NTCIR Experiments at Matsushita: Ad-hoc and CLIR Task

SATO, Mitsuhiro msato@trl.mei.co.jp
ITO, Hayashi haito@trl.mei.co.jp
NOGUCHI, Naohiko noguchi@trl.mei.co.jp

Multimedia Systems Research Laboratory, Matsushita Electric Industrial Co., Ltd.
4-5-15, Higashi-Shinagawa, Shinagawa-ku, Tokyo 140-8632 JAPAN
+81-3-5460-2744

ABSTRACT

In this paper, we describe our experimental results for NTCIR ad-hoc task and CLIR task utilizing *MEISTER*, a group of software libraries to construct high performance full-text search applications, designed especially for agglutinative language text such as Japanese. The results show that the basic facilities of *MEISTER* are in fact effective for ad-hoc task, and supportive for CLIR task. We also show that a kind of corpus-based approach adopted here for CLIR task is helpful to cover the weakness of translation dictionaries and MT systems.

Keywords

n-gram-based index, word-based index, full-text search system, information retrieval model, term extraction, similarity measure, Boolean operator, coordination level scoring, query term translation, cross-language information retrieval

1. INTRODUCTION

In recent years, the n-gram-based indexing method has been broadly used in full-text search systems for agglutinative language, such as Japanese[1][13][17] and Korean[12], because it does not need the maintenance of dictionaries nor word-segmentation processes, either of those is an expensive process. Another advantage of the n-gram-based indexing is that one can search text not only with registered words (dictionary terms), but also with arbitrary strings, which might be un-registered words (non-dictionary terms). So it is free from the problems of word-based indexing, such as the problem of unknown words or the problem of segmentation failure.

But the n-gram-based indexing can not be directly applicable to other types of request such as "extract significant terms from search results", or "expand queries with related terms", because it does not recognize "words" in principle. To handle those kinds of request, the n-gram-based indexing has to be combined with other sorts of devices.

We have developed a new type of word-based indexing method, *maximal word indexing*[11][16], which enjoys both merits of the n-gram-based and of the word-based indexing. The *maximal word indexing* method uses a dictionary, which merely defines "words" as character strings, and only choose *maximal* occurrences of words in the text as indexing units. Here, a single occurrence of some word in the text is defined "*maximal*" if and only if it is not included by any other occurrences of other words in the dictionary. This simple indexing mechanism enables us to search the full-text with arbitrary strings at high speed, with relatively compact size index, compared to the n-gram-based indexing[10].

Moreover, since the *maximal word indexing* is word-based, it can be a uniform platform to handle those kinds of request described again below:

・Rank the documents according to the relevance to a query.

・Extract significant terms from documents or query sentences.

・Pick up some related (similar) terms to the original query terms.

Based on this new indexing method, we have developed a group of software libraries called *MEISTER*, which realizes all the above mentioned functions[15]. *MEISTER* comprises several software modules, for example, an indexing module, a searching and ranking module, a term extraction module, a term similarity calculation module, and so on.

For the experiments on NTCIR tasks, we constructed a full-text search system for the test collection documents combining several modules of *MEISTER*. Besides, query construction and query expansion processes were also implemented utilizing *MEISTER*. We used an EDR-based dictionary as a sole source of linguistic information. We did not use any linguistic processing such as morphological analysis nor syntactic analysis.

In section 2, we introduce *MEISTER* briefly, its IR model and its basic facilities we are going to utilize in our experiments are presented. Section 3 describes the experiments on ad-hoc task. As for the ad-hoc task, we put our focus on the evaluation of the basic ranking facilities of *MEISTER*. Section 4 describes the experiments on CLIR task. As for the CLIR task, the term similarity calculation module is applied to parallel (bilingual) corpora for selecting translated terms. Section 5 summarizes the paper.

2. A BRIEF DESCRIPTION OF *MEISTER*

2.1 IR Model

As described, *MEISTER* is designed as a software platform so as to be easily applied to various and broad full-text search application systems. So the conventional search facilities such as fast string search and Boolean search need to be supplied in order to satisfy expertise users. Here, string search facility is fairly important especially for Japanese text, because non-dictionary terms, such as newly introduced terms or compound

terms could be typical query terms. Moreover, the document ranking facility need to be supplied as well to satisfy novice users. Here again, arbitrary strings should be handled as query terms by this document ranking.

Thus, the IR model of *MEISTER* is a kind of mixture of a Boolean model and a vector space model. *MEISTER* can handle Boolean queries composed of arbitrary strings which could often be non-dictionary terms, then fixes the result set of documents which satisfies input Boolean query, and ranks each document in the result set by a certain similarity measure. In both processes, *MEISTER* can handle arbitrary strings, as well as dictionary terms. As for the similarity measure, *MEISTER* leaves it to be freely defined by users. Currently and for this experiments, we use a similarity measure represented in the formula (1).

$$(1)\quad Sim(d_j, q) = C_1 \sum_{q_i \in Q} w_i \cdot |tf_{ij}| \cdot idf_i + C_2 M_j$$

Here, q is a query, d_j is a document, Q is a set of query terms included in the query q, tf_{ij} is the occurrence frequency of the term q_i in the document d_j ($|tf_{ij}|$ means some normalization is done on tf_{ij}), idf_i is the inverse document frequency of q_i computed by the formula (2), w_i is a given weight of q_i, and C_1, C_2 are constants.

$$(2)\quad idf_i = \log(\frac{N}{df_i}) + 1$$

(N is the number of the whole documents, df_i is the document frequency of q_i)

The first component of the formula (1) can be thought as the inner product of a document vector and a query vector. The second component M_j is the number of co-occurring terms among Q in the document d_j, which represents a kind of coordination level information. We added M_j here in the formula (1) because of the facts that coordination level information is useful especially for short queries[6][19][22], that most of real user queries are composed of a few words, and that users tend to read just a few top-ranked documents. The default setting of *MEISTER*, which takes these facts seriously, makes C_2 considerably bigger than C_1[9][14]. That is, C_2 has to satisfy the condition (3).

$$(3)\quad C_2 \geq \max_{d_j \in D} \{C_1 \sum_{q_i \in Q} w_i \cdot |tf_{ij}| \cdot idf_i\}$$

Here, D is the set of the whole documents. The condition (3) means that the result set of documents are ranked by coordination level information first, producing several layers of score, then the documents in each layer are ranked by a kind of *tfidf* measure. We will call this default setting of similarity measure "Coordination Level Scoring", ***CLS*** for short, and call the setting where C_2 = 0 "***naïve tfidf***" for the ease of following discussion.

