On Two Inference Methods for Database Systems

Taiho KANAOKA*, Hiroshi MORINAGA*, Kiyohiro KOBAYASHI*,
Shingo TOMITA*, Masashi TAKAHASHI**, Noboru KAWAKAMI**,
Tsuguyasu IMAMURA** and Fumirou TANOUE**

(Received July 15, 1989)

Abstract

This paper proposes two inference methods for database systems. One is called the string inference by which the similarity measure between two strings can be derived. The similarity measure is examined on condition that the string can be divided into significant substrings. The other is called the common item inference which finds some kinds of valid attribute values in the noniput attributes, and using these values the other candidate documents are retrieved. A system construction including those two inference methods is also described.

1. Introduction

In current database systems at first the user must input strings as the keywords or attribute values concerning his desired documents. At that time, he must inputs exact strings and understand the construction of the attribute values. Moreover those systems enforce to input other strings upon the user, when he can not get the desired documents. In order to improve those problems, the system planer should intend to make full use of informations concerning the attribute values given by the user.

This paper proposes a basic idea of two inference methods to retrieve the desired documents effectively, and describes a system construction including the inference methods. Those inference methods will be available the case of that the user does not exactly know the attribute values and he dose not find the desired one in the documents presented by the system even if he inputs strings exactly. We call two inferences string infrence and common item inference.

The string inference supports the user to input strings as attribute values. Namely, when the user inexactly inputs the attribute values, by using the string inference the system shows to the user the other attribute values similar to the input strings. As for the similaity measure between two strings, some ideas based on edit operations such as substitution insertion or deletion (1), (2), the number of times a substring occurs in two strings(3) and hierarchically organized file(4) have been suggested. However, they are entirely syntactic approach and don't involve semantic considerations for strings. This paper suggests a similarity measure on condition that the string can be divided into significant substrings semantically.

If an attribute value in a noninput attribute frequently appears in the documents retrieved by the input attribute values, it may be natural to think that it has some

*Faculty of Engineering, Yamaguchi University Ube 755, Japan
**NEC Software Chugoku, Ltd Hiroshima 732, Japan

useful information to retrieve the other candidate documents. Namely, the common item inference finds some kinds of valid attribute values in the noninput attributes and using those values the other canidate documents are retrieved.

2. Basic construction of the system

Before idea of two inferences are described, we introduce the basic construction of a database system including the inference modules. A rough flow of the system is shown in Fig.1.

Let for each i($1 \leqq i \leqq n$) document d_i has m attributes A_1, A_2, $\cdots$, A_m and for each

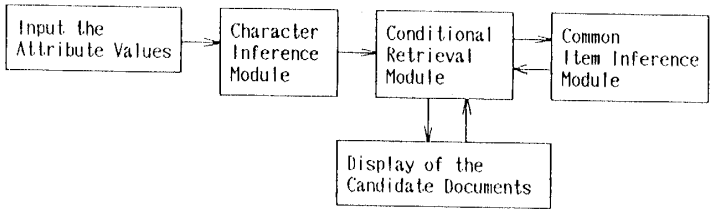

Fig. 1 Rough flow of the system

$t(1 \leqq t \leqq m)$ A_t has a set of attribute values $\{a_{tk} \mid 1 \leqq k \leqq \rho_t\}$ (refer to Fig.2). At first the user must input some attribute values for some attribute to find his desired document. After inputting the strings as the attribute values are completed, the system begins to retrieve the document. Then string inference module, conditional retrieval module and common item inference module are operated one after another where necessary.

2.1 String inference module

When the user intends to input some strings as the attribute values, there exist some cases such that he does not exactly know to which attribute each string belongs or makes a mistake to input a part of the string. In the situations the string inference module derives some strings similar to the input string.

We describe detail of the string inference in subsequent section.

2.2 Conditional retrieval module

From now on, for simplicity we assume that the user inputs only one attribute value for each attribute.

