計算機処理用ファイル概説

関本年彦

近時、端末装置からの計算機利用という、利用形態の変革とうらはらに、計算機内の大形ファイルの構築と、その処理法の高度化、という現象が著しい。この趨勢は、今後ますます発展し、過去における計算機の量的普及以上の社会的影響をもたらすことが予想される。

本稿では、計算機によるファイル処理の発展の跡をたどりながら、近年開発された、検索機能を有するファイル処理用汎用ソフトウェアの仮想記憶装置処理法——Virtual Memory Access Method（通称 VSAM（ヴィサム））——に焦点をあてて現況を考察する。

本論に入る前、しばらくを予備的考察にあてる。

図一(イ)は、入学試験の際の志願者に関するデータを、機械処理向にまとめてみた一つの形式である。これをレコードといい、レコードを、たとえば志願者の数だけまとめたものがファイルである。計算機は、ファイル内のレコードが、いずれも、たとえば上述のような一定の形式をもっているファイル、すなわち、formatted record から成るファイルに限って処理することができ、現在では、小説の文章の取扱い等に関しては、まだ非力であ

図一

(イ) 志願者レコード，37文字から成っている。受験番号中最初の２ケタは学部・学科コード

受験番号	氏名	生年月日	高校コード
310105	アオキ　イチロオ	371015	13255
1 6	7 26	27 32	33 37

(ロ) ある科目の採点レコード

受験番号		得点		
		(1)	(2)	(3)
310105	1	33	20	22
1 6	7 8			13

↓科目コード

(ハ) (イ)の志願者レコードに(ロ)のある科目の得点が記入されたレコード。この後さらに何科目かの得点と総得点が記入される。

受験番号	氏名	生年月日	高校コード	得点
310105	アオキ　イチロオ	371015	13255	075
1 6	7 26	27 32	33 37	38 40

る。

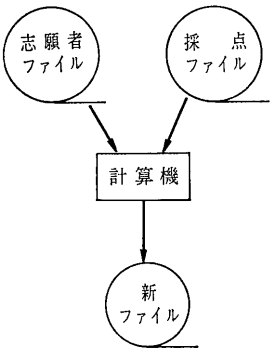
図一(ロ)は、入学試験実施後のある科目の採点レコード形式を示す。磁気テープ上に記録された志願者ファイルの各レコードに、採点結果を記入するには、採点ファイルも磁気テープ上にあるとして、まず、両ファイルのレコードを受験番号順に整列させ、その上で、計算機に両ファイルから一レコードずつ送り込む。計算機は、両ファイルからのレコードの受験番号を照合し、志願者レコードに採点を付け加えたレコード（**図一**(ハ)）を作って、第三のテープ上に出力する（**図二**参照）。ここでは、最初に、二つの入力ファイルについて、レコードを整列する操作を行っているが、この操作を分類—sort—という。また、ここでは受験番号に関する分類を行っているが、この場

合、受験番号を分類の制御項目—control field—、あるいは、見出語—index—という。

このような、分類の済んだファイルについて、その全レコードを一度に処理する方法を、逐次処理法—Sequential Access Method—という。

ところで、第二次大戦末期に科学技術計算用の機械として出現した計算機は、一九六〇年代において、その初頭にファイル媒体用磁気テープ装置の実用化が成功するにおよび、事務処理用の機械として、一般企業を中心に急速な普及を見た。計算機がある分野に普及するためには、その分野で用いられる汎用ソフトウェアの開発が基本的要件となる。たとえば、フォートランの完成は、科学技術用計算機が普及する原動力となった。計算機が、事務処理用機械として普及するのに貢献した汎用ソフトウェアとしては、オペレーティングシステム、およびコボル、PL／Iなどのプログラミング言語があり、また、磁気テープファイル処理用として発達したIOCS(Input/Output Control System)、すなわち、後の逐次処理法—Sequential Access Method—も看過できない重要なものである。

図二　志願者ファイルへ採点結果の記入



第三節に述べるISAM（アイザム）は、磁気ディスク装置の実用化とともに開発された最初の、検索機能をもつファイル処理用汎用ソフトウェアであったが、最近の高度なファイル処理に耐えられず、このため新しく開発されたのが、第四節で述べるVSAMである。