2.2 Basic Facilities of *MEISTER* Used in the Experiments

*2.2.1 **ADD** Operator*

Conventional Boolean systems have set-theoretic concepts as the conceptual interface between users and systems. Users are conscious of a set of retrieved documents constructing structured queries. For the systems of mixture model like the one we are proposing, users might imagine some query terms that should contribute only to ranking the documents, but should not contribute to fixing the result set of documents. *MEISTER* has a special operator other than Boolean operators, ***ADD*** operator, handling these situations. Users can input queries like (4).

(4) (Matsushita ***OR*** SONY) ***ADD*** DVD

The result set of documents is either a document including "Matushita", or a document including "SONY", or a document including both, ranked by similarity calculation using all of the terms here, "Matsushita", "SONY", and "DVD". That means while "DVD" is neither necessary nor sufficient to fix the result set, documents in which "DVD" occurs would be ranked higher. That is, the ***ADD*** operator only adds information for ranking.

*2.2.2 Treating synonyms and the like properly: **Synonym** Operator*

Expanding conceptually one term into several linguistic expressions (synonyms, acronyms, abbreviations, etc.) is an important feature for IR systems in general. Usually, synonym dictionaries or thesaurus dictionaries are used for expanding such terms, and the expanded expressions are added to the query as disjunctive components. That is, they are ***OR***-ed to the query. But simple addition of these expressions might results in the problematic situations as described below:

- Frequency data of those expressions are in a sense not accurate because all the expanded expressions should be treated as same single term. This causes over (under)-estimation for the similarity calculation regarding such expressions. (Inverse document frequencies of such expressions are typical examples.)
- When there are several query terms, the relative importance of these terms are influenced by the number of expanded expressions each of these terms has. And the number of expanded expressions for each query term is heavily dependent on a dictionary.

MEISTER has another operator, ***Synonym*** operator, handling these situations. Users can specify queries like (5).

(5) Matsushita**,** MEI**,** Panasonic ***OR*** SONY

The ***Synonym*** operator represented by "**,**" acts as an ***OR*** operator in fixing the result set. But in similarity calculation, it acts as an unifying operator for those concatenated expressions. For example, "Matsushita", "MEI", and "Panasonic" are treated as one term in (5), that is, term frequencies and document frequencies for those terms are accumulated into one statistics to compute the similarity of each document. This makes relative importance of "Matsushita" and "SONY" stay equal.

2.2.3 Extracting Terms

MEISTER also has term extraction facilities utilizing word-based indices. It can extract terms from documents or from sentences specified by users. This is a key facility to realize various processes, such as query construction, query expansion (relevance feedback or pseudo feedback), similar document retrieval, or automatic summarization of result documents.

In our experiments, we used this facility for query construction and query expansion processes.

2.2.3.1 Extracting Terms from Documents

When a set of documents S in the whole documents D is specified, *MEISTER* computes the weight of each term occurring in S and selects heavily weighted terms. The term weighting formula can be user-defined, but conceptually, it can be represented as in the formula (6).

$$(6)\quad w_i = g(\sum_{d_j \in S} tf_{ij}, idf_i, M_i)$$

The formula (6) means that the weight w_i of the term t_i is computed with its occurrence frequency in S, its inverse document frequency, and M_i, which is the number of documents among S in which term t_i occurs. M_i, represents a kind of coordination level information relating to the term t_i and a set of documents S. The default setting of *MEISTER*, again taking this coordination level information seriously, adopts a weighting formula shown in (7).

$$(7)\quad w_i = (K_i \sum_{d_j \in S} tf_{ij}) \cdot idf_i \cdot M_i$$

$$K_i = (2 - \frac{df_i}{\sum_j tf_{ij}})$$

In the formula (7), a kind of *tfidf* weight is multiplied by M_i. N_i is a normalization factor for term frequency chosen by our previous experiments.

After term weights are computed, *MEISTER* then selects terms in order of their weight. Users can specify how many terms should be selected, and also can put some conditions on selecting terms. For instance, threshold values (either upper and lower threshold) for *idf* can be specified, and the selecting conditions about extension relation between terms can be set. Also, user-defined stop words can be given to this process.

2.2.3.2 Extracting Terms from Sentences

When sentences are specified, *MEISTER* extracts terms using some dictionary. Here again, *MEISTER* does not concern any linguistically sensitive processes, such as morphological analysis or syntactic analysis. Besides, non-dictionary terms are also extracted using character type information, because they could represent important concepts in user queries.

If necessary, extracted terms can be weighted by the formula (6), so as to be selected.

2.2.4 Term Similarity Calculation

Other than term extraction, *MEISTER* has another term relating facility, term similarity calculation facility. Term similarity is calculated based on occurrence patterns of each term at a document level. In other words, the similarity measure among terms corresponds to the degree of document level collocation in particular document set. *MEISTER* uses *term vector* to represent the occurrence pattern of each term, and calculates similarities between terms using this vector representation.

A *term vector* has N dimensions, where N is the number of documents in a target document set. The d-th value of the vector of the term t_i is determined by *tfidf* weight of the term t_i in the d-th document. Similarity between two terms t_i and t_j is calculated using one of the following three formulae, or their combination[5].

(8) Inner Product $$IP_{ij} = \sum_{d \in D} S_{id} \cdot S_{jd}$$

S_{id} : d-th value of the term vector t_i

(9) Mutual Information $$MI_{ij} = \log_2(\frac{P(i,j)}{P(i) \cdot P(j)})$$

$P(i)$: occurrence probability of the term t_i

(10) t-score $$TS_{ij} = \frac{P(i,j) - P(i) \cdot P(j)}{\sqrt{P(i,j)/N}}$$

N : number of the whole documents

We have tried several combination of these formulae, and used following formula as the default setting.

$$(11)\quad Sim(t_i, t_j) = IP_{ij} \cdot TS_{ij}$$

To make this facility practical, *MEISTER* builds a *term-document matrix* in advance, which comprises the set of all the term vector occurring in the whole documents. Once the term-document matrix has been built, it can calculate similarity of every term pair in the matrix efficiently.

However, there are only dictionary terms in the matrix, thus it cannot calculate similarity of a term pair if one or both terms are non-dictionary terms. This is often problematic, because non-dictionary terms sometimes represent important concepts. To solve this problem, *MEISTER* employs a dynamic approach as well. That is, a term vector for a non-dictionary term can be constructed on-demand using its frequency data, which is obtained by searching an index. As mentioned before, *MEISTER* has string search facility for arbitrary strings, so in its dynamic approach, it can calculate similarity of any string pair, either of which could be a non-dictionary term, as far as both of which occur in the documents.

Using term similarity calculation, *MEISTER* can select similar (or collocated) terms in order of their similarity.

In our experiments, term similarity calculation facility was used for query expansion in ad-hoc task, and for translated term selection in CLIR task.

