The Multi-list Structure and Information Retrieval

Toshio Nakanishi*

Recently there is a tendency in the information processing by computer to attach special importance to the treatment of mass file. Especially in the real time processing, as it is of vital importance to process any request as soon as it appears, rapid file maintenance has become more and more essential.

In general, when we have gathered information, we classify it into several groups according to the value of some attribute, and keep them in separate files for the efficiency of business, in other words, due to the circumstances that an item of information is inserted and perhaps deleted once, but it will be retrieved many times during its lifetime.

Circumstances are almost the same with the information processing by computer, although in computerized system it is more general to organize a file structure convenient for information retrieval. In case that an item has several attributes, the complete classification of data items is very troublesome and awkward and is also unadvisable from the viewpoint of memory efficiency.

The multi-list structure is a method proposed in place of classification. There are several methods to organize the multi-list file, for example a method by tree[1] (especially ballanced tree[2]). In this article we propose the chaining method to organize multi-list file, and various IR techniques available only in the multi-list structure are discussed. At the end its application in the patent information retrieval system is described.

1. *Definitions of terminology*

The smallest and complete unit of information is called *item,* and a set of related items is called *record.* A collection of related records constitute a file.

As a matter of fact, it is difficult to standardize all information unit and therefore the difinition of item or record does not necessarily apply in every case. Here we consider an item is a fundamental element of a file. In general, an item is divided into two parts, namely *description* and *data.* The description is a set of symbols called descriptors, which define the properties of the information in the item. The description is a part by which the item may be retrieved, and the data is a part by which it need not be retrieved.

This paper first appeared in Japanese in Joho Shori (the Journal of the Information Processing Society of Japan), Vol. 7, No. 3 (1966), pp. 143–151.

* Railway Technical Research Institute.

For the sake of representation in the computer, an item is made up of smaller elements which are called catenas. There are *data catena* and *associative catena* and each corresponds data and description respectively. Fig. 1 illustrates them.

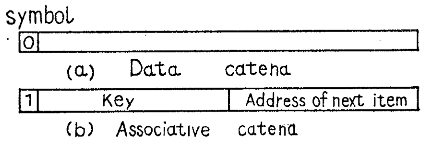

Fig. 1.

An associative catena consists of a key and an address of next item. A key may be one or more descriptors or a part of descriptor. A group of items which have at least one key in common is called *list.*

Generally, one item has several keys and it is often necessary to retrieve all the items which have some keys in common. The multi-list structure is a file organization useful to such retrievals. The data area organized into multi-list structure is called MAA (Multi Association Area).

2. *Organization technique of MAA*

In order to organize MAA, it is nessary to transform each key of item into an address which is used to store the pointer of the list to which the item belongs. Techniques in which trees are used as translator have been discussed. The utilization of balanced tree[2] is an example. A problem of this method is that the values of keys have to be maintained in monotone increasing order in both a tree item and a tree level and therefore ordering of keys in some way is indispensable.

Here we propose the chaining method[3] to organize MAA. the chaining method is considered to be more advantageous in such respects as memory saving and transformation or retrieval speed.

An item in the patent document has several independent sets of descriptors which belong to the same attribute class as shown in Fig. 2.

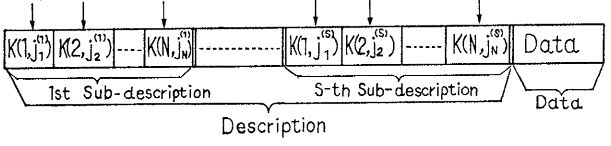

Fig. 2.

Therefore we will call such an independent set of descriptors *sub-description* and our discussion will be developed over such an item which has several independent sub-descriptions covering several attributes.

We will assume as follows:

N: number of attributes.
C_i: attribute class $i=1, 2, \ldots\ldots, N$.
$K(i, j_i^{(l)})$: attribute value, namely descriptor.
$i=1, 2, \ldots\ldots, N$; attribute class designator.
$j_i^{(l)}=1, 2, \ldots\ldots, m_i$: attribute value designator.

