Text Matching by Dynamic Programming

SEIJI ISHIKAWA*, TAKAFUMI MATSUDA**, KOUICHI SHINOHARA***,
HIDETOSHI FUTAMURA****, KATSUYA MATSUNAGA**** and HIROSHI MORI**

One of the major areas of historical research is to analyze the concordance between similar texts. We have developed a system which analyzes concordance between texts automatically. The system examines the correspondence between each pair of texts by employing dynamic programming, and then unites all the results by merging to produce an output form where the texts are juxtaposed and the words common in the texts are put in order. It also provides a numerical expression for the degree of concordance between texts. The system was applied to three Latin texts and gave satisfactory results.

Introduction

One of the major areas of historical research is the analysis of concordance between similar texts, where these texts are individual interpretations of a common original document. The purpose of concordance analysis is typically to aid the reconstruction of original documents, when these are lost, or to aid the giving of historical interpretations to any differences between texts.

This kind of text processing was formerly performed manually which required great effort and much time, but the progress of computer technology in recent years has made possible the automation of this work. Genicot [1] was an early worker in the field of computerised concordance analysis. In his paper he has analyzed the concordance between fourteen different texts, although he never refers to the algorithm.

We have developed a system for the computer analysis of concordance between texts. In the first step, it analyzes concordance between two texts chosen from n ($n=2, 3, 4, \ldots$) texts using dynamic programming. Although dynamic programming is already used in text matching [2], we employ it hierarchically to find correspondence between texts. Each pair out of n texts are processed, and the results are then merged in the second step to obtain n arranged texts. The system also provides a numerical measure between 0 and 1 for the degree of concordance between the n texts.

Matching Two Texts by DP

A text is composed of text elements such as paragraphs, sentences, words, or letters. We analyze concordance between texts by examining concordance between text elements hierarchichally; thus, we examine concordance between the paragraphs of one text and the paragraphs of the other to analyze concordance between the texts; we examine concordance between the sentences of one paragraph and the sentences of the other to analyze concordance between the paragraphs; and so on.

In the following are given the definitions of concordance between each text element. Let us denote the letters of a word w_k by l_{ki} ($i=1, 2, \ldots, m(w_k)$) and the letters of a word w_l by l_{lj} ($j=1, 2, \ldots, n(w_l)$), i.e.,

$$w_k: l_{k1}l_{k2}\cdot\cdot\cdot\cdot\cdot l_{km},$$
$$w_l: l_{l1}l_{l2}\cdot\cdot\cdot\cdot\cdot l_{ln}.$$

Here we denote $m(w_k)$ by m and $n(w_l)$ by n for simplicity. We assume that $m \geqq n$ without loss of generality. The degree of concordance c^l_{ij} between letters l_i and l_j is defined by

$$c^l_{ij}=\begin{cases}1 \ldots l_i=l_j\\ 0 \ldots l_i\neq l_j.\end{cases}$$

For the purpose of analyzing the concordance between w_k and w_l, let us consider a $m\times n$ matrix M^w as in Fig. 1 whose (i, j) element is $c^l_{ki,lj}$ (abbreviated as c^l_{ij}). If w_k and w_l are the same word, all diagonal elements of M^w are equal to 1 (Fig. 1(a)), but if they are different, 0's and 1's are scattered on M^w (Fig. 1(b)). The problem of analyzing concordance between two words can therefore be transformed into that of finding a path on M^w which passes through as many '1' elements as possible (Fig. 1(c)). It is a maximum path search problem on the matrix M^w. Since correspondence of each letter between two words should be neither overlapped nor reversed, a path is placed with its direction restricted as indicated in Fig. 1(d). This means that a path goes from the upper-left to the lower-right on the matrix. If a path on M^w is denoted by $j=j(i)$, the degree of concordance c^w_{kl} between w_k and w_l is defined by

$$c^w_{kl}=\max_{j=j(i)}\{\sum_{j=j(i)} c^l_{ij}\}/m. \qquad (1)$$

*Department of Computer Science, Faculty of Engineering, Kyushu Institute of Technology, Tobata-ku, Sensui-cho 1-1, Kitakyushu 804, Japan.
**Department of Occidental History, Faculty of Literature, Kyushu University.
***Hitachi Co. Ltd.
****Department of Psychology, Faculty of Literature, Kyushu University.

