論　文

ミニコンによるある連想記憶システムの作成について*

細見輝政** 荒木俊郎** 萩原兼一** 本田直人***

Abstract

AS-11 (Associative System-11) has been developed, which is a general purpose data structure processing system with the associative function, the accessing of data through a partial specification of its content. AS-11 has two data types; "associative pair" and "associative triple" introduced by J. A. Feldman et al, which provide the user not only a strong and efficient retrieval ability but also a ability for expressing n-ary relations. The remarkable features of this system are as follows; (1) the asymmetric entry is possible; (2) the user can introduce the ordered relation into the set of associative triples or pairs; (3) the table is able to be made handy in size.

This system runs on a mini-computer PDP-11/20 equiped with 24 K core memory, 1.2 M disk-pack, teletypewriter, and etc. This paper gives the outline of AS-11.

1. まえがき

図形処理，オペレーティング・システムあるいは人工知能などにおけるデータ構造の取り扱いに関する大きな問題は情報の検索である．連想記憶方式はこれに対する方法の一つである．この方式をハードウェアで構成したもので小規模なものは大型計算機のページング機構などに取り入れられているが，一般のデータ処理に対応できるような大規模なものは現われていない．この機能をソフトウェアで実現して，汎用あるいは図形処理などの特定の目的のデータを扱えるようにしたものは数多く提案，作製されており，汎用のものでは，ASP[1], LEAP[2], EDSP[3] などが知られている．

一般にデータ間の関係はポインタまたは位置関係を用いて表現されている．前者の方法は複雑な関係を表現するのに適している反面，あるデータと特定の関係をもつデータを求めることが比較的煩雑である．また後者の方法はハッシュ技法[7] に代表されるように迅速に目的のデータを得ることができる．LEAP などにみられる連想三つ組 (associative triple) はこの両者の方法を組み合わせてデータ相互の関係を表現し，ハッシュにおける検索速度の向上とポインタによるデータ間の関係の表現の“豊かさ”を折衷したものと言えよう．

さらに，三つ組を形成している三つの要素は互いに対等に扱われており，どの要素からの検索もすべて同様の速さ，手続きで行うことができ，これが“連想”三つ組と呼ばれる所以となっている．

我々はこの連想三つ組を基本として非対称登録も可能とし，連想三つ組間に全順序関係の導入を可能とするなど，従来のシステムの機能をさらに拡張した汎用データ構造処理システム AS 11 (Associative System 11) を作製した．AS 11 は小型計算機 PDP 11/20 または 45 の上で稼動するように設計されている****．

本稿では，小型計算機に伴う制約（主として主記憶の制限）の下での実現という見地から，AS 11 の概要を紹介し，稼動状況などについても若干報告する．

---

* Implementation of associative data processing system on a mini-computer system by Terumasa HOSOMI, Toshiro ARAKI, Kenichi HAGIHARA (Faculty of Engineering Science, Osaka University) and Naoto HONDA (Information systems Department, JAL)

** 大阪大学基礎工学部情報工学科

*** 日本航空(株)情報システム部

**** 本システムの開発に用いられた計算機システムは，24 k 語の主記憶，1.2 M 語のディスクパック装置，約 100 k 語の磁気テープ装置を備えている．

2. 連想三つ組について

AS 11 の連想三つ組および連想対を構成する単位を項目 (item) と呼ぶ．項目は 32 ビット* (2語) のビット系列の外部識別子 (external identifier) として指定される．項目は，それを成分とする連想三つ組および連想対を登録する前に，予め登録しておかなければならない．

連想三つ組 (以下単に三つ組と呼ぶ) は，三つの項目の順序集合であって，その各成分を順にそれぞれ A (attribute) 成分，O (object) 成分および V (value) 成分と呼ぶ．いま a, o, v で三つの項目を表わしたとするとき，それらを A, O, V 成分とする三つ組は ⟨a, o, v⟩ で表わす．三つ組を用いることによって，たとえば，二項関係 "BILL の SON は TOM である" は，⟨SON, BILL, TOM⟩ と表現できる．ここで，"SON", "BILL", "TOM" はそれぞれ項目である．

