Implementation of Associative Data Processing System on a Mini-computer

Terumasa HOSOMI*, Toshiro ARAKI*, Kenichi HAGIHARA*
and Naoto HONDA**

1. Introduction:

Associative System 11 (AS-11) has been developed, which is a general purpose data structure processing system with the associative retrieval facility; that is, the function of accessing data through a partial specification of its content. The requests of AS-11 summarized in Table 1 are provided as Macro calls at the assembly language level, and now can be used as FORTRAN subroutines to write a program more easily. AS-11 runs under Disk Operating System of a minicomputer PDP-11/20 equipped with 24KW (1 word = 2bytes = 16 bits) main memory, an 1.2MW disk pack, a console typewrit-

Table 1 Requests of AS-11

Mnemonic	Purpose
.ASINIT	Declares the use of AS-11 to the monitor of PDP-11/20
.ASRLS	Declares the end of using AS-11 to the monitor of PDP-11/20
.AFALOC	Initializes and opens an AF
.AFOPEN	Opens an AF for input and output
.AFREAD	Opens an AF for input only
.AFCLOS	Closes an AF
.MAKEI	Enters an ITEM in IT and its DATUM, if any, in DT
.UPDTI	Updates or deletes a DATUM
.GETIT	Gets information about an ITEM
.DELTI	Deletes an ITEM
.ORDER	Declares that the set of TRIPLEs or PAIRs has a total ordering
.MAKEA	Enters a TRIPLE (PAIR) in TM (PM)
.ATCHI	Attaches a TRIPLE (PAIR) ITEM to a TRIPLE (PAIR)
.DETCI	Detaches a TRIPLE (PAIR) ITEM from a TRIPLE (PAIR)
.GETID	Gets a TRIPLE (PAIR) by specifying a TRIPLE (PAIR) ITEM
.GETOR	Gets the n-th TRIPLE (PAIR) from a set of TRIPLEs (PAIRs) with a total ordering
is-requests	the requests corresponding to FO and GO
all-requests	the requests corresponding to F3 and G1
ret-requests	the requests corresponding to FAi, FOi, FVi (where i=1,2), GH and GT
.NEXT	Continues the unfinished all-request or ret-request
ers-requests	the requests erasing a set of TRIPLEs (PAIRs)
.ERASE	Erases a TRIPLE (PAIR)
.STAT1	Gets the statistics of an AF
.STAT2	Updates the statistics of an AF
.PUSH	Pushes data into the AS-11 stack
.POP	Pops data out of the AS-11 stack
.PEEP	Peeps the n-th content of the AS-11 stack

This paper first appeared in Japanese in Joho－Shori (Journal of the Information Processing Society of Japan), Vol. 16, No. 11 (1975), pp. 974～981.

* Faculty of Engineering Science, Osaka University
** Information System Department, Japan Air Lines Co., Ltd.

er etc. AS-11 is aimed specifically at space efficiency with keeping the retrieval processing time resonable because of being implemented on the minicomputer.

Two basic data structure entities used here are an associative pair (PAIR)[1] and an associative triple (TRIPLE) introduced by J.A. Feldman and P.D. Rovner[2], which provide a user not only an ability for expressing n-ary relation but also a strong and efficient retrieval ability.

2. The AS-11 Data Structure:

Conceptually, an AS-11 data structure consists of a universe of ITEMs (ITEM Table: IT), a universe of DATA (DATA Table: DT), a universe of PAIRs (PAIR Map: PM) and a universe of TRIPLEs (TRIPLE Map: TM). They are stored in the disk as a contiguous file called an Associative File (AF).

Since IT is referred with high frequency, it is resident in main memory while the AF is opened. On the other hand, DT, PM and TM are segmented, and a segment is swapped in on demand. The segment of these is of common size, but the number of segments differs among tables. The segment size and the numbers of segments in various tables are specified at the initialization of the AF. Each segment consists of a number of cells (See Sec. 2.2-2.4.).

An ITEM is an entity referred in terms of 32-bit pattern called an external identifier (EI). An information which a user attaches to a certain ITEM is called a DATUM. Note that any DATUM has no influence on the associative processing. A TRIPLE is an ordered collection of three ITEMs. It is denoted by <a,o,v>. Similarly, a PAIR denoted by <h,t> is an ordered collection of two ITEMs. The symbols a, o, v, h and t can be thought of as mnemonics for attribute (A-component), object (O-component), value (V-component), head (H-component) and tail (T-component) resp. Furthermore a TRIPLE <a,o,v> or PAIR <h,t> itself can become an ITEM. Such an ITEM is called a TRIPLE (or PAIR) ITEM (TI (or PI, resp.)), which can become a component of another TRIPLE or PAIR recursively. In this way, an n-tuple or an n-ary relation can be expressed in general. For example, a PAIR <h,i> will express an ordered collection of four ITEMs <h,a,o,v>, if i=<a,o,v>.