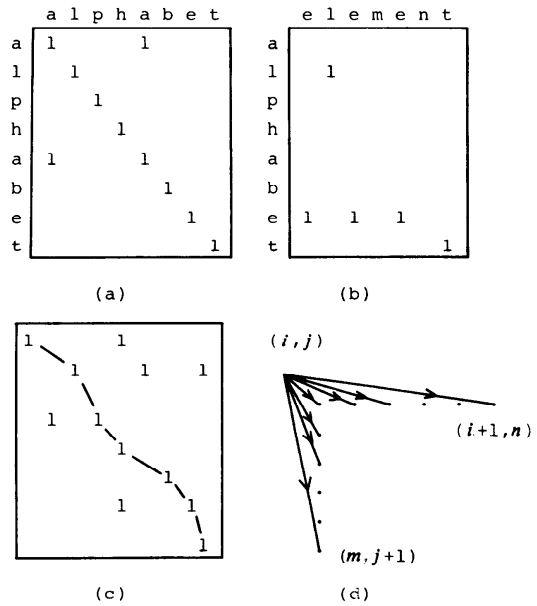

Fig. 1 A matrix expressing the concordance between letters (a) the case where two words coincide with each other (b) the case of two different words (c) the maximum path and (d) the restriction placed on the direction of a path.

Obviously, $0 \leqq c_{kl}^{w} \leqq 1$, and in the case of Fig. 1(a), $j=i$ and $c_{kl}^{w}=1$.

The maximum path search problem expressed by (1) can be solved by dynamic programming. Let us define a function f_{ij} in the form

$$f_{mn}=c_{mn}^{l},$$

$$f_{ij}=c_{ij}^{l}+\max\{f_{i+1,j+1}, f_{i+1,j+2}, \cdots, f_{i+1,n}, f_{i+2,j+1}, f_{i+3,j+1}, \cdots, f_{m,j+1}\}, \quad (2a)$$

where $i=0, 1, 2, \ldots, m$ and $j=0, 1, 2, \ldots, n$. For conveniency, $i=0$ and $j=0$ are included. Then c_{kl}^{w} is given by

$$c_{kl}^{w}=f_{00}/m = [c_{00}^{l}+\max\{f_{11}, f_{12}, \ldots, f_{1n}, f_{21}, f_{31}, \ldots, f_{m1}\}]/m, \quad (2b)$$

where $c_{00}^{l} \equiv 0$. Therefore all that has to be done to solve (1) is to apply (2) to the elements of matrix M^{w}. Table 1 shows some examples of c_{kl}^{w}.

By employing the value of c_{kl}^{w} obtained from the above procedure, we define the degree of concordance between two sentences. Suppose a sentence s_k contains words w_{ki} $(i=1, 2, \ldots, m(s_k))$, and a sentence s_l contains words w_{lj} $(j=1, 2, \ldots, n(s_l))$, i.e.,

$$s_k: w_{k1} w_{k2} \cdots w_{km},$$

$$s_l: w_{l1} w_{l2} \cdots w_{ln}.$$

Here, for simplicity, $m(s_k)$ and $n(s_l)$ are denoted by m and n, respectively. An $m \times n$ matrix M^{s} is a matrix whose (i,j) element is $c_{ki,lj}^{w}$ $(\equiv c_{ij}^{w})$ (See Fig. 2). The

Table 1 Numerical examples of the degree of concordance between two words.

words	c_{kl}^{w}
search search	1.00
concord concorded	0.78
succeed success	0.71
experiment implement	0.60
fantasy Faraday	0.57

	He	will	go	to	school	today
He	1.00					
went	0.25	0.25		0.25		0.20
to			0.50	1.00	0.17	0.40
school		0.17	0.17	0.17	1.00	0.17
today			0.20	0.40	0.17	1.00

Fig. 2 An example of a matrix M^{s} whose elements represent the degree of concordance between two words.