3. AD-HOC TASK

As for the ad-hoc task, we put our focus on the evaluation of the basic ranking facilities of *MEISTER*. Document indexing was done by the *maximal word* indexing method, then its term extraction facilities were used in query construction, and in searching and ranking the documents, several parameter settings were tried to evaluate each facilities.

In 3.1, we describe our query construction process. We considered characteristics of each field in the topic of NTCIR test collection and designed suitable way of constructing for each field. Then we show some results of the experiments on the basic ranking facilities of *MEISTER* from 3.2 through 3.5. To make clear the effect of each facility, we compared the result of a run using each facility to the result of a base run one by one. Finally in 3.6, the result of our formal runs are presented.

3.1 Query Construction

Since "the <DESCRIPTION> field only" run is obligatory in the workshop, we first extracted terms from this field. The <DESCRIPTION> field is describing users' information need in several sentences, thus we used *MEISTER*'s term extraction facility mentioned in 2.2.3.2. The <TITLE> field expresses a main concept of users' need in simple linguistic form, often in a form of a single phrase or in one compound term (which is sometimes a specific technical term). So we used this field as one query term. As for terms in the <CONCEPT> fields, we thought them expressing relevant concepts for retrieval, thus we simply used these terms as query terms. We did not use the <NARRATIVE> field because we thought it gave information for relevance assessment rather than for retrieval requirement.

In term extraction from the <DESCRIPTION> field, both dictionary terms and non-dictionary terms were extracted. As described in 2.2.3.2, *MEISTER* can extract non-dictionary terms using character type information, so it was often the case that some dictionary terms were overlapped or included by non-dictionary terms. For example, from topic 0032, "分散ネットワーク環境" (distributed network environment) is extracted as a non-dictionary term and "分散ネットワーク" (distributed network), "環境" (environment) are extracted as "overlapped" dictionary terms. We believe that those cases where a set of query terms includes not only short unit terms ("環境") but also relatively long compound terms ("分散ネットワーク環境") would have good influence to the retrieval effectiveness.

The number of terms to extract is controllable in *MEISTER*. We set this number at most four for dictionary terms and for non-dictionary terms respectively, considering the result of the preliminary run. In the formal run, the average numbers of extracted dictionary terms and non-dictionary terms were 3.4 and 0.6 respectively. We used the default setting for the condition of selecting terms, such as the threshold values for *idf*, the conditions on extension relation between terms. We also used 16 stop words which were selected from the result of preliminary run.

In the term extraction process, all the extracted dictionary terms were weighted with the formula (7) shown in 2.2.3.1 to be selected. We used these weights as query term weights. For the extracted non-dictionary terms, the terms from the <TITLE> field, and the terms from the <CONCEPT> field, we assigned the maximum weight of the extracted dictionary terms.

Finally, all the weighted query terms are ***OR***-ed to construct weighted structured query, which would be an input to searching and ranking processes of *MEISTER*. We did not use any other Boolean operator, ***AND*** nor ***NOT***, because these operators are too restrictive in fixing the result set to use appropriately in a fully automatic query construction process. Instead, we used ***ADD*** operator and ***Synonym*** operator in the following experiments which we believe have relatively weak but good influence to the retrieval effectiveness, when properly used. An example of constructed query is shown below.

TOPIC:0036
DESCRIPTION:モバイル環境におけるグループウェアの問題点とは何か。
TITLE:グループウェア
J.CONCEPT:携帯端末,モバイル通信
J.CONCEPT:グループウェア,共同作業,コラボレーション
J.CONCEPT: Personal Terminal, Mobile Communication
J.CONCEPT: Groupware, Collaborative Work, Collaboration
extracted terms from DESCRIPTION: モバイル環境,モバイル,
グループウェア,問題点,環境
extracted terms from TITLE: グループウェア
extracted terms from CONCEPT: 携帯端末,モバイル通信,
グループウェア,共同作業,コラボレーション,
Personal Terminal, Mobile Communication,
Groupware, Collaborative Work, Collaboration
constructed query:
モバイル環境 ***OR*** モバイル ***OR*** グループウェア ***OR*** 問題点
OR 環境 ***OR*** グループウェア
OR 携帯端末 ***OR*** モバイル通信 ***OR*** グループウェア
OR 共同作業 ***OR*** コラボレーション
OR "Personal Terminal" ***OR*** "Mobile Communication"
OR "Groupware" ***OR*** "Collaborative Work"
OR "Collaboration"

The term weights are omitted for convenience. Note that a term from the <TITLE> field, "グループウェア", is taken as one term. The duplicates of terms are not eliminated, simply taken as isolated terms.

3.2 Effect of *ADD* Operator

We first experimented on newly introduced operators in *MEISTER*, ***ADD*** operator and ***Synonym*** operator. As described in 2.2.1, ***ADD*** operator can modify the ranking of the retrieved documents without changing the number of all the retrieved documents. So when a term is ***ADD***-ed to a query instead of ***OR***-ed to it, some false drops could be avoidable. But, to the contrary, recall rate could be getting worse when terms are improperly ***ADD***-ed to a query.

As for the NTCIR topics, we observed that while the <DESCRIPTION> field had key concepts to fix the result set, the <CONCEPT> field sometimes merely had supportive concepts for users' need. So we made a comparison between two types of query constructions, one of which only uses ***OR*** operator, and another uses ***ADD*** operator for terms from the fields other than the <DESCRIPTION> field to make clear the effect of ***ADD*** operator.

Table 3-1 shows the result. Here, DTC indicates that all the query terms from the <DESCRIPTION>, the <TITLE> and the <CONCEPT> fields are ***OR***-ed to construct queries, D-ATC indicates that only the query terms from the <DESCRIPTION> field are ***OR***-ed and the query terms from the <TITLE> field and the <CONCEPT> field are ***ADD***-ed to queries. As for the ranking, ***naïve tfidf*** setting was used for the similarity calculation. An example of a constructed query for D-ATC is shown below (for topic 0036).

constructed query for D-ATC:
モバイル環境 ***OR*** モバイル ***OR*** グループウェア ***OR*** 問題点
OR 環境 ***ADD*** グループウェア ***ADD*** 携帯端末
ADD モバイル通信 ***ADD*** グループウェア
ADD 共同作業 ***ADD*** コラボレーション
ADD "Personal Terminal" ***ADD*** "Mobile Communication"
ADD "Groupware" ***ADD*** "Collaborative Work"
ADD "Collaboration"

D-ATC has better performance than DTC in both precision and recall. This shows that the terms assigned ***ADD*** operator could raise relevant documents in the first 1,000 ranked documents. Note that the average number of the whole retrieved documents was 23,129 in DTC and 13,965 in D-ATC, but the recall ratio was improved from 0.5283 to 0.5455. One rational conclusion from those facts could be that the query terms from the <DESCRIPTION> field are enough to collect relevant documents and the query terms from the <TITLE> and the <CONCEPT> field work for the relevance assessment. This experiment shows that the terms with the ***ADD*** operator can improve precision without expanding the set of retrieved documents if the operator is properly used.