As for TRIPLEs (or PAIRs) there are eight (or four, resp.) primitive associative retrieval requests(PARRs) (Table 2). For example, FA1 <a,o,-> is a PARR which gets all the TRIPLEs that have the specified A-component a and O-component o.

Table 2 Primitive associative retrieval requests

(a) TRIPLE forms

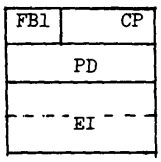
F0	<a,o,v>	F02	<-,o,->
FA1	<a,o,->	FV1	<a,-,v>
FA2	<a,-,->	FV2	<-,-,v>
F01	<-,o,v>	F3	<-,-,->

(b) PAIR forms

G0	<h,t>
GH	<h,->
GT	<-,t>
G1	<-,->

2.1. IT (ITEM Table):

All ITEMs used in an AF are stored in IT which consists of a number of cells (Fig.1). Each ITEM corresponds to a cell. The position of a cell in IT is calculated by hashing the EI of the ITEM. This position is expressed by 13 bits, and this 13 bits binary number is used as its internal identifier (II). Thus, the maximum number of IIs available is 2^{13}.

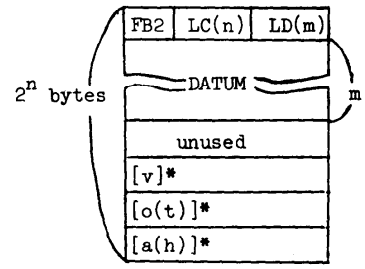


FB1 : function bits
CP : conflict pointer
PD : pointer to DT
EI : external identifier

Fig.1 Structure of a cell in IT.

2.2. DT (DATA Table):

A DATUM is stored in a cell of DT. Since the length of each DATUM is variable, this cell is of variable length, 2^n bytes (where n is a positive integer) (Fig.2). Therefore, the segment of DT is managed using Buddy System[3]. The position of cell in DT is indicated by the PD field of the cell in IT.



FB2 : function bits
LC : length of a cell
LD : length of a DATUM

* AS-11 automatically uses the three (or two) words at the bottom of a cell if the ITEM which has this DATUM is a TI (or PI, resp.).

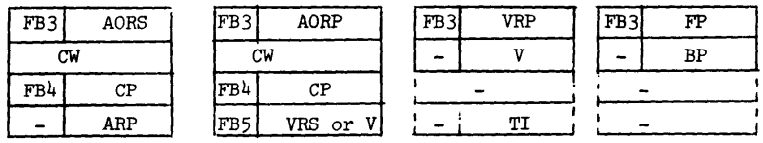
Fig.2 Structure of a cell in DT.

2.3. TM (TRIPLE Map):

TM is composed of Attribute Table (AT), Object Table (OT) and Value Table (VT). The following is the explanation about AT: This description applies equally to OT (or VT) by replacing A, O and V with O, V and A (or V, A and O, resp.), cyclically.

The TRIPLEs with the same A-component are placed in the same segment of AT, so that only one segment need be brought into main memory to fully answer the PARRs specified by forms FA1 and FA2. There are six kinds of cells in AT: A-header-cell (AHC), AO-header-cell (AOHC), 2-words-value-cell (2WVC), 4-words-value-cell (4WVC), 2-words-free-cell (2WFC) and 4-words-free-cell (4WFC) (Fig.3). The function bits (FB3) discriminate among them. The position of AHC or AOHC is calculated by hashing the II of A-component or the IIs of both A-component and O-component, resp. The conflicts of AHCs and AOHCs are resolved by the direct chaining method which uses the CP field as a conflict pointer. If the position where a conflict start cell should be placed is already occupied by another cell which is not a conflict start cell, then the latter is copied to an adequate free cell; and the former occupies its position. The function bits (FB4) indicate whether the cell is a conflict start cell or not, etc. If the

PARR FA1 is issued, then this system gets the position of AOHC by hashing the IIs of both A-component and O-component, and checks that this AOHC is a desired cell by computing the A-component and O-component of this cell from its position and the word in the CW field which is calculated from A-component and O-component. Fig. 4 depicts an outline of TM. The TRIPLEs represented are: $\langle A_1,O_1,V_1\rangle$, $I_1=\langle A_1,O_1,V_2\rangle$, $\langle A_2,O_2,V_2\rangle$, $I_2=\langle A_2,O_3,V_3\rangle$.



