LDAP Cross-searching for Traditional and Simplified Chinese

Chi-Chien Pan and Kai-Hsiang Yang and Tzao-Lin Lee
Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan, R.O.C.
E-mail: {d5526001, f6526004, tl_lee}@csie.ntu.edu.tw

0. Abstract

Chinese language has a lot of different properties with the western language, such as encoding methods, vocabulary, and character sets, etc.; it is usually presented in two forms: Simplified Chinese (SC), used in the PRC and Singapore, and Traditional Chinese (TC), used in Taiwan, Hong Kong, Macao, and among most overseas Chinese. Therefore, it is very difficult to search or integrate information from Chinese data containing the two forms. On the other hand, as the Internet services rapidly grow up, the LDAP service is widely deployed for the convenience of management, however the search mechanism in LDAP does not consider the properties of Chinese language. In this paper, we designed a new mechanism for LDAP server to cross-search between Traditional and Simplified Chinese, and we also implemented it in an open source LDAP server (OpenLDAP) [20].

1. Introduction

As the interaction between Taiwan and PRC increases persistently, more and more enterprises have branch companies at both regions; the company members come from different places, and even use different Operating Systems (such as TC or SC). Those enterprises always face an urgent need to integrate all kinds of resources by the Internet VPN network technology, however, the first incoming problem is the conversion between the TC and SC information., such as: document, email, database, etc.

Recently LDAP service is widely deployed to integrate and manage all kinds of resources for a company; however, the original design of LDAP does not consider the special properties of Chinese language at all, this makes the integration of company resources more difficult. For example, one enterprise has two branch companies at Taiwan and PRC, and the names of employees may be in both TC and SC. There are several problems needed to be solved: (1) How to integrate all name strings into LDAP server, and (2) the employees of two branch companies use different Operating Systems (one for TC, and one for SC), but how to input data simultaneously into one LDAP server, and (3) how could we search or retrieve data from the information containing both forms in the LDAP server. (4) How to convert between TC and SC documents when we search for one employee's name.

Against these problems, there are several researches [2,3] have been proposed for conversion between the TC and SC, and there are also a variety of products to perform different level conversion for special needs, for example: e-mail content conversion [21] , text processing, and Web Page conversion [22] , etc.

In this paper, we focus on how to design a mechanism to handle TC and SC information, in order to search data efficiently in LDAP server. The processes we have to do include: (1) the analysis of the LDAP server internal, and (2) the investigation of conversion between TC and SC. Our platform is the "OpenLDAP" system which is developed by LDAP community and is an open source [20].

2. LDAP Internal

In order to design an efficient mechanism in LDAP to perform the cross-search, we need to survey related standards and investigate the internal of LDAP server. LDAP was originally developed as a lightweight alternative to DAP [4,8,12]. The first version of LDAP was defined in X.500 Lightweight Access Protocol (RFC 1487), which was replaced by Lightweight Directory Access Protocol (RFC 1777). The latest LDAP Version 3 was defined by Lightweight Directory Access Protocol (v3) (RFC 2251) [5].

2.1 Information Model

The basic unit of information stored in the directory is called an entry. Entries represent objects of interest in the real world such as people, organizations, and so on. Each entry has a name called a distinguished name (DN) that uniquely identifies it, and it consists of a sequence of

parts called relative distinguished names (RDNs). Entries are composed of a collection of attributes that contain information about the object, and every attribute has a type and one or more values. The type of the attribute is associated with syntax which specifies what kind of values can be stored. The relationship between a directory entry and its attributes and their values is shown in figure 1. In addition to define what data can be stored as the value of an attribute, syntax of attribute also defines how those values behave during searches and other directory operations.

On the other hand, the schema of LDAP defines the type of objects that can be stored in the directory, and also lists the attributes of each object type no matter these attributes are required or optional. Each directory entry has a special attribute called "ObjectClass", and the value of it is a list of two or more schema names; that is, the schema defines what type of object(s) the entry represents.

