harald T. Alvestrand, Randall Atkinson, Mark Crispin, and Peter Svanberg. The Report of the IAB Character Set Workshop held 29 February – 1 March, 1996. RFC 2130, April 1997.

[103] Kiyoshi Yamabana, Kazunori Muraki, Shinichi Doi, and Shin-ichiro Kamei. A Language Conversion Front-End for Cross-Linguistic Information Retrieval. In Gregory Grefenstette, Alan Smeaton, and Páraic Sheridan, editors, *Workshop on Cross-Linguistic Information Retrieval*, pp. 34–39. ACM SIGIR, August 1996.

[104] Yiming Yang, Jaime G. Carbonell, Ralf D. Brown, and Robert E. Frederking. Translingual Information Retrieval: Learning from Bilingual Corpora. *Artificial Intelligence*, Vol. 103, No. 1–2, pp. 323–345, August 1998.

[105] François Yergeau. UTF-8, a transformation format of ISO 10646. RFC 2279, January 1998.

[106] Leong Kok Yong, Liu Hai, and Oliver P. Wu. Web Internationalization and Java Keyboard Input Methods. In *Proceedings of INET'98*, Geneva, Switzerland, July 1998.

List of Publications

Journal Papers

1. Akira Maeda, QingYan Guan, Masatoshi Yoshikawa, and Shunsuke Uemura. Automatic Identification of Coding Systems and Languages of Web Documents. *Transactions of IEICE D-II*. (in Japanese) (to appear)

2. Akira Maeda, Masatoshi Yoshikawa, and Shunsuke Uemura. A Query Disambiguation Method for Cross-Language Information Retrieval Using Web Documents. *IPSJ Transactions of Databases*, Vol. 41, No. SIG 6 (TOD 7), October 2000. (in Japanese) (to appear)

3. Akira Maeda, Myriam Dartois, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Viewing Multilingual Documents on Your Local Web Browser. *Communications of the ACM*, Vol. 41, No. 4, pp. 64–65, April 1998.

4. Akira Maeda, Myriam Dartois, Jun Ohta, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. A Multilingual HTML Document Browsing System for Clients without Multilingual Fonts. *Transactions of IPSJ*, Vol. 39, No. 3, pp. 802–809, March 1998. (in Japanese) *1998 IPSJ Best Paper Award*

International Conferences

1. Akira Maeda, Sadat Fatiha, Masatoshi Yoshikawa, and Shunsuke Uemura. Query Term Disambiguation for Web Cross-Language Information Retrieval using a Search Engine. In *Proceedings of the 5th International Workshop on Information Retrieval with Asian Languages (IRAL2000)*, Hong Kong, China, September

2000. (to appear)

2. Akira Maeda and Shunsuke Uemura. Key Technologies for Multilingual Information Processing on WWW. In *Proceedings of the Fourth International Symposium on Standardization of Multilingual Information Technology (MLIT-4)*, pp. 15–25, Yangon, Myanmar, October 1999.

3. Akira Maeda, Takehisa Fujita, Lee Swee Choo, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Browser for WWW without Preloaded Fonts. In *Proceedings of the International Symposium on Digital Libraries 1995 (ISDL'95)*, pp. 269–270, Tsukuba, Japan, August 1995.

Domestic Conferences

1. Akira Maeda, Qingyan Guan, and Shunsuke Uemura. Towards a Multilingual Knowledge Discovery System. In *IPSJ SIG Notes*, 99-DBS-118 / 99-FI-54, pp. 1–8, May 1999. (in Japanese)

2. Akira Maeda, Qingyan Guan, and Shunsuke Uemura. Automatic Identification of Coding Systems of Documents Based on 1 Byte Code Distribution. In *Proceedings of the 1999 IEICE General Conference*, D-5-3, p. 77, March 1999. (in Japanese)

3. Akira Maeda, Myriam Dartois, Jun Ohta, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Multilingual HTML Document Delivery System without Fonts on the Browser's Terminal. In *Proceedings of the 55th National Convention IPSJ*, 1N-02, pp. 68–69, September 1997. (in Japanese)

4. Akira Maeda, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Multilingual Browser for WWW Documents and its Gateway Service. In *IPSJ SIG Notes*, 96-FI-44, pp. 1–7, November 1996. (in Japanese)

5. Akira Maeda, Takehisa Fujita, Lee Swee Choo, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Browser for WWW without Preloaded Fonts. *Digital Libraries*, No. 4, pp. 21–25, August 1995. (in Japanese)

Other Publications

Journal Papers

1. Myriam Dartois, Akira Maeda, Tetsuo Sakaguchi, Takehisa Fujita, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Electronic Text Collection of Folk Tales for Casual Users Using Off-the-Shelf Browsers. *D-Lib Magazine*, October 1997. http://www.dlib.org/

2. Shigeo Sugimoto, Akira Maeda, Tetsuo Sakaguchi, Koichi Tabata, and Takehisa Fujita. Experimental Studies on Software Tools to Enhance Accessibility to Information in Digital Libraries. *Journal of Network and Computer Applications*, Vol. 20, No. 1, pp. 25–43, January 1997.