In organizing MAA, a key may consist of a descriptor, a set of descriptors or a part of descriptor for the machine convenience.

From now we will assume that an item is represented as shown in Fig. 2 after all relating descriptors have been replaced with keys. In this case the item is represented in MAA as shown in Fig. 3. $A\ (i,\ i_i)$ $(i=1, 2, \ldots\ldots, N)$ is a link address of a key $K(i, j_i)$.

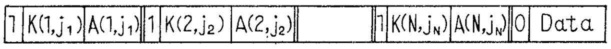

Fig. 3.

The chaining method is as follows.

Every key is transformed into an address by a key-to-address transformation T.

$$T(K(i,\ j_i)=A\ (i,\ j_i)$$

The number of keys M is calculated by

$$M=\sum_{i=1}^{N} m_i$$

The location for $A\,(i,j_i)$ should be more than or at least equal to M. In our application to each transformed address correspond three locations (Fig. 4). These

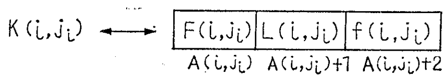

Fig. 4.

contain the location of the first record of the list (FL), that of the last record of the same list (LL) and the number of the records belonging to the list (Fr). These three locations can be reduced to two depending upon the type of memory (random access or serial access.) or storing method (push down list or push up list). Fig. 5 illustrates how the MAA of six records are organized. In this case,

$FL\ (K_1)=FL\ (K_2)=FL\ (K_3)=FL\ (K_4)$
$FL\ (K_5)=FL\ (K_6)=FL\ (K_7)$
$LL\ (K_2)=a_3$
$LL\ (K_1)=LL\ (K_3)=LL\ (K_4)=LL\ (K_5)=LL\ (K_6)=LL\ (K_7)=a_4$

$$F_T(K_1)=4,\ F_T(K_2)=2,\ F_T(K_3)=3,\ F_T(K_4)=3$$
$$F_T(K_6)=3,\ F_T(K_6)=3,\ F_T(K_7)=5.$$

This example shows an extremely simple case and in our system PIRS (Patent Information Retrieval System) far more complicated *MAA* was organized.

3. *Request form and IR efficiency*

The condition of retrieval is called *request.* As a rule, a request is a logical function of several keys. For example, when we want to retrieve all the items which have the same keys with the item illustrated in Fig. 2, the request $R(K)$ is as follows.

$$\begin{aligned}R(K)&=R\{K(1,\ j_1^{(1)}),\ K(2,\ j_2^{(1)}),\ldots\ldots,\ K(N,\ i_N^{(1)}),\\ &\ldots\ldots,\ K(1,\ j^{(S)}),\ K(2,\ j_2^{(S)}),\ \ldots\ldots,\ (K(N,\ j_N^{(S)})\}\\ &=\{K(1,\ j_1^{(1)})\wedge K(2,\ j_2^{(1)})\wedge\ldots\ldots\ K(N,\ j_N^{(1)})\}\wedge K\ \{(1,\ j_1^{(2)})\wedge K(2,\ j_2^{(2)})\wedge\\ &\ldots\ldots\wedge K(N,\ j_N^{(2)})\}\wedge\ldots\ldots\ldots\ldots\ldots\ldots\\ &\{K(1,\ j_1^{(S)})\wedge K(2,\ j_2^{(S)})\wedge\ldots\ldots\wedge K(N,\ j_N^{(S)})\}.\end{aligned}$$

The request should cover all the conditions of retrival, so for the sake of safety $R(K)$ usually takes the form like

$$\begin{aligned}R(K)=\{&K(1,\ j_1^{(1)})\wedge K(2,\ j_2^{(1)})\wedge\ldots\ldots\wedge K(M_1,\ j_{M1}^{(1)})\}\vee\\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\ &\vee\{K(1,\ j_1^{(S)})\wedge K(2,\ j_2^{(S)})\wedge\ldots\ldots\wedge K(M_S,\ j_{MS}^{(S)})\}\\ &(M_i\leqq N,\ i=1,\ 2,\ \ldots\ldots,\ S)\}.\end{aligned}$$

Sometimes we have to handle the request of the form such as $K(1,\ i_1^{(1)})\wedge K(2,\ j_2^{(1)})\wedge \overline{K(3,\ j_3^{(1)})}$.