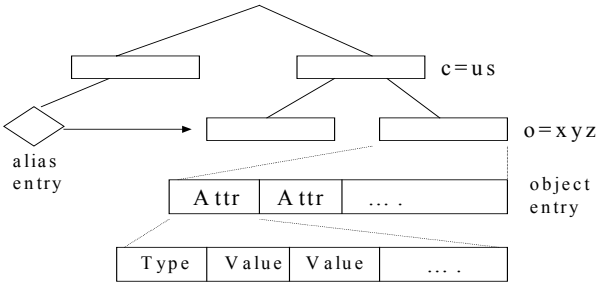

Figure 1: The architecture of one directory tree.

2.2 String Encoding

LDAP uses strings to represent data rather than complicated structured syntaxes such as ASN.1. As the LDAPv3 specifies, all data have to be represented by UTF-8 character encoding, and figure 2 illustrates the UTF-8 encoding algorithm [6,13].

Data = c	Bits	Byte 0	Byte 1	Byte 2	Byte 3
< 0x80	7	C	-	-	-
< 0x800	11	0xC0 | c >>6	0x80 | c & 0x3F	-	-
< 0x10000	16	0xE0 | c>>12	0x80 | c>>6 & 0x3F	0x80 | c & 0x3F	-
< 0x20000	21	0xF0 | c >> 18	0x80 | c>>12 & 0x3F	0x80 | c>>6 & 0x3F	0x80 | c & 0x3F

Figure 2: The UTF-8 encoding algorithm.

Note that in the UTF-8 algorithm, characters in the ASCII region (0x0000 through 0x007F) are represented as the original ASCII characters in a single byte, and the other characters in different ASCII regions are encoded to variable-length bytes.

There are several related RFC standards: (1) Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names (RFC 2253), and (2) The String Representation of LDAP Search Filters (RFC 2254) [6,7].

2.3 Search Index

Since the design of LDAP server is optimized for quick searching, the search index plays a very important role; the search performance depends on the indexing method. In order to search quickly, OpenLDAP uses additional spaces to speed up the search performance, and makes independent index for each attribute. Furthermore, LDAP supports three types of filter to search: (Equality, Approximate, Substring), and OpenLDAP also provides different type of indices for each filter.

In the kernel of OpenLDAP server, the Berkeley DB library is used to handle the search indices as folloes. The attribute prefix, value, and syntax are used to perform the MD5 hashing. Then the corresponding hashing key is produced. The OpenLDAP server stores the pairs (hashing key, entry id) in the Berkeley DB database as search index.

There are two main processes for searching data in the LDAP server:

(1) The first process is to compute the MD5 hashing key by using filter conditions (prefix + value + syntax), then retrieve one possible entry list from which each entry is checked with the filter conditions (And, Or, Not). Finally we will get the candidate list or fail to do so. Figure 3 shows the architecture of the filter ***: (ou=computer), scope is subtree.***

(2) The second process is to check each attribute of the entry in the candidate list, then get the final correct answers. However, if the search attribute does not have index for it, all entries in LDAP would be in the candidate list during process (1), and the search speed will be very slow.

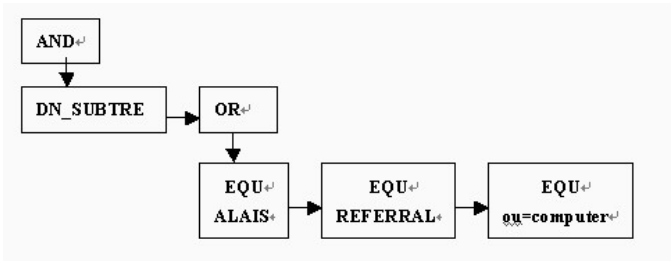

Figure 3: The architecture of the ***filter: (ou=computer).***

The design of LDAP server does not consider the properties of Chinese, especially in the search process. For example, it uses the "Soundex" method designed for western language to perform the approximate searching function. However, it is not adequate for other language. Besides, all LDAP client APIs support Unicode, but we have to convert Chinese (TC: Big5, CNS11643 or SC: GB2312) to Unicode when we call these LDAP APIs.