一　カードファイルからテープファイルへ

前述のように、計算機が事務処理機械として利用され始めたのは、磁気テープ装置が実用化された。一九六〇年代初期である。それまでの機械化された事務処理においては、カードがファイル媒体であり、カードファイル処理用の機器群は Punch Card System (PCS) と称され、日本においても、昭和初期からＩＢＭ社製のものが、銀行・保険会社などで用いられており、一九六〇年前後には、ＩＢＭあるいはレミントン社製のものが、企業間に広く普及していた。なお、このカードそのものの歴史はかなり古く、前世紀末、米合衆国の国勢調査集計用に考案されたもので、いまでも、その考案者に因んで、Hollerith カードと呼ばれることがある。

カードファイルは、機械処理用として考えるかぎり、逐次処理専用ファイルであり、ＰＣＳ中にも、分類用機器 sorter がある。計算機によるテープファイル処理法は、基本的にはＰＣＳを通して開発された逐次ファイル処理技術をそのまま応用したものと見なせるが、前者がもたらした処理速度の向上は画期的であり、さらに、計算機本体（中央処理装置）に接続された少数種類の機器が、ＰＣＳの各種機器が独立に行っていた各種プロセスを、すべて自動的に処理してしまうので、省力化を含む低コストでの大量事務処理が可能となり、計算機の普及を見たのである。実際、両者の処理速度を比べてみると、ＰＣＳのカード処理速度は、当時、毎分五〇〇カードが限界であったのに対し、計算機によるテープファイル処理速度は、カード換算すれば、毎分三万～六万カードに当り、二ケタの差があった。また、逐次処理においては必須のプロセスである分類については、上述ＰＣＳの sorter が、その機構上一度に見出語の一文字に関する分類しかできず、五文字から成る見出語については、五度の分類が必要となるのであるが、その速度は、最も高性能な機器で毎分一〇〇〇カード、通常機器では毎分五〇〇カードであったのに対し、テープファイルの分類では、見出語の文字数にはほとんど無関係に、カード換算で

毎分五千～一万カードが可能で、しかも、後者においては、カード移送等の人力操作は一切不要で、いわばワンタッチ操作が可能となった。

計算機は、電子計算機とも称されるように、その内部における計算、あるいは情報処理を電子的に行うのであるから、いわば、光の速度の水準でそれを行っている。一方、ファイル媒体の駆動は、電動モータにより機械的に行うのであるから、当然、両者にスピードギャップが生ずる。それゆえ、計算機システム設計において、つねに基本的課題となることの一つは、このスピードギャップを埋める方法、いいかえれば、計算機のもつ高速性の活用技術を創案することである。計算機の出現以来、この目標に沿って、数多くの技術が考案され、実用化されてきているが、その一つに、ファイル媒体へのレコードブロッキングによる記録法というのがある。これを、磁気テープについて見ると、データを記録する際、テープの駆動を開始し、定速度が得られてから実際の記録を始めるのであるが、駆動開始後定速度到達まで数ミリ秒を要する。この所要時間は、計算機本体における数千回の演算時間に匹敵し、また、磁気テープの記録速度にしても、定速度到達後においては、カード換算で数十枚分のデータを記録できる時間であるから、無視できないものである。そこで、このロスタイムを実質的に短縮するため、通常一度に数十から数百のレコードをまとめて書込む。磁気テープ装置側としては、これら一度に書込まれるレコードを、一個のレコードと見なすので、これをとくに、物理レコードと言い、実際の事務処理で扱うレコードを論理レコードと言って区別することがある。一方、テープからのデータ読出しに際しては、物理レコード単位で行われるが、ユーザプログラムの読込命令に対しては、逐次処理用汎用プログラムが、計算機内部記憶装置にあらかじめ読込んでおいた物理レコードから一論理レコードだけ切り離して提供する。後述する磁気ディスク装

置の場合も同様の事情があり、一般に、ファイル媒体上でいくつかの論理レコードをまとめて物理レコードとして扱うことを、ブロッキングといい、一物理レコード中の論理レコードの数を、ブロッキングファクタという。