Table 3-1

	DTC	D-ATC
Ave. Precision	0.1313	0.1474
R-Precision	0.1666	0.1831
Recall	0.5283	0.5455
Ave. retrieved docs	23,129	13,965

3.3 Effect of *Synonym* Operator

As described in 2.2.2, we can expect better retrieval effectiveness when ***Synonym*** operator is used only for concatenating synonyms, acronyms, abbreviations or whatever we can think them as different linguistic expressions of the same single concept. As for NTCIR topics, the query terms from the <CONCEPT> field can be good candidates being concatenated by this operator.

Table 3-2 describes the effect of ***Synonym*** operator. DTC indicates simply the same as stated in previous section. DTC-SYN1, 2 use the same fields as DTC to extract query terms, but the query terms from the <CONCEPT> field are concatenated by ***Synonym*** operator into a single term. In DTC-SYN1, terms written in the same line of the <CONCEPT> field are automatically concatenated by ***Synonym*** operator. For example, in topic 0031, two terms "flow control" and "rate control" are written in the same line of the <CONCEPT> field, so these two terms are concatenated into a single term. In DTC-SYN2, all terms in the <CONCEPT> field in a topic are investigated and selectively concatenated with ***Synonym*** operator by hand when we can see them as synonyms or acronyms. For example, "QOS" and "Quality of Service" are concatenated but "flow control" and "rate control" are not in DTC-SYN2. Again, ***naïve tfidf*** setting was used for the similarity calculation. An example query of topic 0036 for DTC-SYN1 is shown below.

constructed query for DTC-SYN1:
モバイル環境 ***OR*** モバイル ***OR*** グループウェア ***OR*** 問題点
OR 環境 ***OR*** グループウェア
OR 携帯端末,モバイル通信
OR グループウェア,共同作業,コラボレーション
OR "Personal Terminal", "Mobile Communication"
OR "Groupware", "Collaborative Work", "Collaboration"

In this example, "携帯端末" and "モバイル通信" are concatenated into one term, "グループウェア", "共同作業", and "コラボレーション" are also concatenated by ***Synonym*** operator. ***Synonym*** operator works well when it concatenates synonyms, acronyms, abbreviations, etc., because it treats concatenated terms as one term to prevent overestimating the *idf* values. Generally the adjusted *idf* value of the concatenated terms is lower than the sum of all the *idf* values of each term. When *Synonym* operator is used to concatenate conceptually related but not substitutable terms, as in the case of "flow control" and "rate control", the adjustment of *idf* makes no sense, or even makes things worse. For given topics, we have observed that some <CONCEPT> fields had in fact synonyms or acronyms, but others had merely conceptually related terms, which cannot be substitutes for each other.

This might be a reason why the precision and recall of DTC-SYN1 were slightly lower than DTC, which did not use ***Synonym*** operator. But the result of DTC-SYN2 shows that the ***Synonym*** operator can improve effectiveness when used properly. We think that it is difficult to automatically know which terms are appropriate for ***Synonym*** operator for the tasks like NTCIR, but in the real information retrieval situation where synonyms are supplied by some dictionaries, ***Synonym*** operator can simply be applicable to improve effectiveness.

Table 3-2

	DTC	DTC-SYN1	DTC-SYN2
Ave. Prec	0.1313	0.1290	0.1351
R-Prec	0.1666	0.1673	0.1732
Recall	0.5283	0.5225	0.5288

3.4 Effect of *Coordination Level Scoring*

Our second experiment is on the ranking facility of *MEISTER*, especially on the effect of *coordination level scoring* (***CLS***). It was reported by several researchers [7],[22] that coordination level information was in fact useful for short queries getting better performance. Our ***CLS*** is on the same line of those researches. In the experiment, we compared two ranking method, ***naïve tfidf*** and ***CLS*** described in 2.1. To see the sole effect of ***CLS*** clearly, we left other conditions equal and simple. Queries were constructed only by ***OR*** operator, ***ADD*** and ***Synonym*** operators were not used.

Table 3-3 shows the result. In the table, D indicates that query terms are extracted only from the <DESCRIPTION> field, and the ranking was done by ***naïve tfidf***. D-CLS is the same as D in query construction, but the ranking was done by ***CLS***. DTC and DTC-CLS indicates that query terms are extracted from the <DESCRIPTION>, the <TITLE> and the <CONCEPT> fields, and the ranking done by ***naïve tfidf*** and ***CLS***, respectively. Average numbers of query terms are 4.06 for D and D-CLS, 17.32 for DTC and DTC-CLS. So D and D-CLS correspond to (very) short queries, and DTC and DTC-CLS correspond to relatively long queries.

The precision and recall were improved drastically in both D-CLS and DTC-CLS compared with D and DTC. This result is supporting the previous studies. Besides, as far as our experiments on NTCIR topics and documents are concerned, coordination level information is fairly effective either for short queries and (relatively) long queries.

Table 3-3

	D	D-CLS	DTC	DTC-CLS
Ave. Prec	0.0899	0.2348	0.1313	0.3288
R-Prec	0.1144	0.2536	0.1666	0.3439
Recall	0.4555	0.5031	0.5283	0.6147

To see the effect of ***CLS*** more clearly, we investigated what was going on in ranking the documents with ***CLS***. As described in 2.1, ***CLS*** produces layers of document score depending on the number of matched query terms in the document. If the number of query terms is four, there will possibly be four layers of retrieved documents. The first layer contains documents that include all four query terms, and the second layer has those that match any three out of four query terms. Users' simple intuition about those layers is that if a document contains more query terms, then the relevance of the document to the query goes up. Thus we can expect that the upper a layer is, the better its precision rate is. Table 3-4 shows the average precision of each score layer. The first column indicates number of matched query terms. Topic 0050 and topic 0053 are picked up as typical examples.