(1) Information retrieval when the request takes the form of logical product of keys.

We will assume that

$C_1,\ C_2,\ \ldots\ldots,\ C_N$: attribute classes.
$K(1,\ j_1),\ K(2,\ j_2),\ \ldots\ldots,\ K(N,\ j_N)$: keys in request.
$f(1,\ j_1),\ f(2,\ j_2).\ \ldots\ldots f(N,\ j_N)$: frequencies.
$R(K)=K(1.\ j_1)\wedge K(2,\ j_2)\ \ldots\ldots\wedge K(N,\ j_N)$: request.

$R(K)=\sum_{i=1}^{N}\sum_{k=1}^{mi}f(i,\ k)$: total number of items in the file.

The events that the key which belong to C_i is $K(i,\ j_i)$ $(i=1,\ \ldots\ldots N)$ are assumed to be independent one another. When R_f is sufficiently large, the probability $p(i,\ j_i)$ that the key belonging to C_i is $K(i,\ j_i)$, is approximated by

$$p(i,\ j_i)=f(i,\ j_i)/R_f(i=1,\ 2,\ \ldots\ldots,\ N).$$

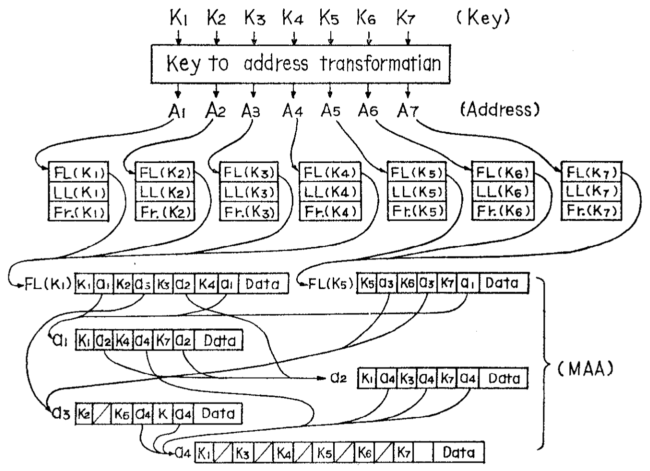
K1 K2 K3 K4 K5 K6 K7 (Key)
Key to address transformation
A1 A2 A3 A4 A5 A6 A7 (Address)
FL(K1) LL(K1) Fr.(K1)
FL(K2) LL(K2) Fr.(K2)
FL(K3) LL(K3) Fr.(K3)
FL(K4) LL(K4) Fr.(K4)
FL(K5) LL(K5) Fr.(K5)
FL(K6) LL(K6) Fr.(K6)
FL(K7) LL(K7) Fr.(K7)
FL(K1) K1 a1 K2 a3 K3 a2 K4 a1 Data
FL(K5) K5 a3 K6 a3 K7 a1 Data
a1 K1 a2 K4 a4 K7 a2 Data
a2 K1 a4 K3 a4 K7 a4 Data
a3 K2 K6 a4 K a4 Data
a4 K1 K3 K4 K5 K6 K7 Data
(MAA)

Fig .5.

No.	Completion date	confir-mation check	Desired completion date	Application date	Requester		
					Laboratory	Name	Tel.

Field name		Additional class	Supplemental class	Sub-division	AND/OR	Application date		Announcement date		Invention/utility model	Application code	Applicant	C	CONTROL
						FROM	UNTIL	FROM	UNTIL					
Column	1 2 3	4	5	6 7 8 9 10 11 12 13	14	15 16 17 18 19 20 21	22 23 24 25 26 27 28	29 30 31 32 33 34 35	36 37 38 39 40 41 42	43	44	45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	75	76 77 78 79 80
Data														
Data														
Data														
Data														
Data														
Data														
Data														
Data														
Data														
Data														

Rem. 1 When the fields 'class', 'additional class', and 'supplementary class' are blank, they are considered to be the same as the previous data. As to the field 'sub-division', even if it is the same as the previous datum one should write the same datum again. The fields 'class', 'additional class' and 'supplementary class' of the data which follow the data with E code in the field 'AND/OR' should not be left blank.