磁気テープ一巻の通常の長さは二四〇〇フィート、また、現今もっとも多く使用されている磁気テープ装置では、記録密度が一インチ当り一六〇〇文字であり、この場合、テープ一巻には約一〇万カードに相当するデータを記録できる。なお、テープ上の各物理レコード間には Inter-Block Gap（IBG）と呼ばれる〇・六インチ巾のギャップが生じるので、テープ一巻当りのデータ容量はブロッキングファクタにより左右される。もちろん、ブロッキングファクタが大きい程、容量も大きくなる。テープ一巻に一〇万レコードが記録されているとして、この処理時間は、計算機本体内の処理時間を一応無視して考えると、全レコードを読出す時間ということになり、前述の毎分五万レコードの処理速度を仮定すると、二分である。これは逐次処理を前提としたものであるが、もしも、各レコードごとに検索を要する個別レコード処理を行うならば、検索時間に平均一分を要することになり、したがって、テープファイルを、検索が必要となる個別レコード処理に利用することは、実際上不可能である。

二　磁気ディスク装置

磁気ディスク装置は、磁気テープ装置とはその機構が根本的に異り、検索可能ファイル用媒体として要求される諸特性を備えた装置である。一九六〇年代の半ばにおいては、性能、価格、安定性において実用性を有するものが現れ始めたが、ファイル媒体として利用するのには記憶容量の面で十分でなかったため、一九六〇年代にお

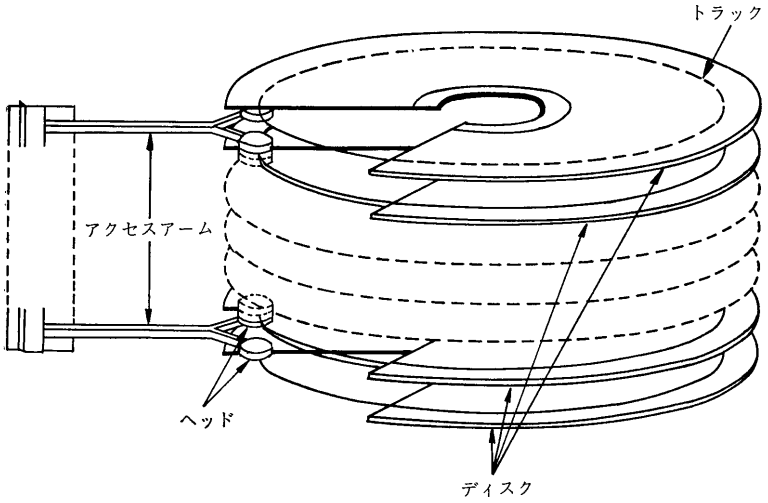
いては、主としてシステムプログラムの収納等に利用され、オペレーティングシステムの高性能化への貢献が顕著であった。因みに、当時一パック当りの容量は、一千万文字前後が普通であり、一九六〇年代末に現われた高性能磁気ディスク装置IBM二三一四で、漸く、一パック当り、三千万文字の記憶容量を有した。一九七〇年代に入り、一パック当り一億文字（カード換算で、約一〇〇万カード）の記憶容量を有する磁気ディスク装置が出現し（たとえばIBM三三三〇）、同時に、速度、価格、安定性等の面でも向上が著しく、ファイル媒体としての利用が緒についた。

磁気ディスク装置においては、データはディスク上に磁気的に記録されるのであるが、ディスクとは、酸化鉄粉の薄層を付着したアルミ合金製の円盤で、これが数枚、回転駆動軸に固定されている。これら円盤のセットは、磁気ディスク装置のタイプにより、交換可能なものと、固定のものとがあり、交換可能なものは、とくに、ディスクパックあるいはディスクモジュールと呼ばれる。データは、各ディスクの両面に同心円状に記録され、これらの各同心円をトラックという。