Table 2 Numerical examples of the degree of concordance between two sentences.

sentences	c_{kl}^{s}
He went to school today. He will go to school today.	0.71
There are airports in all the main cities. Some airports are in the main cities.	0.66
Sed compone prius mentis utae cognitionem, et intelliges veritatem. sed conpone prius mentis tuae cognitionem, et intellege ueritatem.	0.92

following expression gives the definition of the degree of concordance between two sentences.

$$c_{kl}^{s}=\max_{j=j(i)}\{\sum_{j=j(i)} c_{ij}^{w}\}/m. \quad (3)$$

Clearly, $0 \leqq c_{kl}^{s} \leqq 1$. The DP formulation for solving (3) is given in the form

$$f_{mn}=c_{mn}^{w},$$

$$f_{ij}=c_{ij}^{w}+\max\{f_{i+1,j+1}, f_{i+1,j+2}, \cdots, f_{i+1,n}, f_{i+2,j+1}, f_{i+3,j+1}, \cdots, f_{m,j+1}\}, \quad (4a)$$

where $i=0, 1, 2, \ldots, m$ and $j=0, 1, 2, \ldots, n$. The value c_{kl}^{s} is then given by

$$c_{kl}^{s}=f_{00}/m = [c_{00}^{w}+\max\{f_{11}, f_{12}, \ldots, f_{1n}, f_{21}, f_{31}, \ldots, f_{m1}\}]/m, \quad (4b)$$

where $c_{00}^{w} \equiv 0$. Some examples of c_{kl}^{s} are given in Table 2.

The same argument as above is applicable to the concordance between two paragraphs and the con-

cordance between two texts. In the former case, if we denote $m(p_k)$ by m and $n(p_l)$ by n, the degree of concordance c^p_{kl} in reference to two paragraphs

$$p_k: s_{k1}s_{k2}\cdot\cdot\cdot\cdot\cdot s_{km},$$
$$p_l: s_{l1}s_{l2}\cdot\cdot\cdot\cdot\cdot s_{ln},$$

is defined by

$$c^p_{kl}=\max_{j=j(i)}\{\sum_{j=j(i)} c^s_{ij}\}/m. \tag{5}$$

Here k_i $(i=1, 2, \ldots, m)$ and l_j $(j=1, 2, \ldots, n)$ are abbreviated as i and j, respectively. The DP formulation for solving (5) is given by

$$f_{mn}=c^s_{mn},$$
$$f_{ij}=c^s_{ij}+\max\{f_{i+1,j+1}, f_{i+1,j+2}, \cdots, f_{i+1,n}, f_{i+2,j+1}, f_{i+3,j+1}, \cdots, f_{m,j+1}\},$$
$$c^p_{kl}=f_{00}/m=[c^s_{00}+\max\{f_{11}, f_{12}, \cdots, f_{1n}, f_{21}, f_{31}, \cdots, f_{m1}\}]/m, \tag{6}$$

where $i=0, 1, 2, \ldots, m$, $j=0, 1, 2, \ldots, n$, and $c^s_{00}\equiv 0$.