3. Chinese Conversion

3.1 Character Sets and Encoding

Chinese language has two different forms for most of Chinese characters: Traditional Chinese (TC) and Simplified Chinese (SC). There are a lot of differences between these two forms including character sets, encoding methods, and choice of vocabulary.

In TC, the common character sets are Big5, and CNS11643 (Chinese National Standard); the Big5 character set has its special encoding method, and the CNS11643 [9] character set uses the EUC-TW as its encoding method. On the other hand, in SC, there are some character sets, such as: GB2312, and its expanded version GBK; both two character sets use the EUC-CN as their encoding methods.

Ten years ago, we encountered great difficulties in the conversion between TC and SC, but as the Unicode Operating Platform is widely accepted, and its character set simultaneously includes the major portions of TC and SC, Unicode seems to solve the problem of simultaneously existence of TC and SC.

3.2 Conversion

As we know, SC comes from the character simplification of TC, however, there are still lots of different situations; that is, this is not a straightforward correspondence between the two forms. Some differences are as follows:

1. There are many new created SC words without corresponding TC words.
2. One SC word may maps to many TC words.
3. Although one SC word maps to many TC words, only one of them is correct in the document, depending on the context of document.

For the conversion between TC and SC, it is quite simple to convert from TC to SC, but the reverse conversion from SC to TC is relatively complicated and full of pitfalls [2]. According to different needs, the conversion can be implemented on four levels (as figure 4 shows), in increasing order of sophistication, from a simplistic code conversion that generates numerous errors, to a complex approach that takes the semantic and syntactic context into account and aims to achieve near-perfect results. Each of these levels is described below.

Level 1 **Code** Character-to-character, code-based substitution
Level 2 **Orthographic** Word-to-word, character-based conversion
Level 3 **Lexemic** Word-to-word, lexicon-based conversion
Level 4 **Contextual** Word-to-word, context-based translation

Figure 4: The four conversion levels

[Level 1 Code Conversion] character-to-character, code-based substitution mapping table

This method is an easiest, but most unreliable, way to convert between TC and SC. It uses a one-to-one mapping table to replace each single word. There are several familiar methods such as: Simplistic conversion (mapping table), Frequency-based conversion (selected from a list ordered by frequency of occurrence), Candidate-based conversion (either interactively in the user interface (UI), or as a list in brackets). The result of Code Conversion is not good enough for common applications.

[Level 2 Orthographic Conversion] word-to-word, character-based conversion

The basic conversion unit in this level is called **orthographic unit**: one word or one term (which is the meaningful combinations of words), and it is treated as a single entry in dictionaries and mapping tables. This level conversion focuses on the mapping from one SC word to many TC words, and it use some technologies to identify meaningful nouns to help the mapping correct. These processes are :

1. Segmenting the source sentence or phrase into word-units.
2. Looking up the word-units in orthographic (word-unit) mapping tables.
3. Generating the target word-unit.
4. Outputting the target word-unit in the desired encoding.

[Level 3 Lexemic Conversion] word-to-word, lexicon-based conversion

A lexeme is a basic unit of vocabulary, such as a single-character word, affix, or compound word. This level of conversion mainly focuses on the cases where SC and TC have entirely different words for the same concept. For example, "计算机" (SC)-"電腦" (TC), "信息" (SC)-"資訊" (TC), "网络" (SC)-"網路" (TC), etc.

On the other hand, we could regard this level of conversion as one kind of translation between two languages, and it is similar to the word-units used in orthographic conversion, but the term *lexeme* is used here to emphasize the semantic nature of the conversion process.

The main processes are like the orthographic conversion, but the mapping tables must map one lexeme to another at a semantic level. The segmentation algorithm must be sophisticated enough to identify proper nouns, since the choice of target term depends on whether the lexeme is a proper noun or not.

[Level 4 Contextual Conversion] word-to-word, context-based translation

In this level, the semantic and syntactic context must be analyzed to correctly convert certain ambiguous lexemes that map to multiple target lexemes. This level of conversion focuses on the phases both existing in TC and SC, but having entirely different meaning. For example, in SC "文件" (which means "data file") should map to "檔案" in TC, and can not map to "文件" (which means "document") in TC. It is very complicated to achieve this level of conversion, so we don't implement it.