Let α_i be a input string for a attribute A_i, and

$$d_l(A_i) = \{a_{ik} \mid 1 \leqq k \leqq \sigma\} \quad (1 \leqq \ell \leqq n)$$

be a set of attribute values which belong to A_i in a document d_l. And let

$$Q(\alpha_i, a_{ik}) \quad (0 \leqq Q(\alpha_i, a_{ik}) \leqq 100)$$

be a similarity measure between α_i and a_{ik}, and we put

$$Q^* = \max_{1 \leqq k \leqq \sigma} Q(\alpha_i, a_{ik}) \tag{1}$$

Then we say that, for the document d_l, α_i is

$$\begin{cases} T(ture) & \text{if } Q^*=100 \\ T' & \text{if } t\leqq Q^*<100 \\ F(false) & \text{if } 0\leqq Q^*<t, \end{cases} \quad (2)$$

, where t is threshold.

Now we put D as a set of all documents in the data file, namely

$D=\{d_i \mid 1\leqq i\leqq n\}$.

In this time, based on the input attribute values, D can be partitioned into subsets as follows.

$$D=\overset{R}{\underset{i=1}{\cup}}D_i \quad (i\neq j\rightarrow D_i\cap D_j=\phi,\ R=3^p,\ D_i\subseteq D)$$

where we assume that p attribute values are input in order of attributes A_{i_1}, A_{i_2}, $\cdots$ A_{i_p} and if $i>j$ then the priority of attribute A_i is greater than A_j. D_1 is a subset that all input attribute values α_{i_1}, α_{i_2}, and α_{i_p} are true for any document in D_1, and D_2 is a subset that α_{i_1}, α_{i_2}, $\cdots$, $\alpha_{i_{p-1}}$ are true and α_{i_p} is T′ for any document in D_2. Moreover, D_3 is a subset that α_{i_1}, α_{i_2}, $\cdots$, $\alpha_{i_{p-1}}$ are true and α_{i_p} is false for any document in D_3. More detail is shown in Fig.4. The set of documents D_i stated above is called a set of documents of rank i. In conditional retrieval module, the system tries to find candi-

Table 1 Constitution of the documents (d_i : document's name, A_i : attribute, ai_j : attribute's value)

	A_1	A_2	A_3	.....	A_m	...
d_1	a_{11} a_{12}	a_{21}	a_{31} a_{32} a_{33}	.....	a_{m1}	...
d_2	a_{13} a_{14}	a_{21} a_{22}	a_{32}	.....	a_{m1} a_{m2}	...
: :	: :	: :	: :	.....	: :	...

Table 2 A partition of the documents in data file

	A_{i_1}	A_{i_2}	.....	$A_{i_{p-2}}$	$A_{i_{p-1}}$	A_{i_p}
D_1	T	T	.....	T	T	T
D_2	T	T	.....	T	T	T′
D_3	T	T	.....	T	T	F
D_4	T	T	.....	T	T′	T
D_5	T	T	.....	T	T′	T′
D_6	T	T	.....	T	T′	F
:	:	:	.....	:	:	:
D_R	F	F	.....	F	F	F

Table 3 A table for explaining the conception of the common item (input attribute values are Ball, Tokyo and Yamada)

retail store	good's name	site	person in charge	・・・・	company of purchase	
s_1	Ball	Tokyo	Yamada	・・・・	Locky	
s_2	Ball	Tokyo	Yamada	・・・・	Locky	
s_3	Ball	Tokyo	Yamada	・・・・	Locky	
s_4	Ball	Tokyo	Yamada	・・・・	Metro	
s_5	Ball	Tokyo	Yamada	・・・・	Locky	
s_6	Ball	Osaka	Yamada	・・・・	Locky	
s_7	Ball	Kyoto	Yamada	・・・・	Locky	
s_8	Ball	Osaka	Yamada	・・・・	Locky	

date documents in D_1. If no desired document is in D_1, retrieval area is removed to D_2, and so on. However, in the process of retrieval in each rank, the user can access the common item inference module stated below.

2.3 Common item inference module

The situations that there is not desired one in documents retrieved by conditional retrieval module are sometimes occur. In such case, we try to derive some kinds of valid attribute values called common item which exist in non input attribute. Based on those valid attribute values this module continues to find other candidate documents. For example of common item inference, in Table 3 we give an information of retail stores possessed an wholesaler. The user wants to get a retail store having attribute values Ball as the good's name, Tokyo as the site and Yamada as the person in charge. There are five candidate documents of retail store s_1, s_2, ・・・, s_5 satisfied above three attribute values, but they don't meet his desire. Now, we refer to non input attribute values in s_1, s_2, ・・・, s_5. Then we can see that the most of the company of purchase is Locky. As the documents that attribute value of the company of purchase is Locky and the most of input attribute values are satisfied, s_6, s_7 and s_8 are retrieved. Namely, as the candidate document the system can show s_6, s_7 and s_8 to the user. As seen in this example, by using the common item inference, paying attention to the appearance ratio of a noninput attribute value in documents satisfied input attribute values, the system can retrieve other candidate documents. Detail of the common item inference is discussed in the later.