Rem. 2 Specical characters (—, ;, ×) added to the subdivision have special meanings. One of them has to be added without fail to the field 'sub-division'.

(A) 1123;——only 1123 without any number code to follow.

(B) 11× ——Any number code can follow 11.

(C) 11–4——All the number codes of 3 places except 114.

Rem. 3 AND/OR field represents the types of logical combination.

A——Logical product
O——Logical sum
E——End

Up to and including 10 data can be processed simultaneously.

Rem. 4 Application/announcement date.
The application/announcement date falls within [(FROM), (UNTIL)] (including both ends). Blank field of (UNTIL) means after (FROM), and that of (FROM) means before (UNTIL).
When * appears in the first column of this field, it means the data is the same as that of the request data with E in the field AND/OR.

Rem. 5 Invention/utility model
U: utility model
P: invention
Blank: without distinction
*: the same as that of Rem. 4.

Rem. 6 Application code
1: company
2: governmental
3: private
4: foreign
Blank: without distinction
*: the same as that of Rem. 4.

Rem. 7 Applicant
The last character must be ';'. When one field is not enough to write an applicant name or names, one can extend the field to that of the next row. The logical sum of applicants combined by '+', can be processed.

Rem. 8 C
When the field of applicant is extended to the next row, 'C' has to be written in the 'C' field of the first row.

Rem. 9 Col. 76 ~ col. 80 is for operator's use only.

Rem. 10 When the field 'application' code is 4, the name of the country (in Olympic code) should be written in the 'applicant' field without ';' at the end.

Fig. 6. Patent information request data sheet.

We will discuss various techniques of retrieval for the request $R(K)=K(1, j_1)$ $K(2, j_2)\wedge \ldots\ldots \wedge K(N, j_N)$.

(A) one-by -one testing method.

This is the most primitive method of retrieval. All the items are checked one by one if it satisfies the request. In this case the number of memory accesses ν_1 is

$$\nu_1=R_f. \tag{1}$$

The expectation of the number of tests μ_1 is

$$\mu_1=R_f\{(1+p(1, j_1)+p(1, j_1)\cdot p(2, j_2)+\ldots\ldots+p(1, j_1)\cdot p(2, j_2)\cdot\ldots\ldots\cdot p(N-1, j_{N-1})\}. \tag{2}$$

Let us assume that when $p(1, j_1)$, $p(2, j_2)$, ..., $p(N, j_N)$ are arranged in increasing order, we get

$p(i_1, j_1')\leqq p(i_2, j_2')\leqq\ldots\ldots\leqq p(i_N, j_N')$ (Here j_k' is the value j_k corresponding to i_k)

Then μ_1 takes the smallest value when the test of keys is executed in the order C_{i_1}, C_{i_2}, ..., C_{iN}, and it becomes

$$\mu_1=R_f\{1+p(i_1, j_1')+p(i_1, j_1')\cdot p(i_2, j_2')+\ldots\ldots+p(i_1, j_1')\cdot(i_2, j_2')\cdot \ldots\ldots\cdot p(i_{N-1}, j_{N-1}')\}.$$

(B) The method which makes use of MAA characteristics (1).

First of all $K(i_1, j_1')$ is transformed into an address.

$$A(i_1, j_1')=T(K(i_1, j_1')).$$

$A(i_1, j_1')$ contains the address of the location where the first item of the related list is stored, so the items which have the key $K(i, j_1')$ are retrieved by following up the chain. These items are checked for other conditions in the order C_{i2}, C_{i3}, ...C_{iN}.