3.3 Segmentation

Because the meaningful units "words" in Chinese are not delimited by spaces, tokenizing is considerable more difficult than for western languages. The segmentation of Chinese text is the basic technique to process Chinese, and it had been applied to many application areas: information retrieval, text processing, machine translation, text-to-speech, linguistic analysis, etc.

Furthermore, on the conversion from SC to TC, the quality of segmentation has great influence on the accuracy of conversion. In fact, we have to segment TC and SC respectively because of the different vocabulary sets used in them.

There are a lot of researches [1,2,9] on the segmentation in Chinese, such as: lexical method, statistical method, and hybrid of statistical and lexical method. The lexical method and statistical method have their different advantages and could be applied for different needs.

In this paper, we focus on the LDAP server, and therefore, the search speed is our main concern. The segmentation algorithm we choose to implement is the updated "Maximum Matching Algorithm", and on the other hand, we also strengthen segmentation to identify people names [19].

4. System Design

In this section, we describe our system architecture and main concerns.

4.1 Definition

In the LDAP server, all data are stored in the attributes of entry. However, Chinese words could be used in the DN to identify one entry. In this paper we focus on the search function for data, therefore, our topic focuses only on the attribute value. On the other hand, according to the search types in LDAP, we mainly divide them into two types: **equality search** and **approximate search**, and we also implemented different Chinese conversion level for these two kinds of search. We apply the level1 and level2 conversions to equality search, and level3 conversions to approximate search.

We have defined the following functions:

(L is the input string, n is the conversion level.)

(1) Convert-S-T(n, L): convert L from SC to TC using level n.

(2) Convert-T-S(n, L): convert L from TC to SC using level n.

(3) Basis(L): update L to the basis word.

In the third function above, we change the original string L into its basis word. It is not only for the different forms (TC and SC) but also for the same forms. For example, the two words '台' and '臺' are the same in TC, they should be regarded as the same word in our system.

4.2 The language form

The LDAP server provides several search APIs for users to send suitable search filters for their needs. However, the filter string is composed of Unicode characters, so the system could not identify what Chinese form user use, and what display mode user want. Besides, the segmentation process depends on the input form, and our system want to provide suitable display mode to user, so we first have to identify the input form (which is either TC or SC).

In our system, we provide two approaches to identify the input form; one is user specified and the other is program automatically identified.

1. User specified:

To concern about the consistency with the existing systems, we do not change any APIs and filter format, but directly extend the filter to specify the input form and display mode. The filter also follows current LDAP protocols. We describe the extension approach below.

If f is the original filter, we extend the filter to f1 by inserting one "**OR**" operation; that is, we use additional filter element (c=xx) in OR operation to specify the input form. For example, we want to specify that the input form is TC and the output form is the original form of output document, then the filter will be (OR (c=T) f), which is in prefix style defined in the LDAP. The related symbols are as follows.

Output form Input	Document form	TC	SC
TC in filter	T	TT	TS
SC in filter	S	TS	SS

On the other hand, we also have to change the filter parser, so that it could restore the filter f1 to original filter f, and get the input and output forms user specified.

2. Program automatically identified:

We use the n-grams technique to automatically identify the input form, and perform two parses (one is n=1, and the other is n=2) to compute the matching score. We define the score function to store the score of different n-grams and forms, for example: score(1,t) is the matching score of TC when applying 1-gram to detect.

(1) for n=1, we created in advance one mapping table [15,16,17] which stores the forms (TC or SC or both) of all Unicode Chinese words in order to lookup each 1-gram of filter string from it; if the 1-gram exists in TC, the score(1,t) adds 1, and if the 1-gram exists in SC, the score(1,s) adds 1, and if both of TC and SC have the 1-gram, both of the two scores add 1 respectively.

(2) for n=2, we use separate thesauruses for TC and SC to lookup each 2-gram of filter string; if the 2-gram exists in TC, the score(2,t) adds 1, and if the 2-gram exists in SC, the score(2,s) adds 1, and if both of TC and SC have the 2-gram, both of the two scores add 1 respectively.