3. Akira Maeda, Tetsuo Sakaguchi, and Koichi Tabata. Performance Measurement of Campus LAN in ULIS. *Research Report of University of Library and Information Science*, Vol. 15, No. 1, pp. 1–14, September 1996. (in Japanese)

International Conferences

1. Hachim Haddouti, Akira Maeda, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Towards Arabic Rendering Issues — MHTML Approach. In *Proceedings of the Arabic Translation and Localisation Symposium (ATLAS'99)*, Tunis, Tunisia, May 1999.

2. Shigeo Sugimoto, Akira Maeda, Myriam Dartois, Jun Ohta, Shigetaka Nakao, Tetsuo Sakaguchi, and Koichi Tabata. Experimental Studies on an Applet-based Document Viewer for Multilingual WWW Documents — Functional Extension of and Lessons Learned from Multilingual HTML. In *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries (ECDL'98)*, Lecture Notes in Computer Science 1513, Springer-Verlag, pp. 199–214, Heraklion, Greece, September 1998.

3. Shigeo Sugimoto, Shigetaka Nakao, Myriam Dartois, Jun Ohta, Akira Maeda, Tetsuo Sakaguchi, and Koichi Tabata. Extension of MHTML to Text Input and

Text Search Functions in Multiple Languages on Off-the-shelf Browsers. In *Proceedings of the International Joint Workshop on Digital Libraries (IJWDL'98)*, pp. 75–82, Bangkok, Thailand, September 1998.

4. Myriam Dartois, Akira Maeda, Takehisa Fujita, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Building a Multilingual Electronic Text Collection of Folk Tales as a Set of Encapsulated Document Objects: an Approach for Casual Users to Browse Multilingual Documents on the Fly. In *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries (ECDL'97)*, Lecture Notes in Computer Science 1324, Springer-Verlag, pp. 215–231, Pisa, Italy, September 1997.

5. Shigeo Sugimoto, Akira Maeda, Takehisa Fujita, Tetsuo Sakaguchi, and Koichi Tabata. A Gateway Service on WWW for Browsing Multilingual Documents. In *Proceedings of the Second NAIST Symposium on Digital Libraries*, Nara, Japan, December 1996.

6. Tetsuo Sakaguchi, Akira Maeda, Takehisa Fujita, Shigeo Sugimoto, and Koichi Tabata. A Browsing Tool for Multi-lingual Documents for Users without Multilingual Fonts. In *Proceedings of the First ACM International Conference on Digital Libraries (DL'96)*, pp. 63–71, Bethesda, MD, March 1996.

Domestic Conferences

1. Sadat Fatiha, Akira Maeda, Masatoshi Yoshikawa, and Shunsuke Uemura. Integrating Dictionary-based and Statistical-based Approaches in Cross-Language Information Retrieval. In *IPSJ SIG Notes*, 2000-DBS-121 / 2000-FI-58, pp. 61–68, May 2000.

2. Tetsuo Sakaguchi, Shigetaka Nakao, Akira Maeda, Shigeo Sugimoto, and Koichi Tabata. A Multilingual Full-text Retrieval System for Tagged Documents. In *Proceedings of the 7th Annual Conference of Japan Society of Information and Knowledge*, pp. 49–52, May 1999. (in Japanese)

3. Shigetaka Nakao, Myriam Dartois, Akira Maeda, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. A System for Building a Full-text Multilingual

Database Accessible from Any WWW Browser. In *Proceedings of the 6th Annual Conference of Japan Society of Information and Knowledge*, pp. 61–64, May 1998. (in Japanese)

4. Akira Maeda and Shunsuke Uemura. A Size Analysis of Multimedia Files on WWW. In *Proceedings of the 9th Data Engineering Workshop (DEWS'98)*, March 1998. (in Japanese)

5. Tetsuo Sakaguchi, Shigetaka Nakao, Jun Ohta, Myriam Dartois, Akira Maeda, Shigeo Sugimoto, and Koichi Tabata. Accessing Multilingual Information on Digital Libraries. In *IPSJ SIG Notes*, EIP-3, pp. 5–12, January 1998. (in Japanese)

6. Takehisa Fujita, Akira Maeda, Tetsuo Sakaguchi, Shigeo Sugimoto, and Koichi Tabata. Japanese OPAC for Overseas Users. *Digital Libraries*, No. 6, pp. 32–39, February 1996. (in Japanese)