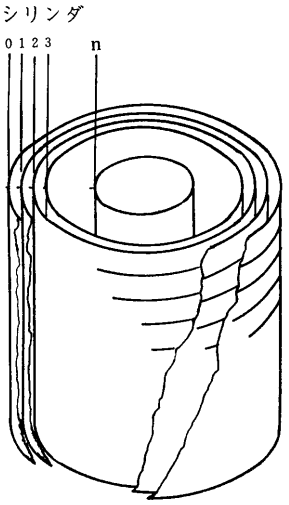
ディスクに対するデータの書込み、読取りは、ヘッドにより行われるが、ヘッドは各ディスクの片面に一個ずつあり、櫛状に並ぶアクセスアームの先端に取付けられている。櫛状のアクセスアーム部は、全体が同時にディスクの半径方向に動作し、ディスクパック中の数千あるトラックの一つが選択されると、ヘッドがそのトラック上に来るようアーム部が作動する。これを、ヘッドポジショニングという。ヘッドポジショニング後、そのトラックがあるディスク面に対応するヘッドが選択され、書込み、または、読取りが行われる。

ヘッドポジショニングに要する時間は、現状の機種で平均二五ミリ秒前後であるが、実際に書込み、または読

図三(イ)　ディスク



図三(ロ)　シリンダ概念図



取りが始まるのは、ヘッドポジショニング後さらに選択したトラック上の所要の物理レコードの先端が、ヘッド直下に到達した時点であり、この間の所要時間は平均一〇ミリ秒前後（これは½回転時間である）である。したがって、ハードウェアの見地からすると、ディスク装置上のレコードは、いずれも、平均約三五ミリ秒で取り出せることになる。この所要時間を待時間（アクセスタイム）という。

以上の話から、ディスク上のファイル編成を考える際、ヘッドポジショニング時間をなるべく少なくするため、一度のヘッドポジショニングで読み書きが可能

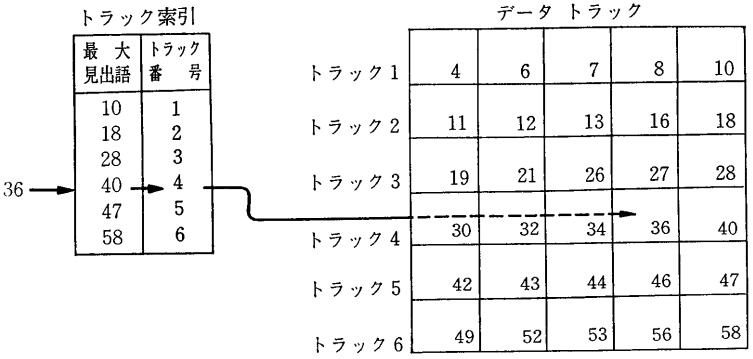
な諸トラック（図三では一二トラックある）を、ディスクパック上の隣接領域と見なすとよいことがわかる。これらのトラックを一括して、シリンダという。一般に、大量のレコードをディスク上に記録する場合、一シリンダの領域がいっぱいになってから、隣のシリンダへの書込みに移る、という手法がとられる。

三　ISAM（アイサム）——索引付逐次処理法

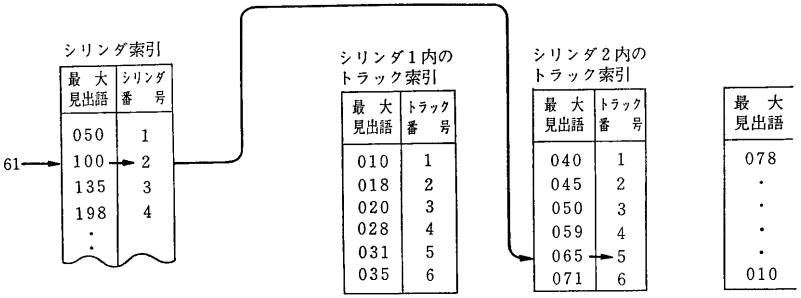
ISAM—Indexed Sequential Access Method—は、すでに一九六〇年代の半ばにおいて開発された、検索可能ファイル処理のための汎用ソフトウェアであるが、ディスク装置の性能が、現在に比べればはるかに低く、また、端末装置を利用したファイル処理等は、数少ない大規模システムでのみ実現可能であった時代の産物であったため、窮屈な制約をもったソフトウェアであり、今後は、VSAMの普及の前に姿を消すことが予想されるが、検索可能ファイル処理法を理解するには、格好の材料であるので、その解説に一節を当てる。