On the other hand, if we denote $m(t_k)$ by m and $n(t_l)$ by n, the degree of concordance c^t_{kl} between two texts

$$t_k: p_{k1}p_{k2}\cdot\cdot\cdot\cdot\cdot p_{km},$$
$$t_l: p_{l1}p_{l2}\cdot\cdot\cdot\cdot\cdot p_{ln},$$

is defined by

$$c^t_{kl}=\max_{j=j(i)}\{\sum_{j=j(i)} c^p_{ij}\}/m. \tag{7}$$

Here k_i $(i=1, 2, \ldots, m)$ and l_j $(j=1, 2, \ldots, n)$ are abbreviated as i and j, respectively. The DP formulation for solving (7) is given in the form

$$f_{mn}=c^p_{mn},$$
$$f_{ij}=c^p_{ij}+\max\{f_{i+1,j+1}, f_{i+1,j+2}, \cdots, f_{i+1,n}, f_{i+2,j+1}, f_{i+3,j+1}, \cdots, f_{m,j+1}\},$$
$$c^t_{kl}=f_{00}/m=[c^p_{00}+\max\{f_{11}, f_{12}, \cdots, f_{1n}, f_{21}, f_{31}, \cdots, f_{m1}\}]/m, \tag{8}$$

where $i=0, 1, 2, \ldots, m$, $j=0, 1, 2, \ldots, n$, and $c^p_{00}\equiv 0$. From (5) and (7), it is clear that $0\leqq c^p_{kl}, c^t_{kl}\leqq 1$.

The analysis of concordance between two given texts is therefore performed by employing the equations (1) through (8) hierarchically. We call the method *DP matching*.

Juxtaposing Texts by Merging

In order to distinguish the difference between texts, it is useful to juxtapose all the texts and to put the corresponding words in order. For that purpose, we unite all the results that are obtained by DP matching. We call the procedure *merging*. As the result of having applied DP matching to two texts or text elements, a table is obtained which shows correspondence between text elements. We call it *a match table*. If we confine the argument to the case of sentences, a match table obtained by applying DP matching to two sentences shows which word of the first sentence coincides with which word of the second. If n texts or text elements are processed by DP matching, ${}_nC_2$ match tables are produced. What should be merged are the contents of these match tables.

Here we explain the process of merging by an example. Let us suppose $n=3$, and let the texts or the text elements be denoted by u_1, u_2, and u_3. We also denote the text elements composing them by v_{1i} $(i=1, 2, \ldots, 5)$, v_{2j} $(j=1, 2, \ldots, 5)$, and v_{3k} $(k=1, 2, \ldots, 8)$, respectively, i.e.,

$$u_1: v_{11}v_{12}v_{13}v_{14}v_{15}$$
$$u_2: v_{21}v_{22}v_{23}v_{24}v_{25}$$
$$u_3: v_{31}v_{32}v_{33}v_{34}v_{35}v_{36}v_{37}v_{38}.$$

After having applied DP matching to each pair of u_1, u_2, and u_3, we obtain the following match tables;

$$u_1: 2\ 5 \quad u_2: 2\ 3\ 5 \quad u_1: 1\ 2\ 3\ 5$$
$$u_2: 3\ 5, \quad u_3: 2\ 4\ 6, \quad u_3: 1\ 4\ 5\ 6. \tag{9}$$

Here only the second subscript j of a text element v_{ij} is shown for simplicity. The match table, say the first one of (9), indicates that v_{12} of u_1 corresponds with v_{23} of u_2, while v_{15} of u_1 corresponds with v_{25} of u_2. The process of merging produces a match table

$$u_1: 0\ 2\ 5$$
$$u_2: 2\ 3\ 5$$
$$u_3: 2\ 4\ 6 \tag{10}$$

from the first and the second match table of (9). The third match table of (9) and (10) are then merged to yield

$$u_1: 1\ 0\ 2\ 3\ 5$$
$$u_2: 0\ 2\ 3\ 0\ 5$$
$$u_3: 1\ 2\ 4\ 5\ 6. \tag{11}$$

For performing these procedures, we only have to compare the order of the numbers (except '0') in each row between a given and a new match table, and add the content of the former to the latter by putting a number to an appropriate column or inserting a column in a proper position, so that no contradiction may occur between the match tables with respect to correspondence among text elements. The numbers are always required to be in ascending order. The remaining numbers not appearing in (11) are inserted in the second step to yield