3. String inference

String inference module relates to the constitution of the rank of the documents in conditional inference module and to the inference procedure in common item inference

module. The role of the string inference is to estimate the similarity between a input string and the strings stroed in data files, and to pick up a string in data files whose similarity is comparatively large.

How to formulate the similarity measure is very important. It is best to imitate the human sense and intuitiveness in decision process. Considering this point of view, we formulate the similarity measure.

3.1 Segmentation of the string

In many retrieval systems, attribute value or keyword as input or date usually consist of some kinds of singnificant substrings. For example "ACOS4-AVP 1.2" consists of alphabets ACOS, AVP and figures 4 and 1.2. That is, we firstly devide the string into substring such as alphabet, Chinese character, Katakana and figure. Also we devide the string in the point of special letters " ", "-", etc. In "ACOS4 AVP-1.2" we can devide it into "ACOS", "4", "AVP" and "1.2".

3.2 Similarity measure of substrings

Let $l=l_1\ l_2\ \cdots\ l_u$ $(u\geqq 1)$ and $F=F_1\ F_2\cdots\ F_v$ $(v\geqq 1)$ be an input string and a string in data file respectively, where l_p and F_q are substrings. And let $l_p=c_1\ c_2\ \cdots\ c_m$ $(m\geqq 1)$ and $F_q=f_1\ f_2\ \cdots\ f_n$ $(n\geqq 1)$, where c_k and f_k are letters.

3.2.1 Number of common letters

When we put l_p and F_q in order that just below of the letter c_i is f_j, $h_{l_pF_q}(i, j)$ denotes the number of positions to coincide the letters. For example in the case $l_p=$ACOS6 and $F_q=$FACOM, $h_{l_pF_q}(1, 2)=3$, $h_{l_pF_q}(i, j)=0$ (for $i\neq 1$, $j\neq 2$).

3.2.2 Sequential search of maximum number of common letters

We define maximum number of common letters between l_p and F_q as follows.

$$\max_{\substack{1\leqq i\leqq m\\ 1\leqq j\leqq n}} h_{l_pF_q}(i, j)=h_{l_pF_q}(i_1, j_1)=h^1 \tag{3}$$

Then h^1 common letters, obtained by putting l_p and F_q in order that just below of the letter c_{i_1} is f_{j_1}, replace blank in l_p and F_q respectively. We put such strings as follows.

$$l_p^1=c_1^1\ c_2^1\ \cdots\ c_m^1$$
$$F_q^1=f_1^1\ f_2^1\ \cdots\ f_n^1$$

Then we can define the following

$$\max_{\substack{1\leqq i\leqq m\\ 1\leqq j\leqq n}} h_{l_p^1F_q^1}(i, j)=h_{l_p^1F_q^1}(i_2, j_2)=h^2$$

, where blank is not common letter.

Furthermore we put l_p^2 and F_q^2 in which new blanks are replaced with h^2 common letters in l_p^1 and F_q^1 respectively. We continue the procedure stated above till common letter does not appear.

3.2.3 Serial common letters

We assume that common letter does not appear for the first time at the w+1th procedure in §3.2.2. In this time, at kth procedure we put substrings of serial common

letters in order from leftmost as follows.

$S_1^k, S_2^k, \cdots, S_{v_k}^k \ (1 \leqq k \leqq w)$

, where $\sum_{i=1}^{v_k} |S_i^k| = h$

For sequence of letters C, $|C|$ denotes the length of C.