Table 3-4

terms	D-CLS	DTC-CLS	topic 0050	topic 0053
10>		0.750		
9		0.750		
8		0.667	1.000	0.000
7		0.456	1.000	0.000
6		0.354	1.000	0.200
5	0.467	0.173	0.667	0.361
4	0.211	0.064	0.316	0.114
3	0.097	0.022	0.062	0.019
2	0.022	0.010	0.004	0.018
1	0.010	0.004	0.000	0.000

Both in D-CLS and DTC-CLS, the higher layers have better average precision. This result supports our expectation and suggests that ***CLS***, which uses the number of matched query term for scoring, is appropriate way of ranking, looking at its average performance. For example, the result of topic 0050 showed typical layers of score. The precision of the highest three layers, 8-6, were 1.0 and the precision of the lowest layer (including only one term) was 0. But if we are looking at each topic one by one, there are plenty of cases where our expectation simply blurred. Topic 0053 is such a case where the precision of 5 terms layer (0.361) was higher than the precision of 6 terms layer (0.200). Our guess for this case is that there might be at least one non-relevant term out of six terms extracted from topic 0053. The similar phenomena, that lower layers have higher precision than higher layers, were seen in 20 out of 53 topics in DTC-CLS.

One of the possible directions improving ***CLS*** is introducing some weights on coordination level information. WCM (weighted coordination match) of [22] is such an example. Another possible direction is introducing some normalization factor into coordination level information based on the query length. SQR (Short Query Ranking) function of [19] is such an example, also. Or, according to our ***CLS*** formula (1), changing its ratio of mixture with another information (changing C_1 and $\boldsymbol{C_2}$) would be another possibility.

Although those directions are worth being pursued, either of those would lead us to where all we have to do is parameter tuning. Our another motivation for heavy use of coordination level information in ***CLS*** is that ***CLS*** is fairly intuitive and transparent to users when they look at the ranking of the result documents. In practical text retrieval situation where efficient interactions between users and systems are very important, this transparency might be a great help. In this sense, another route to improve ***CLS*** would be that we let the combinatorial nature of coordination level information remain unchanged, let it be localized. For example, query terms should be localized so that only meaningful pairs of terms are considered. Or, the coordination of query terms in a document should be localized so that only meaningful fragments of the document are considered. The cover density ranking in [6] is taking such a route, or introducing phrase query into ***CLS*** would be another possibility. Those approaches will be investigated in future research.

3.5 Effect of Query Expansion from Non-dictionary Terms

Our third experiments is on query expansion. As described in 2.2.4, *MEISTER* has a term similarity calculation facility, which uses co-occurrence information at a document level, and can calculate similarities between both dictionary terms and non-dictionary terms. We used this facility to expand queries, selecting similar terms for each query term. In query construction processes, we extracted non-dictionary terms as well using *MEISTER*'s term extraction facilities. Then in query expansion processes, we can choose these non-dictionary terms as expanding terms.

The documents of NTCIR tasks are papers of Japanese academic societies. So there would be a lot of technical terms in such papers and queries, which are very often not registered in a dictionary (if the dictionary is not equipped with technical terms in certain domain). Then the ability to expand such non-dictionary terms would be a key to improve effectiveness through query expansion.

To make clear the merit of expanding non-dictionary terms, we compared the effects of expansion from dictionary terms with from non-dictionary terms in the experiment. For example, from the <DESCRIPTION> field of topic 0036, “モバイル環境” (mobile environment), “モバイル” (mobile) and “環境” (environment) were extracted in the query construction process. From a non-dictionary term “モバイル環境”, “モバイル” (mobile) and “いつでも” (any time, whenever) were selected as expanded terms and from a dictionary term “モバイル”, “グループウェア” (group ware) and “コンピューティング” (computing) were selected.

Table 3-5 shows the result. Expanded terms were limited to at most two terms per each. In selecting expanded terms, we used the default setting for term similarity calculation, which was shown in (11) in 2.2.4. In the table, D indicates that only the <DESCRIPTION> field was used for query term extraction, with no query expansion. D-D2 indicates that the dictionary terms extracted from the <DESCRIPTION> field were expanded, and D-ND2 indicates that the non-dictionary terms extracted from the <DESCRIPTION> field were expanded. The average number of expanded terms were 1.95 for a dictionary term and 1.35 for a non-dictionary term.

In the case of expansion from dictionary terms (D-D2), the precision and recall are slightly worse than the case of no expansion (D). In contrast, expansion from non-dictionary terms (D-ND2) got better performance. This result suggests two things. One is that the non-dictionary terms, which are usually compound terms and express more specific concept than dictionary terms, are effective for expansion. Another is that when dictionary terms are used for expansion, our method of query expansion might be in vain because it uses similarity calculation between terms using co-occurrence information in all the object documents.

Table 3-5

	D	D-D2	D-ND2
Ave. Prec	0.0889	0.0795	0.0981
R-Prec	0.1144	0.1023	0.1272
Recall	0.4555	0.4576	0.4607

3.6 Ad-hoc Formal Run

In the formal run, we tried to combine *MEISTER*'s facilities described so far, query construction using the term extracting facilities, ***ADD*** operator, and ***Synonym*** operator, ranking documents by ***CLS***, and query expansion from non-dictionary terms using the term similarity calculation facility.

For the short query task, we extracted at most four dictionary terms and at most four non-dictionary terms from the <DESCRIPTION> field, and selected at most two expanded terms for each non-dictionary term. All the query terms were ***OR***-ed to construct queries. (***ADD*** and ***Synonym*** operator were not used.) For the optional task (long query task), we extracted terms from the <DESCRIPTION> field, the <TITLE> field, and the <CONCEPT> field, then selected expanded terms for non-dictionary terms from the <DESCRIPTION> field. All the terms are once ***OR***-ed and terms from the <DESCRIPTION> field and the <CONCEPT> field are then ***ADD***-ed to the query. Here we decided which field should be ***ADD***-ed on the result of preliminary run. For both tasks, ***CLS*** was used to rank documents.

The average precision of the short query task and the long query task were 0.2560 and 0.3316 respectively. Figure 3-1 and 3-2 show the overall results comparing the formal run and each effect of ***MEISTER***'s facility described through 3.2 to 3.5. Figure 3-1 is a 11pt. precision graph of the short query baseline (D), ranking by ***CLS*** (D-CLS), expansion from non-dictionary terms (D-ND2) and the formal run (NTE151). Figure 3-2 is a 11 pt. precision graph of the long query base line (DTC), query construction with ***ADD*** operator (D-ATC), query construction with ***Synonym*** operator(DTC-SYN2), ranking by ***CLS*** (DTC-CLS), expansion from non-dictionary terms (DTC-ND2) and the long query formal run (NTE152).

4. CLIR TASK

In this section, we describe our experiments on CLIR task. In 4.1, we summarize previous approaches to CLIR and explain our corpus-based approach. In 4.2, we describe our query construction process in CLIR. Then, we show some results of the experiments in 4.3. Finally in 4.4, the results of our formal run are presented.

4.1 Term Translation Using Parallel Corpora

For CLIR, many approaches have been proposed in past researches. The major approaches are classified into following three types or their combination [3].