The number of memory accesses ν_2 is

$$\nu_2=f(i_1, j_1') \tag{3}$$

and the expectation of the number of tests μ_2 is

$$\mu_2=R\{p(i_1, j_1')+p(i_1, j_1')\cdot p(i_2, j_2')+\ldots\ldots+p(i_1, j_1')\cdot p(i_2, j_2')\cdot\ldots\ldots\cdot p(i_{N-1}, j_{N-1}')\}. \tag{4}$$

We can see easily that the method (B) is more advantageous than the the method (A).

(C) The method which makes use of MAA characteristics (2).

All the lists related to the request are followed up and the addresses of the items are checked if they are the same across all the lists. The items common to all the lists are thus retrieved. This method, however, can be available effectively only when the link addresses are maintained in monotone increasing order.

The number of memory accesses ν_3 is

$$\nu_3 = R_f\{1 - \prod_{i=1}^{N-1}[1 - p(i, j_i)]\}$$

and the number of test μ_3 is zero

(2) Information retrieval when the request takes the form of logical sum of keys.

(A) one-by-one testing method.

Let us define ν_i' and μ_i' in the same way with above, then we will get,

$$\nu_1' = R_f$$

$$\mu_1' = R_f\{1 + [p(i_N, j_N')] + [1 - p(i_{N-1}, j_{N-1}')] + \ldots\ldots - \prod_{k=2}^{N}(1 - p(i_k, j_k')]\}.$$

(B) the method which makes use of MAA characteristics.

All the items are taken out by following up all the lists.

$$\nu_2' = R_f\{1 - \prod_{i=1}^{N}(1 - p(i, j_i')\}$$

$$\mu_2' = 0.$$

As to the proof of all the propositions, please refer to the paper of same title in the Journal of JFIP, Vol. 7, No. 3 (in Japanese).

4. *An application to the Patent Information Retrieval System*

On the basis of the above theoretical considerations we have established the patent information retrieval system. The patent bureau gives public notice of approved inventions or utility model patents in a patent journal. Recently. the number of approved patents have remarkably increased and the total number of items with a term of validity of fifteen years becomes enormous.

There are a number of tasks imposed on the automation of patent business. What we aimed at was, first of all, the automatization of searching related patent documents which are necessary for investigation in advance, namely the establishment of an information retrieval system on the basis of master file of patent documents.

The essential tasks of PIRS are the insertion of records (namely the organization of MAA) and the information retrival. Especially in the latter, a very complicated treatment was necessary owing to the characteristica of patent documents.

In writing the program, the greatest emphasis was laid on that of IR.

A request composed of up to and including 10 data can be processed simultaneously. Although the logical structure of a request is quite arbitrary and unpredictable in practice, we aimed to write a program which processes IR most efficiently and which is as simple and general as possible. The retrieval program making use of 'tally' is an accomplishment.

We organized a multi-list file of 1,605 invention and utility model patents which were registered in 1963 and 1964 to the class 77 (vehicle in general), class 78 (railway), class 79 (railway vehicle) and conducted various experiments with satisfactory results. The file was organized in the magnetic tape.

The time required for the insertion of a record was 6 sec. including the time of card reading and printing (3.5 items per page). It took only one min. including the time of printing (2 items per page) to retrieve 13 records which correspond to the request composed of 7 independent data (one datum is composed of 2 logical products and another of 3 logical products).

We confirmed through these experiments the practicability of chaining method applied to the organization of the multi-list structure and the retrieval techniques which make use of chaining characteristics of MAA.

The master file of PIRS now contains only small amount of documents, but it is being extended continually and at present PIRS is very busy with the daily services of patent information retrievals.

Reference

[1] Prywes, N. S. and H. J. T. Colin, The Organization of a Multi-list-type Associative Memory. *IEEE. Trans. on Comm. and E'ec.* Sept. (1963), 488–492.

[2] Landauer, Walter I., The Balanced Tree and Its Utilization in Information Retrieval *IEEE. Trans. on Elec. Comp.* December (1963), 863-871.

[3] Johnson, L. R., On Indirect Chaining Method for Addressing on Secondary Keys. *Comm. of the ACM. 4*, 4 (1961), 218–222.