3.2.4 Similarity measure between substrings

Considering the number of common letters, the number of serial common letters, length of the string and so on, we formulate the similarity measure between l_p and F_q as follows.

$$R_{pq} = \sum_{k=1}^{w} W_c(h^k/m) \sum_{j=1}^{v_k} W_r(|S_j^k|/h^k) \cdot \frac{|S_j^k|}{m} \cdot \frac{\alpha}{|m-n|/m+\alpha} \times 100 \qquad (5)$$

where h^k denotes the maximum number of common letters obtained at kth procedure in §3.2.2 and $W_c(h^k/m)$ represents the dependency of the number of common lettrers to the similarity measure. $0 \leqq W_c(h^k/m) \leqq 1$ and $h^i > h^j \rightarrow W_c(h^i/m) > W_c(h^j/m)$ hold. $W_r(|S_j^k|/h^k)$ represents the dependency of the sequence of serial common letters to the similarity measure. And $|S_i^k| > |S_j^k| \rightarrow W_r(|S_i^k|/h^k) > W(|S_j^k|/h^k)$. $|S_j^k|/m$ implies that similarity measure depends on the length of input string. $\alpha/(|m-n|/m+\alpha)$ implies that the similarity measure depends on two length of input string and string in file. That is, if difference between m and n is large, then similarity measure is small. W_c, W_r and α are the functions and the parameter to be decided suitably.

3.3 Total similarity measure between two strings

In order to estimate the similarity measure between strings by using equation (5), we take into account of priority of substrings and correspondence between substrings of input string and substring in file.

3.3.1 Priority of substrings

When we consider the similarity measure between an input substring l_i and an l_j $(i<j)$ for a substring in file, it will be natural that if $i<j$ then l_i contributes to the similarity measure more than l_j. Namely,

$i<j \rightarrow g(l_i) > g(l_j)$

, where $g(l_t)$ is ratio of contribution of l_t to the similarity measure.

3.3.2 Correspondence and distance between substrings

For each substring l_i $(1 \leqq i \leqq m)$, let F_j^* be a substring in F corresponding to l_i, where

$$\max_{1 \leqq j \leqq n} R_{ij} = R_{ij}^* \qquad (6)$$

Here we asssume that there is not correspondence between different kinds of strings. And when l_i corresponds to F_j we define the distance d(i) from l_i to F_j as follows.

$$d(i) = |i-j| \qquad (7)$$

, where if correspondence doesn't exist then $d(i)=0$.

Namely, we think that if l_i corresponds to substring F_j of F and F_k' of F' respectively and $d(i) < d'(i)$ $(d(i) = |i-j|, d'(i) = |i-k|)$ then the similarity measure between l_i and F_j contributes to the total similarity more than one between l_i and F_j'.

3.3.3 Total similarity measure

In this section using equation (5) we introduce a total similarity measure Q(l, F) between l and F. To derive Q, adding to the priority, correspondence and distance concerning the subcharacter we also consider the length of the string and substring. We think that the contribution of R_{pq} to the total similarity become great in the proportional of ratio of the length of substring to length of total string. From those several points of view, we define Q(l, F) as follows.

$$Q(l,\ F)=\sum_{p=1}^{u}\frac{g(l_p)\cdot|l_p|\cdot f(d_p)}{\sum_{p=1}^{u}g(l_p)\cdot|l_p|}R_{pq}\frac{\beta}{z/u+\beta} \tag{8}$$

, Where $\sum_{p=1}^{u}g(l_p)\cdot|l_p|$ is normalized factor, z is number of substrings which don't correspond to any substrings of F and we assume that l_p corresponds to F_q. g, f and β are the functions and the parameter to be decided suitably.

(Example 1) The similarity measure between I_p=LUNFORCAST and F_q=LANFOR-ECAST is derived as follows.

At first, it is easily seen that

$h^1=h_{I_pF_q}(1,\ 1)=5$, $S_1^1=L$, $S_2^1=NFOR$, $|S_1^1|=1$, $|S_2^1|=4$.

Then $I_p^1=$␣U␣␣␣␣CAST

$F_g^1=$␣A␣␣␣␣ECAST

Thus we get

$h^2=h_{I_p^1F_q^1}(1,\ 2)=4$, $S_1^2=CAST$, $|S_1^2|=4$.

Now, we assume that

$$W_c(X)=\begin{cases}3.2*X, & \text{if } (0\leqq X<0.3)\\ 0.07*X+0.93, & \text{if } (0.3\leqq X\leqq 1)\end{cases},$$

$W_r(X)=1.1-1.1/(10*X+1)$

and $\alpha=2.5$.

Then we get

$W_c(5/10)=0.97$, $W_r(1/5)=0.73$, $W_r(4/5)=0.97$,

$W_c(4/10)=0.96$ and $W_r(4/4)=1$.