$$u_1: 1\ 0\ 0\ 0\ 2\ 3\ 4\ 0\ 5\ 0\ 0$$
$$u_2: 0\ 1\ 2\ 0\ 3\ 0\ 0\ 4\ 5\ 0\ 0$$
$$u_3: 1\ 0\ 2\ 3\ 4\ 5\ 0\ 0\ 6\ 7\ 8, \tag{12}$$

or, if we express (12) by v_{ij},

$$u_1: v_{11}\cdot\ \cdot\ \cdot\ v_{12}v_{13}v_{14}\cdot\ v_{15}$$
$$u_2: \cdot\ v_{21}v_{22}\cdot\ v_{23}\cdot\ \cdot\ v_{24}v_{25}$$
$$u_3: v_{31}\cdot\ v_{32}v_{33}v_{34}v_{35}\cdot\ \cdot\ v_{36}v_{37}v_{38}$$

which gives us the final result.

Algorithms of the System

The text processing system proposed in this paper is composed of four parts as shown in Fig. 3. In the first step, each text is typed in from a keyboard and forms a data file. Sentences of each file are then arranged by a program so that each sentence starts from the head of a line in the file. A line contains 80 characters at most. In the second step, concordance of texts is examined by DP matching according to the previously stated procedure. The results obtained in the second step are merged in the third step according to the method explained formerly by an example. In the last step, n texts are juxtaposed and corresponding words are put in order in a n-line buffer which contains n lines by referring to the result obtained from merging. The content of the n-line buffer is then printed out. Therefore each of n matched texts is printed out every 80 characters at most.

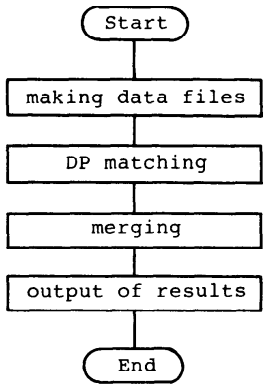

Fig. 3 A flow chart of the text processing system.

Since the main part of the system is the DP matching step, we state its algorithm precisely. Fig. 4 shows a flow chart of the DP matching process. We suppose that the information concerning correspondence of paragraphs among texts is already given. This restriction is placed to reduce the number of calls to the DP routine. In fact, the algorithm in its general form has a four level nesting structure, and if we suppose that each text has n paragraphs, each paragraph has n sentences, and each sentence has n words, then the number of calls to the DP routine is $O(n^6)$.

Another simplification is that, in calculating the degree of concordance between two words, the algorithm examines concordance between two letters one by one from the head of each word instead of applying DP matching to them. In the procedure, we take the minimum discrepancy between two words into account. If the nth ($n=1, 2, \ldots$) letter of a word (say w_i) does not coincide with the nth letter of the other word (w_j), then the $n+1$th letter of w_i and the $n+1$th letter