逐次処理ファイルを分類するのに、各レコードに見出語が必要であったが、ISAMファイルの各レコードにも見出語が必要である。ファイルは、全レコードが、見出語の順に並ぶよう生成され、その後のファイル維持、運用に当っては、オーバフロー領域なるものを設けて、つねに逐次処理ファイルとしても利用可能な形態が保持される。これは、検索処理を主体とするファイルにあっても、まず例外なく、周期的な逐次処理を要するので、必須の条件である。そこで、ファイルに対し、レコードの追加が生ずるたびに、それを見出語に合った適正な場所に挿入することが要求されるが、そのたびに、大量のレコードを一挙にシフトすることは実際上不可能であるので、検索可能ファイル処理には、追加レコードの挿入技法が重要な問題となる。

図四(イ) トラック索引によるレコード検索。ここでは見出語が36のレコードを検索している。

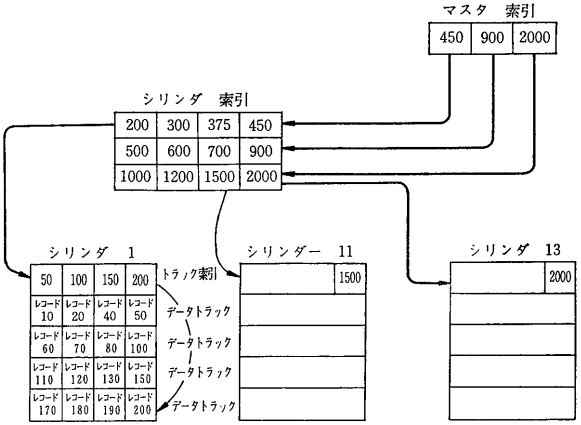


図四(ロ) シリンダ索引とトラック索引との関係



ISAMファイルには、三種類の索引、すなわち、マスタ索引、シリンダ索引、およびトラック索引が用いられる。このうち、マスタ索引は、五シリンダ以上を使用するような大形ファイルにかぎり、実効を有するもので、必要な場合だけ設けられる。トラック索引は、ファイル用に確保した各シリンダの第一トラック上に形成される。シリンダの他のトラックは、データトラックとオーバフロートラックに二分され、ファイル生成時には、レコードはデータトラックにのみ記録され

図四(ハ) ISAM ファイルの索引



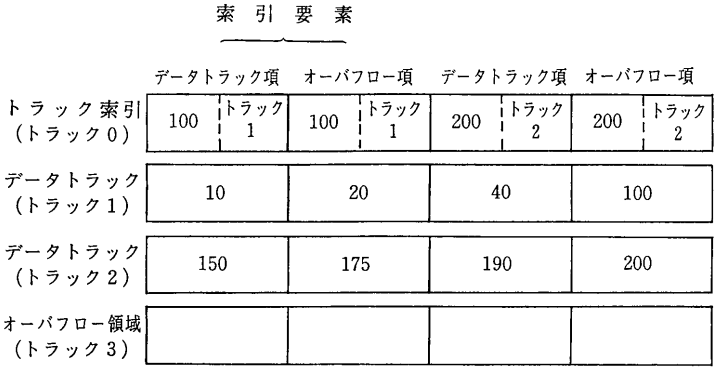
る。トラック索引内には、各データトラックごとに一個の索引要素が作られるが、各索引要素は二項目から成り、その一項目は、対応するデータトラック中の最後のレコードの見出語であり、もう一方の項目には、そのデータトラックからオーバフロウ領域にあふれ出たレコードの情報が書かれている。**図四**(イ)は、ファイル生成時におけるあるシリンダのトラック索引とデータトラックの関係を示している。

シリンダ索引は、ファイルで使用するシリンダごとに一索引要素をもち、そこには、そのシリンダの最後のレコードの見出語が記入されている、いわば、二次索引である。マスタ索引は、**図四**(ハ)によってその機能が明らかなように、三次索引としての機能をもつ。