連想対(単に対と呼ぶ)は，項目の順序対であって，第一成分を H (head) 成分，第二成分を T (tail) 成分と呼び，h, t で二つの項目を表わしたとするとき，それらを H, T 成分とする対は ⟨h, t⟩ で表わす．

さらに，三つ組や対自身に項目を割り当てて，これを他の三つ組や対の成分とすることもでき，AS 11 ではそれぞれを三つ組識別子および対識別子と呼ぶ**．i を三つ組 ⟨a, o, v⟩ の三つ組識別子とすると，⟨a, o, v⟩=i で表わされる．これを用いて ⟨h, t, i⟩ とすれば，これは五つ組 ⟨h, t, a, o, v⟩ を表わしているとみなすことができる．

ユーザは項目に対して，三つ組および対の検索の対象とならないような余分な情報を割り当てることができる．その情報をその項目に関するデータと呼ぶ．ユーザの与えたデータは，AS 11 で扱うファイルに登録されるが，その項目に関連してユーザにより参照されるだけで，三つ組および対の検索にはなんら影響を与えることができない．AS 11 はデータの形式や内容には関与せず，ユーザ自身が管理しなければならない．前例で，BILL や TOM の年令や身長などを項目 "BILL", "TOM" のデータとして登録することができる．

ユーザが登録したひとまとまりの項目，そのデータ，三つ組，および対は一つのファイルとして扱われ，これを連想記憶ファイル (AF) と呼ぶ．

三つ組 (対) に関する検索の種類としては **Table 1** の 8(4) 通りがある．

Table 1 Retrieval requests

F 0	⟨a, o, v⟩
F 1	⟨a, o, −⟩
F 2	⟨a, −, v⟩
F 3	⟨a, −, −⟩
F 4	⟨−, o, v⟩
F 5	⟨−, o, −⟩
F 6	⟨−, −, v⟩
F 7	⟨−, −, −⟩

G 0	⟨h, t⟩
G 1	⟨h, −⟩
G 2	⟨−, t⟩
G 3	⟨−, −⟩

(注) たとえば，⟨a, o, −⟩ は，同一の a, o を持つ三つ組をすべて求めよという検索要求．

3. AS 11 のデータ構造

AS 11 では，ディスクと主記憶間のスワッピング，ディスク上のファイルの管理はすべて PDP 11 のモニタに委せている．

AF には，ディスク上の物理的に連続した領域が割り当てられる．そして，それはファイルヘッダ部と表部とに大別される．

ファイルヘッダ部には，ファイル名，ファイル内の各表を管理するため，それぞれの表の大きさやハッシュのためのパラメータなどが置かれている．表部は，項目を格納する表 (IT)，三つ組，対を格納する表 (それぞれ TM, PM)，TM や PM のあふれを格納する表 (OFT)，項目に割り当てられたデータを格納する表 (DT) から成っている．それぞれの表の詳細については後述する．

IT は，三つ組や対に関するリクエスト (ユーザが AS 11 に出すサービスの要求をリクエストと呼ぶ) にも必ず使用され，またその使用頻度が高いので，その AF がオープンされている間，主記憶上に常駐している．IT 以外の表は，すべて一定語数の大きさにセグメント化されていて，要求時にセグメント単位で読み込まれる．各セグメントは，セグメントヘッダと，2語または4語のセル (後述) に細分されている (DT は例外)．セグメントヘッダには，そのセグメントの識別名，フリーリストヘッダ，統計用領域などこのセグメントを管理するために必要な情報が置かれている．

セグメントの大きさが小さすぎるとあふれが生じる．また，セグメントの大きさが大きくなると，主記憶上のスワップ領域に同時に入ることのできるセグメ

---

* 適当な変換で英数字6文字の系列を項目のニーモニックとして用いることができる．

** これは LEAP で "bracketed triple" と呼ばれているものである．

ント数が少なくなり，スワップ回数が多くなる．そのために，ユーザが自分自身で効率を考慮して登録する三つ組や対の性質に適したAFの大きさを自由に指定するようになっている．すなわち，ユーザはセグメントの大きさをディスクの転送単位である256語の整数倍で指定し，さらに各表に対して必要なセグメントの枚数（0でも可）を指定する．ただし，0以外のときにはそのセグメントの枚数に関して，後述の条件（式1）を満足させなければならない．