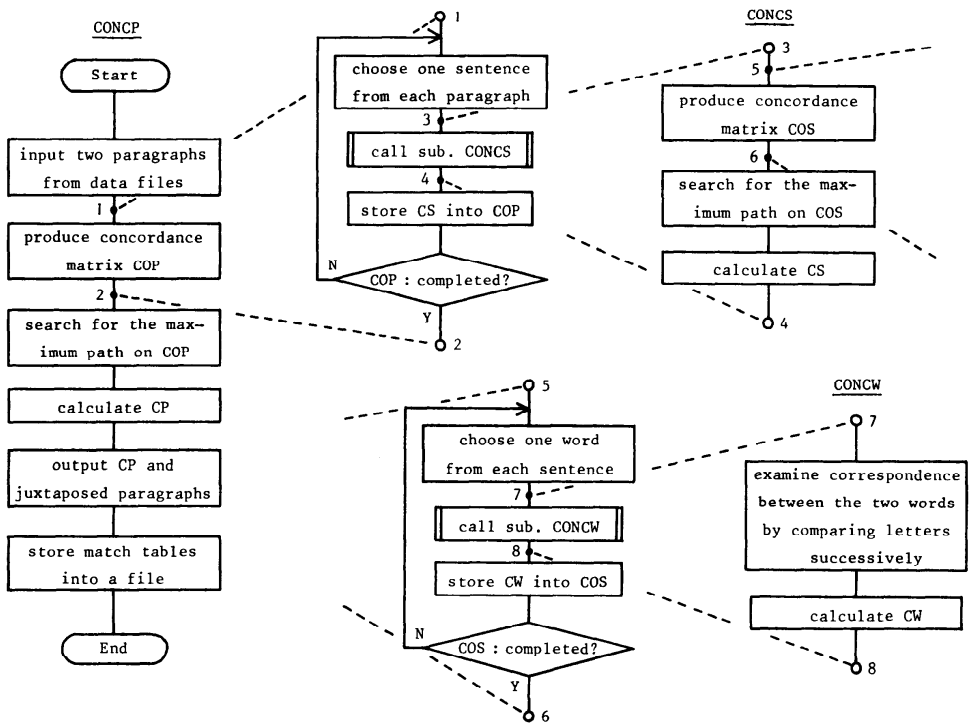

Fig. 4 A flow chart of DP matching.

of w_j are compared with each other. If they do not coincide, then the nth letter of $w_i(w_j)$ is compared with the $n+1$th letter of $w_j(w_i)$. If they coincide with each other, then the next letter of w_i and that of w_j are compared. If they do not coincide or there is no letter left to be compared, the procedure comes to an end and the value of c_{ij}^w is calculated by

$$c_{ij}^w = \frac{n_l}{\max\{l(w_i), l(w_j)\}}$$

instead of (1). Here n_l is the number of coincided letters and $l(w)$ is the length of a word w. This rather simple way of calculating c_{ij}^w suffices for practical use.

In the following are listed the names of the program, subroutines, matrices, and variables in the flow chart of Fig. 4.

CONCP: a program analyzing concordance between two paragraphs,
CONCS: a subroutine analyzing concordance between two sentences,
CONCW: a subroutine analyzing concordance between two words,
COP: a matrix whose elements are c_{ij}^s,
COS: a matrix whose elements are c_{ij}^w,
COW: a matrix whose elements are c_{ij}^l,
CP: a variable representing the degree of concordance between two paragraphs,
CS: a variable representing the degree of concordance between two sentences,
CW: a variable representing the degree of concordance between two words.

Execution and Results

The text processing system was applied to three Latin texts, text 1, 2, and 3, each of which contained about two thousand words. Fig. 5 shows part of the result. The processing time took about an hour for DP matching and 10 minutes for merging including making an output form. The values of the degree of concordance between each pair out of the three texts are

$$c_{12}^t = 0.82, \quad c_{23}^t = 0.86, \quad c_{31}^t = 0.87.$$

The average value is therefore 0.85 which we define as the degree of concordance among the three texts.