(a) MT system based

(b) Corpus based

(c) Language resource (ex. bilingual dictionary) based

Especially, corpus based approach have been often used with other approaches: for example, using corpus based MT system [7] or disambiguating translation words in bilingual dictionary using statistical information in corpora [2]. Also, entirely corpus based (statistical) approaches have been experimented, such as LSI [18] or ETH's query translation using similarity thesauri [4].

We employed a corpus-based approach similar to ETH's. However, we didn't prepare resources such as similarity thesauri, in which terms are statically connected with their translations. Instead, we used a Japanese-English parallel corpus and term similarity calculation facility of *MEISTER*. The essence of our approach is:

(a) Construct a term-document matrix from English (target language) documents of the parallel corpus.

(b) Construct an index from Japanese (source language) documents of the parallel corpus.

(c) Construct term vectors using the Japanese index each of which corresponds to a Japanese query term.

(d) Calculate similarities between Japanese term vector and each term vector in the English term-document matrix.

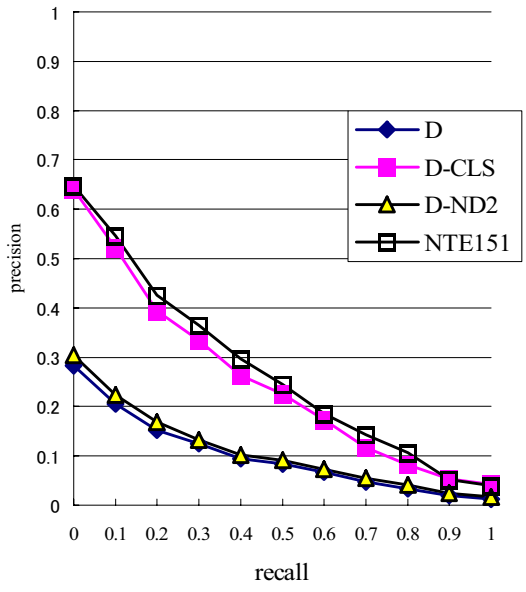

Figure 3-1

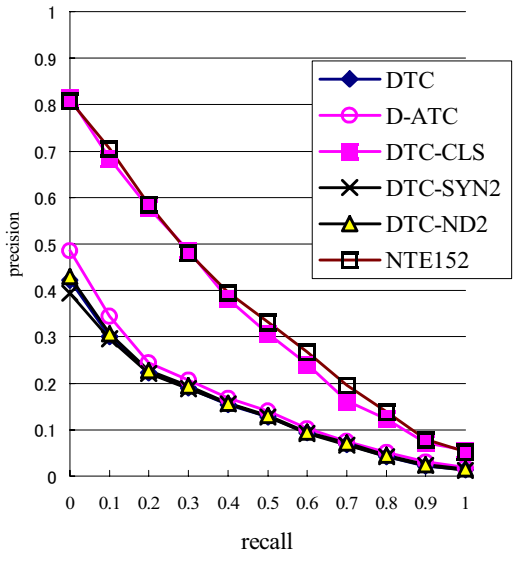

Figure 3-2

Then, select *n* English terms which have the highest similarities.

In step (c), we used string search facility of *MEISTER*. So, for any arbitrary string, we could get "*pseudo-translated*" terms as far as the string occurs in the corpus. In the case that source language doesn't have explicit word boundaries such as Japanese, this feature will be very important. Especially, Japanese queries in NTCIR contains many non-dictionary technical terms. We can pseudo-translate such terms with this feature.

Figure 4-1 shows the diagram of our approach.

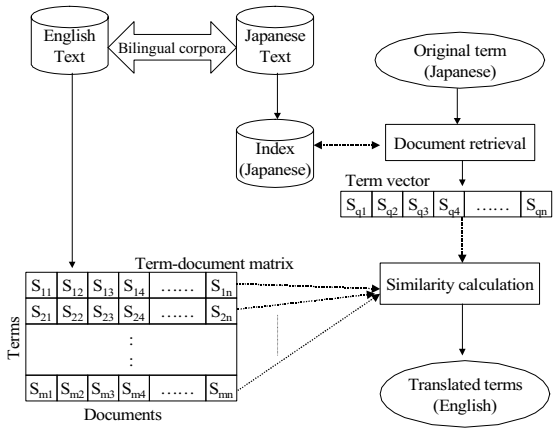

Figure 4-1 Translated term selection using similarity calculation

4.2 Query Construction

In our system, we construct English queries automatically by following steps.

(a) Extract query terms from a Japanese description of query request using the same method described in 3.1.

(b) Pseudo-translate Japanese terms to English terms using the method described in 4.1.

(c) Terms selected in step (b) are *OR*-ed to construct an English query. The duplicates of terms are not eliminated, simply taken as isolated terms.

To build a parallel corpus used in step (b), we extracted documents which have both Japanese and English abstracts from NTCIR JE-collection. Then, we made a Japanese index and an English term-document matrix from this corpus. The number of documents in this parallel corpus is 181,489.

For NTCIR CLIR task, we decided to select two English terms per one Japanese term in step (b), because we thought Japanese terms extracted from Japanese queries in NTCIR would correspond to multi-word English terms (phrases) in many cases. For example, "光ファイバ" would correspond to "optical fiber" , "データ転送" would correspond to "data transfer", and so on. Moreover, we had few phrases (especially technical terms) in our English dictionary.

An example of constructed query is shown below.

TOPIC:	0054	
DESCRIPTION:	「光ファイバを用いたギガビット通信について」	
Japanese terms:	光ファイバ，ギガビット通信，ギガビット，通信	
Pseudo-translation:	光ファイバ	→ optical, fiber
	ギガビット通信	→ gigabit, communication
	ギガビット	→ gigabit, giga-bit
	通信	→ communication, network
English query:	optical ***OR*** fiber ***OR*** gigabit ***OR*** communication ***OR*** gigabit ***OR*** giga-bit ***OR*** communication ***OR*** network	

4.3 Experiments and Analysis

In this section, we describe our experimental results in CLIR task. The point of experiments are (1) the effectiveness of string search facility, (2) the influence of reducing dimensions of term vectors, (3) the effectiveness of using phrases, and (4) the effectiveness of pseudo feedback.

Since "the <DESCRIPTION> field only" run (short query task) is obligatory for automatic query construction system in the workshop, we used following condition as a basic setting.

- Extract Japanese query terms only from the <DESCRIPTION> field.
- Select two English terms per each Japanese query term.
- Rank documents with ***CLS***.

D-C2 indicates this basic setting. We don't describe the long query task (optional run) in this section.

Our English document retrieval system is basically similar to our Japanese one, but have some differences [8]. Especially, it can identify a word and its inflected forms.