以下に，それぞれの表の構造を示す．

3.1 IT (Item Table)

項目を格納する表で，この表は **Fig. 1** に示されるようなセルと呼ばれる4語ずつに区切られており，登録されている項目に対して一つのセルを割り当てる．このセルのIT内での相対位置（4語単位）は，外部識別子をハッシュすることによって決められる．この相対位置を13ビットで表現したものを内部識別子(internal identifier)と呼び，三つ組や対の取り扱いの際に使用する．

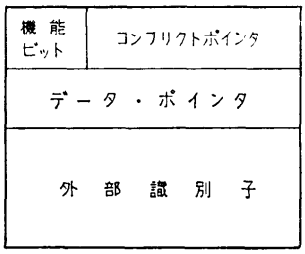

Fig. 1 Structure of a cell in IT

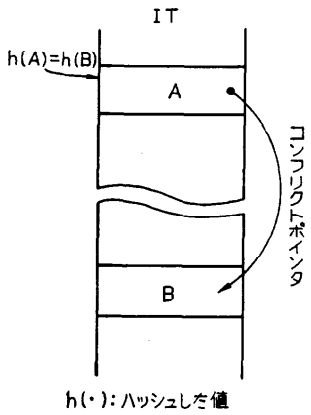

Fig. 2 Critical example on removing a cell from IT.

項目Bをハッシュした値が，すでに登録されている項目Aの値と同じになったとき（これをコンフリクト(conflict)またはコリジョン(collison)と呼ぶ．）一つのフリーセルを求めてきてBを格納し，Aからコンフリクトポインタ(conflict pointer)と呼ばれるポインタでリング状につないでいる（**Fig. 2**）．このような状態で，AをITから除去しようとするとき，IT内での相対位置そのものを項目の内部識別子として使用しているので，BをAの占めていたセルに移すことはできない．また，Aのセルを単なるフリーセルとして扱うことができず，Bへのコンフリクトポインタを残しておかなければならない．このような状態を区別するために，機能ビット（3ビット）により，セルの種類を区別している．

3.2 DT (Data Table)

データを格納する表で，可変長のデータが扱える．DTの各セグメントはBuddy System* で管理されている．**Fig. 3** に，一つの項目に関するデータを格納するセルの構造を示す．セルの最初の1語（データが長いときは2語）は，このセルがどのような使用状況にあるかを表示している．この項目が三つ組（対）識別子である場合には，AS 11が自動的に最後から3(2)語にその三つ組（対）の成分を記録している．

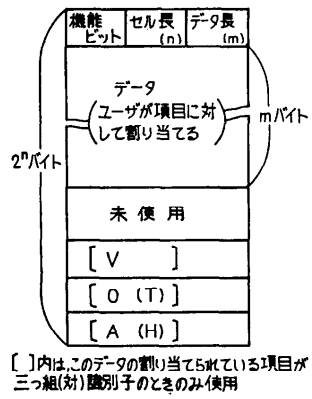

Fig. 3 Structure of a cell in DT.

3.3 TM (Triple Map)

三つ組を格納する，AT (Attribute Table), OT (Object Table), VT (Value Table) の3枚の表をあわせてTMと呼ぶ．ATは，Table 1 における F1, F3, OT は F4, F5, VT は F2, F6 の検索に使用される．このため，同一の三つ組を，この三つの表にサイクリックに格納している．（したがって，一般にこの

* 記憶領域を動的に割り当てる一手法．要求されたスペースに対し，その大きさを越える最小の2の巾数の大きさをもつ記憶領域を割り当てる[4]．

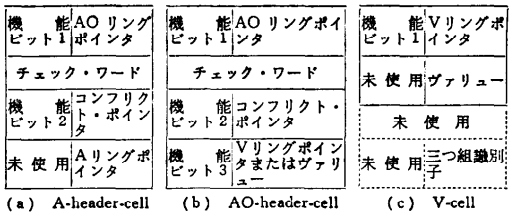

Fig. 4 Structure of a cell in TM.

