NTCIR-3 CLIR Experiments at Osaka Kyoiku University —Comparison of Gram-based Indices—

Takashi SATO Koto HAN
Osaka Kyoiku University
4-698-1 Asahigaoka, Kashiwara, Osaka, Japan
sato@cc.osaka-kyoiku.ac.jp

Abstract

Long gram-based indices are experimented at NTCIR-3 CLIR task. To make gram-based indices, no analyses such as morphological ones are required. Indices in three languages (i.e. Japanese, English and Chinese) are made at this task. They are quite different in some point. The difference of index overhead comes from the difference of character code for example.

Keywords: *gram-based index, gram coding, multi-lingual, NTCIR*

1 Introduction

We participated one of traditional NTCIR task, CLIR. We make indices of three languages (i.e. Japanese, English and Japanese). Index overhead (i.e. the ratio of index size to corpus) is quite different language by language.

2 Index for arbitrary string search

N-gram [1-4] and suffix [5-7] array are known as index structures, which enable arbitrary string search.

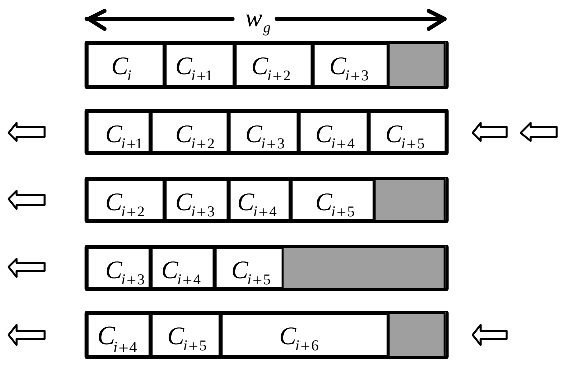

Figure 1. Example of gram coding

N-gram stores strings (character sequences), which start every character in a text. The length of strings is less than 3 (such as 1-gram or 2-gram) in common cases. We realized longer than 4grams in average by encoding grams in w_g byte. w_g is set 5 or 6 in most cases. At first, characters are coded in varying length bit in accordance with their frequency of appearance. Then they are staffed within w_g. Figure 1 shows an example gram coding.

3 J-J subtask

3.1 Index making

We computed grams, document by document. We made two indices as inverted files of gram. One is made from *headline* tag and the other is *text* tag. During index making, we sort grams. We also made wide range map of gram, which are put in main memory when we search grams.

At this subtask, gram length, the number of characters in a gram, ranges from 2 to 6 and 4.3 in average. And they are coded into w_g=6 byte. That is, a gram has almost same length as 3-gram if not coded. Computer used is Compaq DS-20 (64bit architecture, 4GB main memory).

Table 1 shows the size of corpus, extracted tag fields and two indices, which are made from headline and text tag field. Index size overhead against extracted tag fields is 140%. Table 2 shows time to make indices.

Table 1. Size of corpus, tag fields and indices (J-J)

corpus	301Mbyte
<headline> tag <text> tag	28.7Mbyte 259Mbyte
<headline> index <test> index	43.1Mbyte 361Mbyte

© 2003 National Institute of Informatics

Table 2. Time to make indices (J-J)

<headline>	3.29min
<text>	27.7min
total	31.0min

3.2 Query making

We extract query words using morphological analysis from TITLE and DESC tags in given 50 topics (001-050). Compound words are segmented in words, and then all possible combinations of words are made of a compound word.

3.3 Index search and document ranking

Our index has tree structure, which has sorted gram and wide range map of them. So, not only query words whose length is equal to gram length, but also shorter or longer words can be searched efficiently. When we search a longer word, every gram in the words is searched. Then retrieved sets of document numbers are intersected.

Fromset of retrieved documents for query words, we compute tf-idf and similarity using probabilistic model [8] for document ranking. Table 3 shows index search time.

Table 3. Index search time (J-J)

index	<headline>	<text>
word searched	203	327
time to search	3.02sec	20.8sec
average time per word	18.1msec	63.4msec
median time per word	11.7msec	25.4msec

4 E-E subtask

We computed grams as **3.1**. At this subtask, gram length ranges from 4 to 13 and 8.1 in average. And they are coded into w_g=6 byte.

Table 4 shows the size of corpus, extracted tag fields and two indices. Index size overhead against extracted tag fields is 173%. Table 5 shows time to make indices.

Table 4. Size of corpus, tag fields and indices (E-E)

corpus	60.7Mbyte
<headline> tag	0.96Mbyte
<text> tag	55.9Mbyte
<headline> index	3.42Mbyte
<test> index	95.1Mbyte

Table 5. Time to make indices (E-E)

<headline>	0.71min
<text>	6.27min
total	6.98min

5 J-C subtask

We computed grams as **3.1**. They are coded into w_g=6 byte. At this task, no specific tag fields are extracted, so grams are made of entire corpus. Table 6 shows the size of corpus and an index. Index size overhead against corpus is 192%.

Table 6. Size of corpus and index (J-C)

corpus	212Mbyte
index	407Mbyte

This is the only cross-lingual subtask we participated. We made Chinese indices from Chinese corpus. In order to make search keys against corpus, we extracted words and compound words from Japanese topics. Next we translated them into Chinese words. Then we expanded them by synonym in turn. We did not expanded words from topics by synonym in Japanese before translation because this approach was not able to get good results.

6 Discussions

The size of gram-based index depends on many parameters. For example, they include the length (w_g) in which grams are coded, pointer size, the number of byte for document numbering. Moreover the number of grams themselves differs at most twice by whether characters in documents are 1byte or 2byte even if documents are in the same length. Whereas we were able to make long gram-based indices, which are less than 200% overhead, at all subtasks we participated.

7 Conclusions

We participated one of traditional NTCIR task, CLIR. We make indices of three languages. Index overhead is quite different language by language. Though it depends of many parameters. Whereas we were able to make long gram-based indices, which are less than 200% overhead, at all subtasks we participated.

References

[1] Sato, T., Fast full test search with free word using TS-file, *Proc. 19th ACM SIGIR Conf.*, p.342 (1996).

[2] Sato, T., Fast full test retrieval using gram based tree structure, *Proc. ICCPOL '97*, Vol.~2, pp. 572--577 (1997).

[3] Sato, T. *et al.*, Gram based full test search system and its application, IPSJ SIG Notes, 98-DBS-114-2 (1998).

[4] Sato, T, *et al.*, NTCIR-3 PAT experiments at Osaka Kyoiku university, *in this proceedings*.

[5] Gonnet, G., Baeza-Yates, R. and Snider, T., New Indices for Text: Pat Trees, in *Information Retrieval: Data Structure & Algorithms* chapter 5, Frakes, W. and Baeza-Yates, R. Ed., pp. 66-82 (1992).

[6] Shang, H. and Merrett T., Trees for approximate string matching, *IEEE Trans. Knowledge and Data Eng.*, Vol. 8, No. 4, pp. 540-547 (1996).

[7] Yamashita, T., Fujio M. and Matsumoto Y., Language Independent Tools for Natural Language, *Proc. 18th ICCPOL*, pp.237-240 (1999).

[8] Robertson, S.E. and Walker, S., Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval, *Proc. 17th Int. Conf. Research and Development in Information Retrieval*, pp. 232-241 (1994).