Accordingly, it follows from (5) that

$R_{pq}=80(\%)$

(Example 2) The similarity measure between l=LUNFORCAST-AT200 and F=LANFORECAST-210 is derived as follows.

Let $I=I_1I_2I_3$, where I_1=LUNFORCAST, I_2=AT, I_3=200 and let $F=F_1F_2$, where F_1=LANFORECAST, F_2=210.

We assume that

$g(I_p)=0.25*(u-p+1)/u+0.75$,

$f(d_p)=3/(d_p/u+3)$

and $\beta=4$.

Then we get

$g(I_1)=1$, $g(I_2)=0.92$, $g(I_3)=0.83$,

$f(d_1)=1$, $f(d_2)=0$ and $f(d_3)=0.9$.

Thus, it follows from $R_{11}{}^*=80(\%)$, $R_{32}{}^*=58(\%)$ and (8) that

$Q(I,\ F)=60(\%)$.

Table 4 (a) Example of similarity measure for a input character ACOS1.

characters in file	similarity measure(%)
ACOS2	80
NCOS1	77
ACOS2EF	70
ACOS4MVP	70
ITOS1／4	49

Table 4 (b) Example of similarity measure for a input character TOSBAC M1.

characters in file	similarity measure(%)
TOSBAC 25/45	70
HITAC M1××	43
UNIVAC SYSTEM11	35
FACOM M1××	31
UNIVAC U1100	28

Table 5 A table for the example of the common item inference

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
d_1	α_1	β_1	$\gamma_1\gamma_2$	δ_3	ε_1	ζ_2	η_1	θ_1
d_2	α_1	β_1	γ_1	δ_3	ε_1	$\zeta_2\zeta_4$	η_1	θ_2
d_3	α_2	β_1	$\gamma_1\gamma_2$	δ_2	ε_1	ζ_2	η_2	θ_1
d_4	α_2	β_1	γ_1	δ_1	ε_1	$\zeta_2\zeta_3$	η_3	θ_2
d_5	α_1	β_1	$\gamma_1\gamma_2$	δ_1	ε_2	$\zeta_1\zeta_4$	η_4	θ_1
d_6	α_1	β_1	$\gamma_3\gamma_4$	δ_3	ε_1	ζ_2	η_1	θ_1
d_7	α_2	β_1	γ_4	δ_3	ε_3	$\zeta_2\zeta_3$	η_3	θ_2
d_8	α_1	β_2	γ_2	δ_2	ε_1	$\zeta_3\zeta_4$	η_1	θ_1
d_9	α_2	β_3	$\gamma_1\gamma_2$	δ_3	ε_2	$\zeta_1\zeta_3$	η_1	θ_1

In Table 4 we show some examples of similarity measure between strings.

4. Common item inference

In this section, for simplicity we discuss about the common item concerning the documents of rank 1 in Table 2.

As shown in Fig.2, for each document d_i $(1 \leqq i \leqq n)$ attribute values corresponding to at most m attributes A_1, A_2, $\cdots$, A_m are given. Here, for the document d_i, let d_{ij} be the set of attribute values corresponding to A_j and let $\hat{A}_j$ be the set of all attribute values in A_j Namely,

$$d_{ij} = \{a_{ij_1}, a_{ij^2}, \cdots, a_{ij_n}\} \quad (d_{ij} \subseteq \hat{A}_j) \tag{9}$$

Then document d can be represented as follows.

$$d_i = (d_{i_1}, d_{i_2}, \cdots, d_{i_m}) \tag{10}$$

Let

$$\hat{A}^1 = \{a_{q_1 r_1}, a_{q_2 r_2}, \cdots, a_{q_t r_t}\} \tag{11}$$

be a set of attribute value input to retrieve the desired document and

$$N_1 = \{q_1, q_2, \cdots, q_t\}$$

be a set of subscripts in righthand side of (11).

And

$$N_s = \{1, 2, \cdots, m\} - N_1$$

denotes the set of subscripts in non input attribute value. Moreover let

$$D^1 = \{d_1^1, d_2^1, \cdots, d_p^1\}$$

be the set of documents in rank 1, that is, for each $q_i \in N$, $a_{q_i r_i} \cap d_{jq_i} \neq \phi$ for any j ($1 \leqq j \leqq P$).