三つの表は冗長である.) F0, F7 の検索は，どの表からでも検索できる．

AT を例にとって各表の構成を説明する．Aヘッダセル (AHC, **Fig. 4**) はA成分を置くためのセルで，F7 の検索を能率よく行えるためにAリング (開始セルはセグメントヘッダ内にある) で結ばれている．AO ヘッダセル (AOHC, Fig. 4) はO成分を置くためのセルで，F3 の検索を行うためにAO リング (開始セルは AHC) で結ばれている．また，V セル (Fig. 4) は V 成分 (および，三つ組識別子を持てばそれも) を置くためのセルで，F1 の検索を行うためにV リング (開始セルは AOHC) で結ばれている．ただし，同一の A, O 成分を持つ三つ組が複数個存在しなくて，かつその三つ組が三つ組識別子を持たない場合に限り V 成分は AOHC 内に置かれる．

各セルの識別は，機能ビット (3 ビット) により行い，フリーセルの管理は2語セルと4語セルとを別々に二方向リンク付きリスト (開始セルはセグメントヘッダ内にある) により行っている．

たとえば，**Fig. 5** (a) は AT に三つ組 $\langle A_1, O_1, V_1\rangle$, $\langle A_1, O_1, V_2\rangle = I_1$, $\langle A_2, O_2, V_2\rangle$, $\langle A_2, O_3, V_3\rangle = I_2$ が登録されている例である．このような状態で F3 の検索要求 $\langle A_1, -, -\rangle$ を処理する方法は，項目 A_1 をハッシュ (後述) して A_1 の AHC を求め，AO リングを辿り O_1 を求め，V リングを辿り V_1, V_2 を求める．また F1 の検索要求 $\langle A, O, -\rangle$ に対しては，

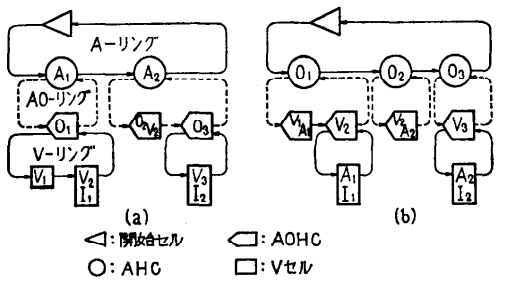

Fig. 5 Outline picture of AT (a) and OT (b)

A と O の二つでハッシュして直接 AOHC を求めて行う．

また，OT (VT) でも同様に，AHC に対応するセルには O(V) 成分，AOHC に対応するセルには V(A) 成分，V セルに対応するセルには A(O) 成分が格納される．この場合の OT を Fig. 5 (b) に示す．

3.4 PM (Pair Map)

対を格納する表で，HT (Head Table) と TT (Tail Table) の2枚の表から成っている．TM と同様に，HT, TT の機能はサイクリックおよび冗長であり，HT は G1, TT は G2 の検索の処理に便利なように作られている．なお，PM では，AOHC と同じ構造の H ヘッダセル (HHC), V セルと同じ構造の T セルがある．各セルの機能は，対 $\langle h, t\rangle$ をすべて同一のA成分 a を持つ三つ組 $\langle a, h, t\rangle$ と考えたときとほぼ同一である．

ところで HT においてT リング上の複数個のTセルは，通常そのT成分の内部識別子の値の小さなものから順につながれている．これにより検索の時間効率が上がる．AS 11 では，このTリング上でのTセルの並べ方をユーザの指定通りにすることも可能である．すなわち，同一のH成分を持つ対の間に全順序関係を導入することができる．LEAP などでは答えのやりとりはすべて集合レベルで行われているのでこのような順序関係を持たせることができない．そのためには対を階層的に構成したり，項目のデータ (LEAP では datum) の中で順序関係を表現することが必要であり，ユーザ自身が順序関係を扱わなければならない面倒がある．しかし，AS 11 では順序関係による検索もリクエスト段階で可能である．なお，同様のことが TT, AT, OT および VT においても可能である．