FB3(4,5): function bits
ARP : A-ring-pointer
AORS(P) : AO-ring-start(pointer)
CW : check word
CP : conflict pointer
VRS(P): V-ring-start(pointer)
V : value
TI : TRIPLE ITEM
F(B)P : forward(backward) pointer
- : unused

Fig.3 Structure of cells in TM.

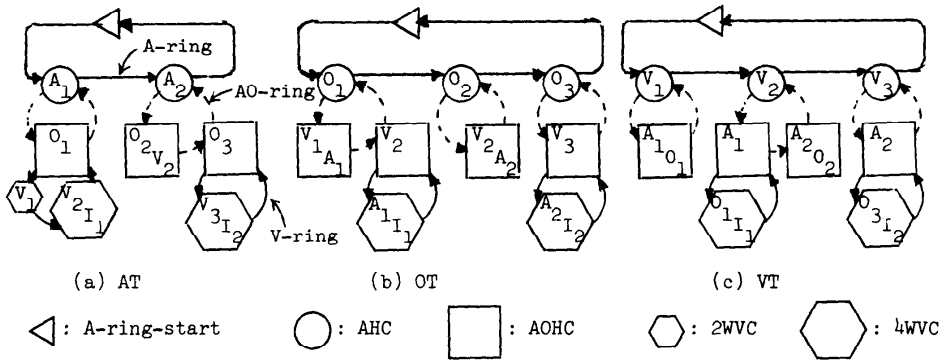


Fig.4 Outline picture of TM.

2.4. PM (PAIR Map):

PM, which is similar to TM, is composed of Head Table (HT) and Tail Table (TT). HH (or TT) is used to answer the PARR specified by form GH (or GT, resp.). There are five kinds of cells in HT: H-header-cell (HHC), 2-words-tail-cell (2WTC), 4-words-tail-cell (4WTC), 2WFC and 4WFC. HHC takes a middle role between AHC and AOHC. 2(or 4)WTC is same as 2(or 4, resp.)WVC. TT is analogous to HT.

3. Special Features of AS-11:

System programs of AS-11 consist of resident modules and overlay modules. Overlay modules are the same size, 1KB. AS-11 runs without any trouble provided that there are 1.5KB resident areas, at least 5KB overlay area, and the data area which can admit IT and at least 3 segments per one AF. Many AFs can be opened at the same time.

AS-11 has several facilities which enable a user to use the space of main memory and secondary storage efficiently, to increase time efficiency and to write a program

easily as follows: (1) According to the characteristics of a user's application, the user can specify the size of segment (= 256×n words where n is a positive integer) and the number of segments of DT, AT, OT and VT resp., which will make the required data area small. (2) In general, there are three copies of each TRIPLE: a copy in AT, one in OT and one in VT. Thus any PARR can be processed time-efficiently. But a certain application may not need some of the PARRs, say FV1 and FV2; then it is unnecessary to enter the TRIPLEs in VT. AS-11 enables a user not to enter some of the TRIPLEs in all the three tables which will save the time of entry and, furthermore, not to establish some of the three tables which will cut down the unnecessary secondary storage area. This function is used similarly in the case of PM. (3) System down will not occur even if the number of TRIPLEs or PAIRs in a segment comes to be too large at execution time because AS-11 has two spare segments called OverFlow Tables (OFT) in which the overflowed TRIPLEs or PAIRs are placed. (4) It is advisable not to use OFT since the usage of OFT deteriorates the time efficiency of retrieve. A utility program called AFPIP is prepared which restructures the overflowed AF to fit into a non-overflowed AF with changing various parameters, such as the segment size and the number of segments on each table. The request .STAT1 will tell whether OFT is used or not, and enable a user to know the profile of AF usage. (5) AS-11 has two software stacks whose elements are of variable length in the disk. This stack will make it easy to write a recursive program etc. (6) At the request level, total orderings can be introduced into the sets of TRIPLEs with the same A and O (or O and V, V and A, resp.) components or PAIRs with the same H (or T, resp.)-component. (7) There are some set manipulation requests[4] implemented, though not shown in Table 1.

References:

[1] K. Furukawa and S. Yamazaki: EDSP: A general purpose data structure manipulating system, Bul. Electrotech. Lab., Vol.37, No.1,2, pp.91-100 (1973), in Japanese.

[2] J.A. Feldman and P.D. Rovner: An ALGOL-based associative language, Comm. ACM, Vol.12, No.8, pp.439-449 (1969).

[3] D.E. Knuth: The art of computer programming Vol.1 / Fundamental algorithms, Addison Wesley, pp.442-445 (1968).

[4] AS-11 user's manual, Kasami Lab. Osaka University (1974), in Japanese.