Then we say attribute value $a_{gr} \in \hat{A}_q$ is common item or common attribute if the following is satisfied.

$$\frac{|D_{qr}^1|}{|D^1|} = C_{qr} \geqq t \quad (t \text{ is threshold}) \tag{12}$$

, where $D^1 = D_{qr}^1 \cup \overline{D}_{qr}^1$ $(D_{qr}^1 \cap \overline{D}_{qr}^1 = \phi)$ and D_{qr}^1 denotes a set of documents whose each elements has attribute value a_{q_r}. Moreover,

for each $d_\mu^1 \in D_{qr}^1$, $a_{qr} \in d_{\mu q}^1$ and

for each $d_\nu^1 \in \overline{D}_{qr}^1$, $a_{qr} \notin d_{\nu q}^1$.

Here, for a set U, $|U|$ denotes the cardinal number. c_{q_r} in (12) implies a appearence ratio of attribute value a_{qr}.

4.2 Evaluation of common item

For a set of attribute values $\hat{A}_q$ $(q \in N_s)$, $\{a_{q1}, a_{q2}, \cdots, A_{qu}\}$ be the set of common items. That is, for each r ($1 \leqq r \leqq qu$) $c_{qr} \geqq t$ is satisfied.

Further let

$$T(q) = \{c_{qr} \mid 1 \leqq r \leqq qu\}$$

be the set of appearance ration of common item and for a σ ($1 \leqq \sigma \leqq |N_s|$)

$$T = T(q_1) \times T(q_2) \times \cdots\cdots \times T(q_\sigma) \tag{13}$$

be a direct sum of $T(q_i)$ $(1 \leqq i \leqq \sigma, q_i \in N_s)$. Now, for a $\delta = (c_{q_1 v_1}, c_{q_2 v_2}, \cdots, c_{q_\sigma v_\sigma}) \in T$, totally evaluation of the tuple of common item $(a_{q_1 v_1}, a_{q_2 v_2}, \cdots, a_{q_\sigma v_\sigma})$ can be defined as follows.

$$J(\delta) = \frac{1}{|N_s|} \sum_{k=1}^{\sigma} c_{q_k v_k} \tag{14}$$

In the common item module, if there doesn't exist a desired document in D_1, firstly common items are searched and in the next rank the documents whose totally evaluated value of the tuple of common items are comparatively large are retrieved. We give a example of retrieval of the document using the common item inference. We assume a set of documents shown in Table 1 and attribute values $\{\alpha_1, \alpha_2\}$, $\{\beta_1\}$ and $\{\gamma_1, \gamma_2\}$ are input for attribute A_1, A_2 and A_3 respectively. If the threshould t in (12) is 0.5, common items are $\varepsilon_1 (c_{51} = \frac{4}{5})$, $\zeta_2 (c_{52} = \frac{4}{5})$, and $\theta_1 (c_{81} = \frac{3}{5})$, where number in the parenthesis shows the appearance ratio of the common item. Thus, total evaluation

value of the document d_6 and d_7 in the next rank are $\frac{1}{5}(\frac{4}{5}+\frac{4}{5}+\frac{3}{5})=\frac{11}{25}$ and $\frac{1}{5}(\frac{4}{5})=\frac{4}{25}$ respectively.

5. Conclusions

In this paper we have proposed a basic idea of two inference methods for database systems and describe a system construction including the inference methods. How decide the optimal function W_c, W_r, g, f and parameters α, β, t, etc., is the problem to be considered in the funture. By using the system, the user will be stimulated his volition to operate the system successively and enable to search his requirement. To testify this fact, we must really develope such system and evaluate it.

References

1) B. A. Wagner, : "The string-to-string correction problem", JACM, Vol.21, No.1, pp.168-173 (1974).
2) T. Okuda, E. Tanaka and T. Kasai, : "A method for the correction of garbled words based on the levenshtein metric", IEEE, Computers, Vol. C-25, No.2, pp.172-177 (1976).
3) N. V. Findler and J. V. Leeuwen, : "A family of similarity measure between two strings", IEEE, PAMI-1, No.1, pp.116-118 (1979).
4) T. Ito and M. Kizawa, : "Spelling correction on a hierachically organized file", IECE of Japan Trans., Vol.J65-D, No.8, pp.1090-1091 (1982).