また，前述したように，AS 11 は各検索要求を効率よく処理するために，対を2枚の冗長な表に登録する．ところで，たとえばアセンブラのシンボルテーブルの作成を AS 11 で行うとして，シンボル t がシンボルテーブルに登録されていることを対 $\langle$'symbol', t$\rangle$ で表わすものとした場合，$\langle -, t\rangle$ のような検索要求を出すことはないであろう．このような場合に，対 $\langle$'symbol', t$\rangle$ を TT に登録するのは時間およびディスク上のファイルの大きさの点から非能率的である．AS 11 では，特定の表 (この場合は TT) に故意に登録しないこと (非対称登録) を可能としている．ただし，この場合は HT と TT の内容は一致しない．それゆえ，G0, G3 の検索要求では，どの表から検索す

るのかをリクエストで指定することになっている．また，HT, TT の大きさを互いに独立に指定（あるいは全く設けないように指定）することができる．

この非対称登録と前述の順序関係の機能は，AS 11 の大きな特徴である．

3.5 OFT (Over Flow Table)

三つ組や対のあふれを収容する表で，セグメント番号0と1との2枚のセグメントから成っている．

PM, TM においてセル間の関係を示すために用いているポインタは，すべて 13 ビットである．上2ビットでセルの置かれているセグメントを示す．すなわち，00 であればそれが本来置かれるべきセグメントを，10 なら OFT の番号0のセグメントを，11 なら同じく番号1のセグメントを示している．下 11 ビットは，指定されたセグメント内におけるセルの相対位置（2語単位）を示している．

3.6 ハッシュ

AT のセグメント内には，前述のように AHC, AOHC, V セルの3種のセルがある．三つ組を格納する場合，AHC と AOHC の置かれるべき位置は，それらに格納される成分をハッシュすることによって決められ，V セルは単にフリーセルの一つに割り当てられる．ここでは，AT を例にとって TM における AHC と AOHC のハッシュの方法について述べる．PM における格納方法もほぼ同じである．

3.6.1 AHC のハッシュ

A をハッシュされる内部識別子とし，AT のセグメント数を N_S，1セグメント内に入る4語セルの数を N_c とすると，AHC は A(mod N_S)* 番目のセグメントの相対位置 A(mod N_c) のセルに置くことにした．この位置にすでに他の AHC または AOHC が置かれている場合(すなわちコンフリクトが生じている)には，フリーセルの一つを AHC とし，元のセルからコンフリクトポインタによりリング状につないでいる．コンフリクトポインタによりつながれているセルのことをコンフリクトセルと呼ぶ．また，この位置にすでにVセルまたはコンフリクトセルが置かれている場合には，これらのセルをフリーセルを使用して別の位置に移動させ（そこの相対位置に置くべき必要性がないので移動させてもかまわない），この位置に AHC を置く．チェックワード（このセルが求める AHC であるかを確認するために使用する語）としては，Aそのものを使用する．

3.6.2 AO ヘッダセルのハッシュ

内部識別子AとOをハッシュしたときのチェックワードとしては，最高 16 ビットまでしか許されない．ハッシュで得られたセルが求める A, O に関する AOHC であるか否かを確認するのに利用できる情報は，チェックワードとそのセルが置かれているセグメントの番号およびセグメント内の相対位置である．したがって，使用できる情報は最大 $\log_2(N_S \times N_c)+16$ (ビット)である．これから，A, O 二つの内部識別子が識別できるためには，N_I を IT に登録可能な項目数，すなわち内部識別子の最大数とすると，内部識別子の有効桁は $\log_2 N_I$（ビット）であるから，条件

$$2\log_2 N_I \leqq \log_2(N_S \times N_c)+16 \qquad (1)$$

を満たさなければならない．

AS 11 では各処理を高速に行うために，ハッシュの際に使用される成分が同じである三つ組や対はすべて同一のセグメントに格納される．つまり，AOHC がどのセグメントに置かれるかはAのみで決められ，ふつう AHC と同じセグメントに置かれる．セグメント内の AOHC の位置とそのチェックワードは，IT と AT の大きさの関係にしたがって，つぎの二通りの方法のうちのいずれかが用いられる．