4.3.1 Effect of String Search Facility

As described in 4.1, we expected that string search facility and dynamic construction of term vectors would be effective, especially using Japanese as a source language. To verify this expectation, we compared our basic setting with the method without string search facility.

[D-C2] can select English terms against both dictionary and non-dictionary Japanese terms, using string search facility to constructing term vectors of Japanese terms.

[D-C2M] can select English terms only against Japanese dictionary terms, since it doesn't use string search facility.

For example, from Japanese non-dictionary term "マルチキャスト", "multicast" and "multicasting" are selected in D-C2, while no term obtained in D-C2M.

Table 4-1 shows the result.

Table 4-1

	D-C2M	**D-C2**
Ave. Precision	0.1383	0.1650
R-Precision	0.1595	0.1726
Recall	0.5467	0.5651

D-C2 is higher than D-C2M in both precision and recall as we expected.

4.3.2 Reducing the Number of Documents in Parallel Corpora

The parallel corpus we used had large number of documents (181,489 documents), and contained almost all of the object documents. However, such ideal situation cannot be expected in practical IR situations. So, we examined the influence of reducing the number of documents in the corpus.

[D-C2] Use the full size parallel corpus.

[D-C2H] Use a half size parallel corpus. (randomly selected 50% documents from full size, about 90,000 documents)

[D-C2Q] Use a quarter size parallel corpus. (randomly selected 25% documents from full size, about 45,000 documents)

We used two corpora in D-C2H, and four corpora in D-C2Q. Values in Table 4-2 is the average of using each corpus.

Table 4-2

	D-C2	D-C2H	D-C2Q	median
Ave. Prec.	0.1650	0.1473	0.1361	0.9680
R-Prec.	0.1726	0.1520	0.1435	0.1197
Recall	0.5651	0.5592	0.5716	-

Precision became lower with smaller number of documents. However, recall of three results are almost the same level. Additionally, D-C2Q achieved higher precision than the median among submitted runs.

4.3.3 Using Phrases

As mentioned in 4.2, we decided to select two English terms per one Japanese query term in our method. But in some cases, this method might not work well, because one of two terms often seemed to be unrelated to the original Japanese term. For example, for the Japanese term "スループット (throughput)", "throughput" and "packet" were selected in topic 0038, where "packet" seemed to be unrelated.

If there are many phrases in English dictionary, we might be able to avoid to select those terms by selecting one English term (or phrase) per one Japanese term. So, we extracted two-word phrases from E-collection automatically, and added these phrases into the dictionary. Phrase extraction step was:

(a) Select stop words among high frequency words by hand, which may not be a constituent of phrase. (520 stop words were selected)

(b) Extract word bi-grams which are not contain stop words.

(c) Add high frequency word bi-grams as phrases to the dictionary. (38,928 phrases, occurring more than 10 times were added)

We then rebuilt the English term-document matrix and the English index for retrieval using this dictionary, and compared with the case without phrases (D-C2).

[D-CP1] With phrase dictionary, select one term per each

[D-CP2] With phrase dictionary, select two terms per each

Surprisingly, the results with phrase dictionary were getting worse as shown in Table 4-3. However, we observed that the automatically extracted phrases were not so bad, especially most of all selected phrases as translated terms seemed to be appropriate. This result suggests that many single-word terms are effective for retrieval, even though they seemed unnecessary at a glance. Those terms might have a kind of query expansion effect.

Table 4-3

	D-C2	D-CP1	D-CP2
Ave. Precision	0.1650	0.1399	0.1258
R-Precision	0.1726	0.1479	0.1344
Recall	0.5651	0.4782	0.5081

We think treating multi-word terms as well as single-word terms is also problematic. One solution to this problem may be using multi-word terms with ***ADD*** operator, as used in the ad-hoc task. Queries in D-C2 are sufficient to fix the result set of documents, since the recall rate is 0.85 in the experiment retrieving top 10000 documents with queries in D-C2. So, it might be a reasonable method that fixing results with single-word terms and ranking results with single- and multi-word terms using ***ADD*** operator. We'll examine this in the future.

4.3.4 Effect of Pseudo Feedback

In recent TREC ad-hoc and CLIR task, many systems employed query expansion facility, especially a kind of pseudo feedback[3][21]. We also tried to use pseudo feedback.

Our pseudo feedback was realized in following steps:

(a) Retrieve documents using translated terms. Then extract *w* terms from *n* top ranked documents using term extraction facilities of *MEISTER*.

(b) Add *w* terms to the original query.

According to the parameter tuning with preliminary run, we decided the number: $n = 5$, $w = 2$. Then we compared a run with pseudo feedback (**D-C2B**) with a run without pseudo feedback (**D-C2**). D-C2B used the same setting of D-C2 in its first retrieval (before expanding query).

Table 4-4 shows the result.

Table 4-4

	D-C2	D-C2B
Ave. Precision	0.1650	0.1587
R-Precision	0.1726	0.1592
Recall	0.5651	0.6077

D-C2B achieved higher recall but lower precision. In the preliminary run, however, using pseudo feedback improved in both precision and recall. This might be caused by the low precision of top-ranked documents in D-C2. The precision of top 5 documents was 0.5238 in the preliminary run against 0.2564 in D-C2.

4.4 CLIR Formal Run

Table 4-5 shows the results of our formal runs. **NTE153** is submitted obligatory run, using only the <DESCRIPTION> field to extract Japanese query terms and using pseudo feedback (=D-C2B). **NTE154** is submitted optional run, using the <DESCRIPTION>, the <TITLE> and the <J.CONCEPT>

fields to extract Japanese query terms. Pseudo feedback was not used in NTE154.

Unfortunately, some bugs were found in English retrieval system after the submission. **RNTE153** and **RNTE154** are indicators of the results after fixing these bugs.

Table 4-5

	NTE153	RNTE153	NTE154	RNTE154
Ave. Prec.	0.1517	0.1587	0.1907	0.1932
R-Prec.	0.1633	0.1592	0.1938	0.1949
Recall	0.5604	0.6077	0.6663	0.6775

Also we checked our CLIR performance against the monolingual-retrieval using NTCR J-collection. Our monolingual-retrieval was the same method as used in ad-hoc task, with following settings.

[MD] Construct queries from the <DESCRIPTION> field only

[MDTJ] Construct queries from the <DESCRIPTION>, the <TITLE> and the <J.CONCEPT> fields.

In both settings, We didn't use ***ADD*** nor ***Synonym*** operator, and any query expansion. So, MD setting of monolingual retrieval corresponds to D-C2 of CLIR, and MDTJ corresponds to RNTE154.

Table 4-6 and Figure 4-2 show the monolingual-retrieval results and our CLIR performance against the monolingual-retrieval.