Finally, we sum up the two scores (score(x) = score(1,x) + score(2,x)), and choose the bigger one as the input and output forms; when the two scores equal, we choose SC form.

4.3 Search Processes

According to the LDAP RFCs, one search process contains three parameters: the first one is "base dn" which is the starting node to search, the second one is scope (base, or one level, or subtree) which defines how to search, and the last one is filter string, such as "name=孫中山". The new Chinese search processes in LDAP server are described as follows.

1. Identification of input and output forms.
2. Parse and Rebuild Filter (this is for approximate search, we discuss this in section 4.5).
3. Retrieve all candidate entries by each filter condition.
4. Perform logical operation for candidate entries.
5. Detailed match all candidate entries.
6. Conversion and output the answers.

4.4 Equality Search

[Indexing processes]

On the other hand, LDAP server makes individual index for each filter type. Therefore, we design the indexing processes for equality search:

1. Identification of input and output forms.
2. Convert(L): convert each word of L into its base word.
3. Convert-T-S(1,L): convert L to SC by using level 1 conversion.
4. Perform MD5 hashing [14] on L as the original LDAP server does.

[Searching processes]

We divide the search processes into filter and match processes. The filter processes are similar to the indexing processes, and we use the same functions Convert(L) and Convert-T-S(1,L) to convert the input search filter, then match it by the indices created beforehand, and finally get the candidate entries. However, there are still some errors:

1. Because in the searching processes all words are converted to SC form, one TC word may match one error TC word which is converted to the same SC word, for example: '干', '幹', and '乾' in TC are all mapped to '干' in SC. Against this situation, the following match processes have to check the input form, and it must satisfy Basis(L1) = Basis(L2) when both words are TC.
2. When we search one term having two or more words, and perform the word-level conversion to SC, this may cause some errors, for example: the answer may be "干淨", or "乾淨", but it can't be "幹淨".

Additionally, the MD5 hashing in LDAP server also causes some errors, match processes have to perform the real complex matching. Finally, the system needs to convert the answers into output form. The function Convert-T-S(1,L) could easily convert TC to SC by merely table mapping, however, the inverse process is not quite easy, so the system has to perform the level2 Convert-S-T(2,L). We use the segmentation technique based on SC thesaurus to segment the data, then lookup each term in order to retrieve corresponding TC term, and finally reform the output answers.

4.5 Approximate Search

In the approximate search, we focus on two parts: one is approximation between different words having the same meaning in TC and SC, and the other is approximation of pronunciation.

[Approximation between TC and SC]

We perform the level3 conversion between TC and SC, and directly use the preceding indices and equality processes to achieve the translation. The main work is the "Parse and Rebuild Filter", which is to segment filter string, then retrieve all corresponding words. However, we use OR operator to rebuild the filter, for example: (1) original filter (AND F1 F2), rebuild filter (AND (OR F1 F1') (OR F2 F2')), where F1' and F2' are the words converted from F1 and F2. (2) search("資訊") is converted to (OR "資訊" "信息"), then we perform the equality search to get the answers.

[Approximation of pronunciation]

Because the pronunciation differs from the other properties, we need to make new index for this approximation. The indexing processes are listed as follows.

1. Identification of input and output forms.
2. Segment search filter and convert each of terms to its sound basic word.
3. Use the original MD5 hashing to process.

The key point of these processes is how to find the correct sound basis words for those words having several pronunciations. We segment filter string first, then lookup each term from one table [17] to get its sound basic word. For example, "一", "依", and "伊" have the same sound basic word "一".