(i) $\log_2 N_I \leqq \log_2(N_S \times N_c)$ の場合

AOHC のセグメント内の相対位置をうまく選ぶことによって，A をセグメントの番号とセグメント内の相対位置だけで識別できるので，A の内部識別子 $\log_2 N_I$（ビット）だけが，セグメント番号およびセグメント内の相対位置を決定するために用いられる．また，O の内部識別子 $\log_2 N_I$（ビット）だけが AOHC のチェックワードとして用いられる**.

(ii) $\log_2 N_I > \log_2(N_S \times N_c)$ の場合

Aをセグメントの番号とセグメント内の相対位置だけで識別できないので，Aの“表わし切れない情報”は，チェックワードの一部を用いて表わす．Aの内部識別子のうち $\log_2(N_S \times N_c)$（ビット）だけがセグメント番号およびセグメント内の相対位置を決定するために用いられる．チェックワードとしては，Oの内部識別子 $\log_2 N_I$（ビット）と A の $\log_2 N_I - \log_2(N_S \times N_c)$（ビット）が用いられる．

---

* X(mod Y) は，整数 X を正整数 Y で割ったときの剰余を表わす．

** A の内部識別子のみで AOHC の位置を決めると，同じ A に関する AOHC はすべてコンフリクトを起こすので，実際には A の内部識別子とOの内部識別子の一部を合わせて位置を決めている．

4. プログラムと AF の管理

Fig. 6 は AS 11 のプログラムと AF の構成図であり，**Table 2** は現在の AS 11 のリクエストの概要を示している．

システム領域｛核領域：常駐ルーチンの置かれる領域
　　PSA：オーバレイモジュールがスワップされる領域
表領域：ユーザ・プログラムが AS 11 をコールしたときにユーザ領域内に位置と大きさを指定する．
IT は AF がオープンされている間，表領域の一部に常駐，AF の IT 以外の表は，表領域の残りの部分でスワップされる．

Fig. 6 Organization of AS-11's program and AF.

Table 2 AS-11's requests

・ASINT	AS 11 の使用をモニタに要求する．
・ASRLS	AS 11 の使用終了をモニタに告げる．
・AFALOC	AF を初期化し，オープンする．
・AFOPEN	AF をオープンする．
・AFREAD	AF を検索専用でオープンする．
・AFCLOS	AF をクローズする．
・MAKEI	項目を IT (およびデータがあればそれを DT) へ登録する．
・UPDTI	項目のデータ部分の変更・削除を行う．
・GETIT	項目に関する情報を取り出す．
・DELTI	項目を削除する．
・ORDER	対や三つ組を順序集合として登録することを宣言する
・MAKEA	対や三つ組を PM・TM に登録する．
・ATCHI	対や三つ組に項目を付ける．
・DETCI	対や三つ組に付けられた項目をはずす．
・GETID	項目に付いている対・三つ組を得る．
・GETOR	順序集合の n 番目を得る．
is-requests	G 0, F 0 に対するリクエスト群．
all-requests	G 3, F 7 に対するリクエスト群．
ret-requests	G 1, G 2, F 1～F 6 に対するリクエスト群．
・NEXT	未完了の ret-requests や all-requests の続行．
ers-requests	対や三つ組を削除するリクエスト群．
・ERASE	対や三つ組を削除する．
・STAT 1	AF の統計を得る．
・STAT 2	AF の統計を更新する．
・PUSH	DT 使用のスタックへ置く．
・POP	DT 使用のスタックから取る．
・PEEP	DT 使用のスタックの n 番目を見る．

* AS 11 の初期設定の処理のための SVC を設けるために，モニタには多少の変更がなされている．

** PDP 11 の DOS は常駐部とオーバレイ部に分かれており，オーバレイされるルーチンが必要なサービスをユーザからリクエストされると，DOS は自らの領域を拡張し，そこへ必要なルーチンを読み込む．AS 11 では AS 11 全体をこの一つのルーチンとみなしている．