Table 4-6

	MD	D-C2/ MD	MDTJ	RNTE154 / MDTJ
Ave. Prec.	0.2077	79%	0.2858	68%
R-Prec.	0.2368	73%	0.3215	60%
Recall	0.4925	115%	0.5957	114%

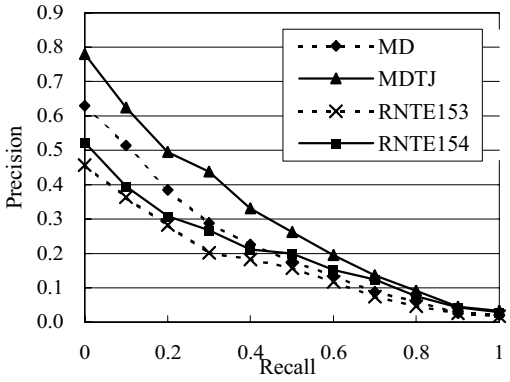

Figure 4-2 11point precision of our monolingual-retrieval and CLIR

Considering monolingual-retrieval as a baseline, our CLIR achieved 68-79% in average precision, and 60-73% in R-precision. Besides, recall of CLIR was 14-15% higher than monolingual-retrieval. We think our method might include effects like query expansion.

5. CONCLUSION

In this paper, we described the result of our experiments on NTCIR tasks, ad-hoc and CLIR task.

As for ad-hoc task, our experiments were mainly on evaluating several facilities of ***MEISTER*** system, to confirm that they can improve retrieval effectiveness. From the results mentioned in section 3, ***ADD*** operator, ***Synonym*** operator, ***CLS*** (coordination level scoring) and query expansion from non-dictionary terms have certain effects in improving retrieval effectiveness, if they are used properly to construct queries.

As for *CLS*, we observed the drastic improvement in each experiment. Some investigations were made in 3.4, but we still need further experiments and investigations, which will be our next research issue.

In CLIR task, we proposed an example of corpus based CLIR, translated term selection based on parallel corpora. Our proposed approach achieved better effectiveness in this task.

In real situations, however, there may be many cases achieving better result with bilingual dictionaries. Because our method is completely depends on the quality and quantity of parallel corpora, we cannot translate terms which do not occur in the corpora. So, a hybrid approach, using both parallel corpora and bilingual dictionaries, will be a good direction to be pursued.

Also, our method currently ignores query contexts, i.e. translated term is selected without considering entire query in which the original query term occur. However, we may be get more adequate translated terms considering query contexts. This is another work we have to do in the future.

6. REFERENCES

[1] Akamine, S., and Fukushima, T. High-speed Full-Text Retrieval System: RetrievalExpress (in Japanese). Proc. 54th Annual Convention IPS Japan (1997), 7L-02.

[2] Allan, J. et al. INQUERY Does Battle With TREC-6. 6th Text REtrieval Conference (1997), 169-206.

[3] Brashler, M. et al. Cross-Language Information Retrieval (CLIR) Track Overview. 7th Text REtrieval Conference (1998), 25-32.

[4] Brashler, M. et al. SPIDER Retrieval System at TREC7. 7th Text REtrieval Conference (1998), 509-518.

[5] Church, K.W. et al. Introduction to the Special Issue on Computational Linguistics Using Large Corpora. Computational Linguistics Vol.19, No.1 (1993), 1-24.

[6] Cormack, G.V., et al. Passage-Based Refinement (MultiText Experiments for TREC-6). 6th Text REtrieval Conference (1997), 303-320.

[7] Franz, M. et al. Ad hoc and Multilingual Information Retrieval at IBM. 7th Text REtrieval Conference (1998), 157-168.

[8] Fukushige, Y. et al. Maximal-Extension Indexing method for Smart Text Retrieval *MEISTER*: Application to English Language Text Retrieval (in Japanese). Proc. 55th Annual Convention IPS Japan (1997), 3N-05.

[9] Inaba, M. et al. Maximal-Extension Indexing method for Smart Text Retrieval *MEISTER*: Ranking Subsystem (in Japanese). Proc. 55th Annual Convention IPS Japan (1997), 3N-03.

[10] Kanno, Y. et al. Maximal-Extension Indexing method for Smart Text Retrieval *MEISTER*: Dictionary/Index

Sybsystem (in Japanese). Proc. 55th Annual Convention IPS Japan (1997), 3N-02.

[11] Kurachi, K., Noguchi, N., Kanno, Y., and Inaba, M. New Indices for Japanese Text: The principles of making index and searching index (in Japanese). Proc. 50th Annual Convention IPS Japan (1995), 4F-02.

[12] Lee, J.H. and Ahn, J.S. Using *n*-Grams for Korean Text Retrieval. Proc. of 19th SIGIR Conference (1996), 216-224.

[13] Matsui, K., Nanba, T., and Igata, N. Hi-speed Fulltext Search Engine (in Japanese). IPS Japan SIG Notes, Vol. No. (1997), 15-20.

[14] Noguchi, N., Inaba, M., Nomoto, M., and Kanno, Y. A New IR Model Utilizing Word Statistics and Linguistic Information (in Japanese). IPS Japan SIG Notes, Vol. 96, No.116 (1996), 33-40.

[15] Noguchi, N. et al. Maximal-Extension Indexing method for Smart Text Retrieval *MEISTER*: Overview (in Japanese). Proc. 55th Annual Convention IPS Japan (1997), 3N-01.

[16] Noguchi, N., Kanno, Y., Kurachi, K., and Inaba, M. New Indices for Japanese Text: A New Word-based Index of Non-segmented Text for Fast Full-text-search Systems. Transactions of IPS Japan, Vol.39, No.4 (1998), 1098-1107.

[17] Ogawa, Y., and Matsuda, T. An Efficient Document Retrieval Method Using n-gram Indexing. (in Japanese) Transactions of IEICE Japan, Vol.J82-D-I, No.1 (1999), 121-129.

[18] Rehder, B. et al. Automatic 3-Language Cross-Language Information Retrieval with Latent Semantic Indexing. 6th Text REtrieval Conference (1997), 233-240.

[19] Rose, D. E., and Stevens, C. V-Twin: A Lightweight Engine for Interactive Use. 5th Text REtrieval Conference (1996), 279-290.

[20] Sato, M. et al. Maximal-Extension Indexing method for Smart Text Retrieval *MEISTER*: Related Keywords Extraction (in Japanese). Proc. 55th Annual Convention IPS Japan (1997), 3N-04.

[21] Voorhees, E. and Harman, D. Overview of the Seventh Text REtrieval Conference (TREC-7). 7th Text REtrieval Conference (1998), 1-24.

[22] Wilkinson, R., Zobel, J., and Sacks-Davis, R. Similarity measures for short queries. 4th Text REtrieval Conference (1995), 277-285.