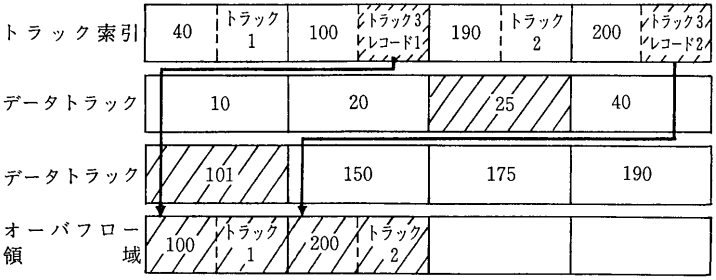
ＩＳＡＭファイルに新レコードを追加するには、索引によってそのレコードに該当するトラックを探し出し、トラック中の適正な場所に置くのであるが、当然、そのトラック中のいくつかのレコードは後方へシフトし、また、それ

図五 ISAM ファイルへのレコードの追加

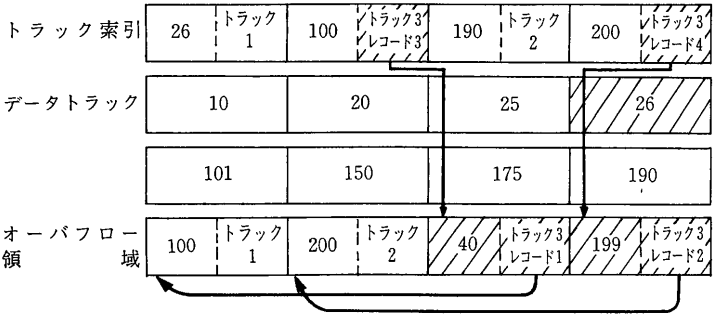
ファイル生成時



見出語25と101とのレコード追加時



見出語26と199とのレコード追加時



までトラック中の最後にあったレコードをオーバフロー領域へ移動する。その模様は**図五**に見られるようなものである。あるデータトラックからオーバフロー領域にあふれ出たレコードは、論理的に一列に繋った状態になるが、これをチェインという。

ファイル処理が進行すると、オーバーフロー領域には、各データトラックからあふれ出たレコードのチェインが入り乱れて交錯し、ファイル処理効率が甚だ低下する。そこで、ＩＳＡＭファイルは、オーバフロー領域上のレコード数がある程度に達した場合、ファイルの全レコードをデータトラック上に再編成—reorganization—しなければならない。この再編成には、ファイル規模が大きくなると、数時間を要することになり、この間、ファイル処理は中断される。また、不要になったレコードは、そのまま放置され、その場所は、再編成時まで他のレコードのために利用されることがない。

四　ＶＳＡＭ（ヴィサム）

この節では、まずファイル索引に関する一般論を述べ、その後でＶＳＡＭを解説する。

四・一　ファイル索引

ファイル索引の機能は、ＩＳＡＭで見たように、一般図書における索引と本質的には同じものである。索引を形成する基本要素は、見出語と、その見出語をもつレコードが記録されているファイル媒体上の位置を示すポインタとが対になったもので、これを索引要素—index entry—という。ディスク上、各索引要素は、アルファベット順、アイウエオ順、あるいは数値の昇列順などによる見出語の順序にしたがって並べられているが、これら

も、計算機本体との間の転送は、一定数ずつまとめられ、物理レコード単位で行われる。大形の索引の場合には、この索引に対して、さらに、索引レコードごとに一つの、最大見出語とポインタとから成る索引要素を作り、それらをまとめて二次索引を作ることがある。ＩＳＡＭのシリンダ索引がその例である。

たとえば、四万の索引要素が、一〇〇個ずつ一レコードにまとめられた、四〇〇レコードから成る索引を考えると、あるレコードを検索するのに、索引内の九レコードを調べることを要するが（二分割法による）、二次索引を作ると、これは索引要素四〇〇、すなわち、四レコードで形成されるから、この場合にはあるレコードを検索するのに、二次索引の高々三レコードと、原索引の一レコード、計高々四レコードを調べれば足りる。

場合によっては、三次以上の索引作成が効果をもつこともあり得る。二次以上の索引がある場合、原索引は、とくにデータレコードの順序を指定するため、シークエンスセット—sequence set—と言い、二次以上の索引をインデクスセット—index set—と言う。