4.1 プログラムの管理

AS 11 のプログラムは，ふつうディスク上に置かれ，核部とオーバレイ部とから成っている．ユーザが AS 11 の使用開始を要求するスーパバイザコール* (SVC) を出すと，モニタはモニタ領域を拡張する** ことによって，AS 11 が入るべきコア領域を用意する．つぎに，AS 11 の核部を，このコア領域の一部に読み込み，システムの初期設定を行う．

核部は常駐で，システム用の表と使用頻度の高い種々のルーチンから成っている．核部以外のルーチン (オーバレイルーチン) は要求時に主記憶に読み込まれる．ユーザのリクエストを処理するルーチンは，ほとんどこのオーバレイルーチンである．

オーバレイルーチンが読み込まれる主記憶の領域は，PSA (Program Swap Area) と呼ばれ，AS 11 用領域の核部以外の部分が割り当てられる．オーバレイルーチンは PSA のどこに置かれるかが不定であって，そのためにオーバレイルーチンは，いわゆる PIC (Position Independent Code) で書かれている．オーバレイルーチンはいくつか集まってある決まった語数 (512 語) のオーバレイモジュール (以下モジュールと略) を構成し，これがスワップの単位となる．512 語を越えるルーチンは，分割して別々のモジュールに配分されている．現在 22 個のモジュールがあり，ルーチン表と呼ばれる表により管理されている．ルーチン表は核部にあり，各ルーチンが現在 PSA にあるか否かを示している．

各ルーチンの途中結果はディスクには戻されない．そのため，あるルーチンが一度スワップアウトされると，そのルーチンがそれまで行っていた仕事に関する情報をすべて失う．このために，PSA の管理は原則として FIFO で行われるが，これに加えて使用計数法と NXTCNT (next count) による処理途中のモジュールのスワップアウトの防止が行われている．PSA はフレームと呼ばれる 512 語単位の区間に分割されている．使用計数法は，各フレームに USECNT (use count) という 1 バイトの領域を設け，そのフレーム上のモジュール内のルーチンに入るごとにこの USECNT を一つ増し，出るごとに一つ減らしていく方法である．これによって，新しいモジュールを読み込む際に，使用中のフレームに新しいモジュールを読み込まないようになっている．また，AS 11 では検索を行う際に，取り出す対象となる集合が大きい場合に，その集合を指定した数ずつの部分に分割して取り出せる

ようになっている．このような場合，検索中のモジュールを PSA に残しておかなければならない．そのために，NXTCNT という1バイトの領域も設けている．NXTCNT は，集合を返す場合にユーザの指定した数だけ取り出した後，まだ返すべき結果が残っている場合に 127 にセットされる．

新しいモジュールを読み込むべきフレームを探すアルゴリズムを述べる．FIFO ポインタ（つぎに使用されるべきフレームを示している）の示す USECNT を調べ，もし0なら NXTCNT から1を引き，結果が負ならばこのフレームを空きフレームとみなし，新しいモジュールを読み込み，USECNT と NXTCNT を0にセットして FIFO ポインタをつぎのフレームへ進める．結果が非負か USECNT が 0 でなければ FIFO ポインタをつぎのフレームに進める．この過程を空きフレームが見つかるまで繰り返す．

NXTCNT の使用により，検索途中のモジュールは 127 回までは PSA に残される．これにより，ユーザが検索を途中でやめたとしても，最後には NXTCNT は0となってそのフレームを再び有効に使用できる．

4.2 AF の管理

IT はユーザが指定した表領域の一部に常駐させられ，その他の表のセグメントは残りの表領域である DSA (Data Swap Area) と呼ばれる領域でスワップイン/アウトされる．複数個の AF を同時にオープンして使用する場合，表領域はこれらのファイルによって分割して使われる．