Discussion

The obtained results are satisfactory. This kind of text processing, however, needs human modification in some case. Suppose a sentence s_{11} in text t_1 is almost the same as a sentence s_{21} in text t_2. In this case, the two sentences successfully correspond with each other. Suppose then a sentence just after s_{21}, say s_{22}, is linked with s_{21} by, e.g., a comma, and they form a single sentence $s_{21}s_{22}$. In this case, even if s_{11} is almost the same as s_{21}, s_{11} and $s_{21}s_{22}$ do not correspond with each other, provided that s_{22} is not a short sentence compared with s_{21}.

```
1 Commoti                igitur insania cives, tripudiabant in festis
2           commot⟨i⟩ igitur insania,       tripudiabant in
3
1 Dionysii.
2           dyonisii festa.
3                          Commota igitur insania, tripudiabant
1                  Quinto autem mense Laodicius
2                  quinto autem mense, ladicius
3 in dyonisii festa quinto autem mense,        la⟨u⟩dicius
1 proconsul          Appoliniae civitatis venit in
2 proconsul apolloniae          ciuitatis uenit in amfibuli,
3 proconsul apolloniae          ciuitatis uenit in amfibuli,
1 Amphipolim, et sacrificavit          Dionysio. Et      alia
2             et sacrificauit dyonisio.          (et) alia
3             et sacrificauit dyonisio.          Et    alia
1 die nuntiatum est ei de forti milite Christi Mucio, quia multos
2 die nuntiatum est ei de forte milite Christi mucio quia multos
3 die nuntiatum est ei de forte milite Christi mucio quia multos
1 revocasset       de solennitate      Deorum, docens nouas
2          reuocat de sollemnitate,           docens nouas
3 reuocasset       de solemnitate ⟨de deorum⟩ docens nouas
1 seductiones, quibus reverti suaderet ad crucifixum et mortuum
2 seductiones       reuerti       ad crucifixum et mortuum
3 seductiones,      reuerti       ad crucifixum et mortuum
1 hominem: et ista credentes multi, reversi sunt a deorum cultura.
2 hominem.
3 hominem.
1
2 Et ista credentes, multi reuersi sunt a cultura.
3 Et ista credentes, multi reuersi sunt a cultura.
```

Fig. 5 Part of the result.

Although separation of $s_{21}s_{22}$ into s_{21} and s_{22} might overcome the difficulty, it is impossible to judge automatically if it should be separated. Therefore the result obtained by the text processing needs to be modified by a human if necessary.

In the former section, we showed the values of c^t_{ij} $(i, j=1, 2, 3)$, the degree of concordance between two texts. They were calculated by

$$c^t_{ij}=\frac{n_s}{\max\{s(t_i), s(t_j)\}} \tag{13}$$

instead of (7). Here n_s is the number of coincided sentences between a text t_i and a text t_j, and $s(t)$ is the number of sentences contained in a text t. If c^t_{ij} is calculated by (7), then $c^t_{12}=0.61$, $c^t_{23}=0.68$, and $c^t_{31}=0.63$. In a practical sense, the values calculated by (13) show the degree of concordance better than those calculated by (7), since the former gives the percentage of coincided sentences between two texts, while the latter does not.

An alternative way of finding correspondence between texts is successive comparing which compares words one by one from the head of each text. A defect of this method is that, as it finds correspondence among text elements successively, it does not necessarily find optimal matching, while the DP matching method assures that it finds optimal matching if the search area is not limited by a window. In the DP matching process employed in the system, the search area is limited by a window. However, since the texts analyzed have a high degree of concordance, we might expect that an optimal pass exists within the search area. Therefore we may reasonably assume that the analysis found the optimal matching.

The present text processing system can deal with three texts at most. We intend to improve the system so that it can deal with more.

Conclusion

A system was presented for analyzing concordance between texts by a computer. It makes use of DP matching to examine concordance between each pair of texts and unites all the results by merging. The procedure for analyzing concordance between two texts has a hierarchical structure. The system produces arranged text data where texts are juxtaposed and the corresponding text elements are put in order. The system also calculates the value of the degree of concordance between two texts. Three Latin texts were processed by the system and desirable results were obtained. The system used deals with at most three texts. We intend to improve the system so that it can deal with more. The system was developed on the MELCOM-COSMO 800III of the Computer Center of Kyushu Institute of Technology.

References

1. GENICOT, L. Études sur les principautés lotharingiennes, Publications Universitaires de Louvain, Louvain, 1975, 217–271.
2. NAGAO, M. Pattern Information Processing, Corona Publishing Co., Tokyo, 1983 (in Japanese).

(Received July 4, 1983; revised Dec. 5, 1983)