インデクスセットにおける一つの索引要素は、一次元下の索引中の索引要素のグループを指示するが、これは、後者の索引において、索引要素の順序と、それらが配列されている物理的順序とが一致しているからである。データレコードについても、もしも、物理レコード（あるいはトラック）内における各論理レコードの物理的順序が、見出語の順序と一致するならば、シークエンスセットの一索引要素をもって、物理レコード（あるいはトラック）全体を指定することができる。これに反し、物理レコード内における論理レコードの物理的順序が見出語の順序と異るならば、シークエンスセット中に各論理レコードに一つずつの索引要素が必要になる。前者の場合、シークエンスセットは稠密—dense—型であるといい、後者の場合には、非稠密型であるという。

四・二　ＶＳＡＭ

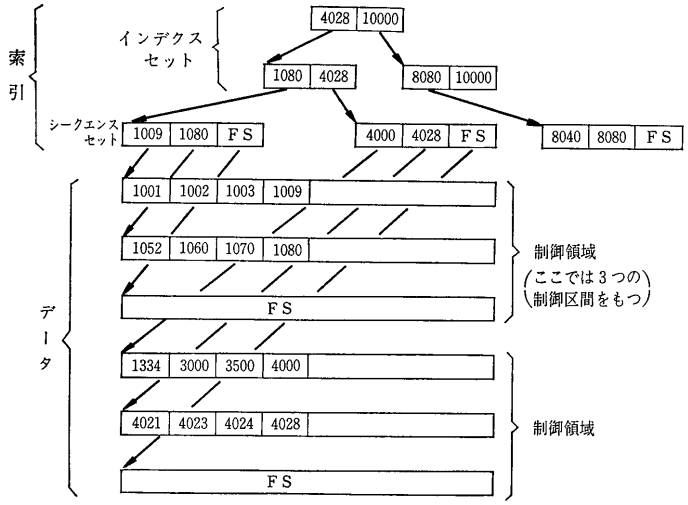
ＩＳＡＭにおいては、追加レコードを、通常のレコード領域とは別の領域を設けて、そこに置いたため、追加レコード量が増加するにしたがい、ファイル処理効率の低下を招いた。ＶＳＡＭの場合には、ファイル生成の頭初において、ディスク上に十分な場所を確保してさえおけば、追加レコードは在来レコードの間の適正な場所に埋め込まれるので、ＩＳＡＭのような処理効率の低下が生じない。

ファイル生成時、ＶＳＡＭは、まず、確保した全空間（これを、データ空間という）を等容量の制御領域―control area―と称する領域に分割し、さらに、各制御領域を、いくつかの制御区間―control interval―と称する小領域に分割しておく。これらの、制御領域と制御区間とは、それぞれ、ディスクのシリンダとトラックというハードウェア上の領域を抽象化したもので、種々のタイプのディスク装置を使用している場合でも、それらのディスク装置のハードウェア特性にあまり拘束されることなく、統一的ファイル規格を定めて、統一的ファイル処理ができるという点で暫新性を有している。

つぎにレコードを書込むのであるが、各制御区間内での論理レコードの順序は、見出語の順序に一致するようにし、さらに、レコードの追加時の操作を容易にするため、どの制御区間への書込みも、全区間を使用せず、半分は空白のまま残しておく。ただし、各制御区間の未使用のまま残しておくべき領域の割合は、必ずしも半分とはかぎらず、ファイル生成時パラメータにより指定することができる。各制御領域についても、ファイル生成時には、そのうちの制御区間の半数のみを使用し、残りは、区間全体を未使用のままにしておく。

ファイル生成時、レコードの書込みと同時に索引も自動的に生成される。その場所は、データファイルと同じ

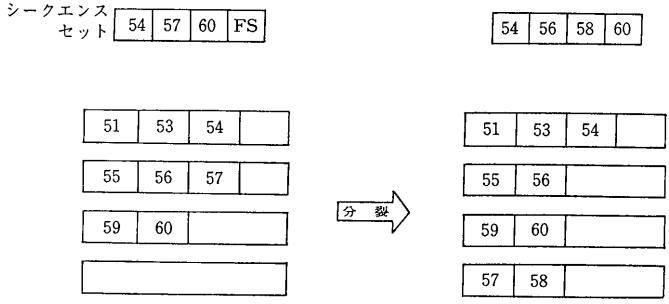
図六　VSAM ファイル　　FS＝余白 (Free Space)