AF のセグメント管理は FSMT (File Segment Management Table) と DSAT (Data Swap Area Table) によってファイルごとに行われる．FSMT はそれぞれ1語の状態語から成り，これはそのファイル中の各セグメントに一対一に対応していて，そのセグメントが DSA にある場合にはその絶対番地/2が，ない場合には0が書かれる．上1ビットはライトフラグ (write flag) であり，このセグメントが DSA で更新されたときに1がセットされる．DSAT は DSA のスワップ領域の枚数（各々をメモリセグメントと呼ぶ）だけ DSA ワードがある．その第1語にはこのメモリセグメントを占めているセグメント名が置かれ，第2語は DSA 管理のためのスワップの順番を示す二方向リングのためのポインタとなっている．

これらの表によって，セグメント管理はつぎのような LRU (Least Recently Used) 方式で行われている．あるセグメントにアクセスする場合，そのセグメントに対応する状態語を調べ，0 (DSA にない) ならディスクから読み込む必要がある．そのときは，DSAT のリングの最先端にあるメモリセグメント上のセグメントをディスク上に返し（ただしライトフラグが0ならばその必要はない）必要なセグメントをここに読み込み，これらの状態語を更新する．さらに，この DSA ワードのリングポインタを修正し，今読み込まれたメモリセグメントに対応する DSA ワードをリングの最後尾に置く．また，状態語が0でない (DSA にある) 場合には，このメモリセグメントが最後尾になるように DSA ワードのリングポインタを修正するだけである．これにより，処理中のセグメントがスワップアウトされるのを防いでいる．このように，LRU を実現するためには FIFO にスワップの順を示す二方向リングのポインタの修正機能を付け加えるだけでよい．リングを並べ換えるアルゴリズムはいくつか考えられるが，メモリセグメントの数はそれほど多くないと考えられるので，二方向リングを用いて時間効率を高めている．

5. あ と が き

我々は記憶領域の効率的な使用という考え方をもとに，大量のデータを扱うデータ構造処理システム AS 11 を設計，製作した．主記憶の節約のためには，セルを小さくまとめたり，AF やシステム自身のプログラムをオーバレイすることにした．また，実験により，平均データ長4語の項目 127 個を登録し，さらにそれらを成分とする 45 個の対，33 個の三つ組を対称登録するのに 30 秒間かかるという結果を得た．

AS 11 を使って再帰的なプログラムを作成できるように，DT の機能を利用した可変長のスタックリクエスト（これは PDP 11 のスタック機能とは異なる）と，集合演算に関するリクエストがすでに作られており，AS 11 の機能を充実させている．

また，AS 11 はアセンブリ言語による基本的なリクエストを処理するシステムであり，より使い易くするためには高級言語の開発が考えられるが，第一段階として我々は FORTRAN への組み込みを現在行っている．

謝辞．日頃御指導いただく嵩忠雄教授，御討論いただいた電電公社の斉藤孝文氏および嵩研究室諸氏に深謝いたします．

参考文献

1) C. A. Lang & J. C. Gray: ASP-A Ring Implemented Associative Structure Package, Comm. ACM, Vol. 11, No. 8, pp. 550～555 (1968).
2) J. A. Feldman & P. D. Rovner: An algol-based associative language, Comm. ACM, Vol. 12, No. 8, pp. 439～449 (1969).
3) 古川康一，山崎成美：汎用データ構造処理システム EDSP について，電子技術総合研究所彙報，Vol. 37, No. 1, 2, pp. 90～100 (1972).
4) D. E. Knuth: The art of computer programming Vol. 1/Fundamental Algorithm, Addison Wesley pp. 442～445 (1968).
5) 萩原，荒木，本田，細見，嵩：連想機能をもつ汎用データ処理システム AS 11 について，昭49信学会全大論文集，講演番号 1457 (1974).
6) 荒木，萩原，本田，細見，嵩：連想処理システム AS 11 のデータ構造について，昭49信学会全大論文集，講演番号 S 10-7 (1974).
7) 古川康一：データ構造（I），情報処理，Vol. 13, No. 5, pp. 302～310 (1972).

(昭和50年4月7日受付)
(昭和50年5月21日再受付)