5 Conclusion

In this paper, we implemented the mechanism for cross-search between TC and SC in OpenLDAP server, however, there are still some problems needed to be studied in advance:

1. Word Segmentation and Named Entity Extraction: In practice, LDAP server always has a great amount of named entities, and therefore, we need better techniques of segmentation and named entity extraction to achieve acceptable outcomes. On the other hand, LDAP server is optimized for quick searching, and there are many techniques and algorithms for segmentation, how to balance between the search time and correctness is a difficult task.
2. In the segmentation and conversion between TC and SC, we need separate thesauruses [15,16,17] for them. However, the management and maintenance of these thesauruses costs a lot and is not easy. We need some automatic mechanisms to handle it, for example: robots which could collect information automatically from the Internet.
3. According to RFC2253, distinguished name (DN) could be encoded by UTF-8, the conversion problem may also happen. However, the distinguished name is related to entire directory tree architecture, this problem needs to be studied.
4. Some Chinese words in CNS11643 are not contained in Unicode, and we also need some means to deal with it.

References

[1] Thomas EMERSON. Segmenting Chinese in Unicode. In *Proceedings of the 16th International Unicode Conference, Amsterdam, The Netherlands, March 2000.*

[2] Jack Halpern, and Jouni Kerman. The Pitfalls and Complexities of Chinese to Chinese Conversion. In *Proceedings of the 14th International Unicode Conference, Cambridge, Massachusetts, March 1999.*

[3] Liu, Shing-Huan. An automatic translator between Traditional Chinese and Simplified Chinese in Unicode. In *Porceedings of the 7th International Unicode Conference, San Jose, California, September 1995.*

[4] An LDAP Roadmap website: http://www.kingsmountain.com/ldapRoadmap.shtml.

[5] Lightweight Directory Access Protocol (v1) RFC1487 (v2) RFC1777 (v3) RFC2251.

[6] LDAP v3 UTF-8 String Representation of Distinguished Names RFC2253.

[7] The String Representation of LDAP Search Filters RFC2254.

[8] Tim Hows, Mark Smith. (Book) LDAP: Programming Directory-Enabled Applications With Lightweight Directory Access Protocol. March 1997.

[9] CJKV Information Processing. Web site: (http://www.oreilly.com/people/authors/lunde/cjkv-ip.html).

[10] Hih-Hao Tsai, MMSEG: A Word Identification System for Mandarin Chinese Text Based on Two Variants of the Maximum Matching Algorithm. Web site: (http://www.geocities.com/hao510/mmseg/).

[11] Hsin-Hsi Chen, Yung-Wei Ding, Shih-Chung Tsai and Guo-Wei Bian. Description of the NTU system used for MET2. In *Proceedings of 7th Message Understanding Conference, Fairfax, VA, 29 April - 1 May, 1998.*

[12] Understanding LDAP (IBM redbook). Web site: (http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg244986.html?Open).

[13] Unicode Transformation Formats. Web site: (http://czyborra.com/utf).

[14] MD5 Hash Algorithm (v1). Web site: (http://www.w3.org/TR/1998/REC-DSig-label/MD5-1_0).

[15] Simplified Chinese thesaurus. The "纵横" input method. Web site: (www.zhhz.suda.edu.cn).

[16] Traditional Chinese thesaurus. The "纵横" input method, and EZ input method (http://ezinput.100k.com.tw/) , and Chih-Hao Tsai's Technology Page (http://www.geocities.com/hao510/).

[17] Thesauruses of basic word, and sound basic word, and Unicode table for TC and SC. CJKV Information Processing. Web site: (http://www.oreilly.com/people/authors/lunde/cjkv-ip.html).

[18] Guo-Wei Bian and Hsin-Hsi Chen (2000). "Cross Language Information Access to Multilingual Collections on the Internet." In *Journal of American Society for Information Science, Special Issue on Digital Libraries, 51(3), 2000, 281-296*.

[19] Hsin-Hsi Chen, Yung-Wei Ding and Shih-Chung Tsai (1998). "Named Entity Extraction for Information Retrieval." In *Computer Processing of Oriental Languages, Special Issue on Information Retrieval on Oriental Languages, 12(1), 1998, 75-85*.

[20] The OpenLDAP web site: (http://www.openldap.org/).

[21] Email content conversion. The product "兩岸文件通". Web site: (http://www.winperturn.com.tw/big5/html/body_page115943.html).

[22] Web Page conversion. The product "龍之旅". Web site: (http://input.foruto.com/ cccl/cccl_article019.htm).