データ空間でも、また、別に設けたデータ空間でもよいが、いずれの場合も、データファイルと同様、制御領域、および制御区間等に分割された空間に生成される。制御区間内の各レコードは、見出語順に配列されているので、索引のシークエンスセットは、各制御区間に対して一つの索引要素をもつ、非稠密型であり、ディスクヘッドのポジショニングタイムをなるべく短縮するため、各制御領域内の一制御区間を、残りの制御区間にあるデータレコードに対するシークエンスセットの部分を記録するのに当てることが多い。

追加レコードの挿入は、VSAM独特の制御区間分裂、あるいは、制御領域分裂を伴うアルゴリズムにより行う。制御区間分裂—control interval splitting—とは、ある制御区間にレコードを挿入した結果、その区間の余白部分が全区間の半分以下になった場合、同じ制御領域内に未使用の制御区間を新たに求め、そこに、前者の区間にあったレコードのうちの半数を移す、という操作の

図七　制御区間の分裂（ここでは見出語85のレコードの追加による）



ことである。また、制御領域分裂—control area splitting—は、制御区間分裂の結果、その制御領域内の未使用制御区間の数が、領域内全区間数の半数以下になった場合、未使用の制御領域を新たに求め、そこに、前者の領域中で使用している制御区間の半数を移す操作である。制御区間または制御領域の分裂が起ると、索引の変更が必要になる。これは、ＶＳＡＭにより自動的に行われる。通常の、分裂が生じない場合のレコードの追加には、そのレコードを挿入する制御区間の一部のレコードをシフトするだけで足り、追加レコードがその区間の末尾に挿入されるとき以外は、索引の変更を必要としない。一方、あるレコードが不要になると、直ちにその制御区間の不要になったレコードより後にあるレコードを前につめ、消滅レコードの場所が有効に利用される。したがって、ＩＳＡＭにおけるファイル再編成作業を必要とせず、ファイルの維持に関しては、いわば、ＩＳＡＭが半自動であるのに対し、ＶＳＡＭの方は全自動化されているのである。

本稿の冒頭で見た志願者ファイルを考えるとき、このファイルを窓口業務にも利用しようとすると、受験番号だけを見出語とするのではなく、氏名によるレコードの索引も必要になる。このような場合、ＶＳＡＭでは、代替索引—alternate index—を作ることができる。代替索引は、それ自体一つのＶＳＡ

Mファイルの形体をもち、データレコードは、代替見出語と原見出語とから成り、このレコードが、ファイルの全代替見出語について作られ、代替見出語順に整列される。代替見出語によりレコードを検索する場合、VSAMは、まず代替索引によりそのレコードの原見出語を見出し、それによって原ファイルの中から所要のレコードを検索する。原ファイルについてレコードの追加あるいは削除があった場合、VSAMは代替索引をも自動的に変更する。原ファイルにおいては、二つ以上のレコードが同一の見出語をもってはならないが、代替見出語についてはこの制限はない。つまり、志願者レコードの受験番号のように、唯一のレコードを指定出来るものを原見出語とし、同姓同名という場合があり得る氏名は代替見出語として使用できる。

後　書

本稿のVSAMの解説は、レコード検索機能に焦点を当てて述べたので、ファイル保護などの他の特徴ある機能については触れなかった。なお、図版は、参考文献[3]のものを参考にして作った。

参考文献

[1] Wagner, R. E., "Indexing Design Considerations", *IBM Systems Journals*, Vol. 12, No. 4, 1973, pp. 351—367.

[2] *DOS/VS Virtual Storage Access Method Planning Guide*, Form No. GC33-5404, IBM Corporation, Data Processing Division, White Plains, New York.

[3] *DOS/VS Data Management Guide*, Form No. GC33-5372, IBM Corporation, Data Processing Division, White Plains, New York.