第３章　入力検索システム

３－１　システム設計の前提

入力検索システムＮＲＤＦ２は、先に開発された同様のシステムＮＲＤＦ１に基いて設計された。ＮＲＤＦ１からＮＲＤＦ２への仕様の改訂において大きく変化した点は、

① システム内部でのデータの表現

② データ間の関連づけ

の２点である。いずれもＮＲＤＦ２への移行において、基礎になる概念を単純化することによって多様な入力データへの対応をはかることを目指した。システム内部でのデータ表現に関しては、データのcompilationとinterpretationという概念を導入し、データのとりあつかい方によって異なる表現のしかたをする。これによってファイル内ではいくつかの異なる様相をもつ部分が現われる。

またデータ間の関連づけのために集合概念が利用された。このことによって、核データの多様性の一部を吸収すると同時にファイルの自己発展のための基礎を与えた。

３－１－１　compilerからinterpreterへ

ＮＲＤＦ１ではcompiler方式をとっていた。（図３－１－１・Ａ）

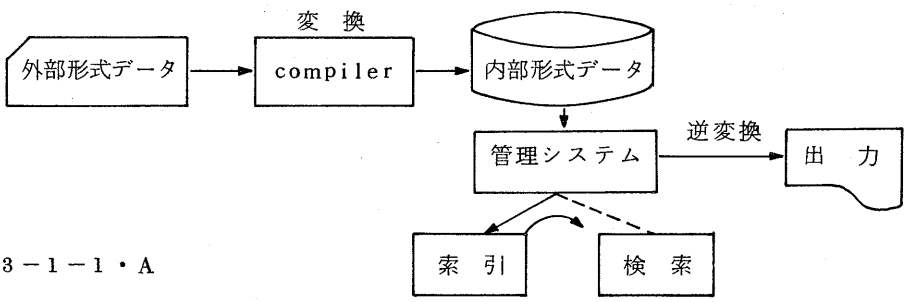


図３－１－１・Ａ

compiler 方式とは、外部形式の入力データをすべて内部形式に変換し内部形式データの中からその一部を索引のための情報としてとり出して登録するものである。この場合、検索を行なって結果を出力するためには内部形式から外部形式への逆変換が必要である。compiler は外部形式、内部形式、Key項目、形式変換規則に関する情報をもつ必要がある。データを内部形式に変換すればシステムにとってデータ処理は容易となり、したがって検索も容易であるが、主たる検索の手がかりとしない情報をも一様に内部形式に変えることは、２次記憶域の使用効率と形式変換のプロセスを必要とする点でマイナスである。一般にある規則でcode化された情報をexplicitなデータ構造の形に展開することは、個別のデータ構造の中に埋め込まれた規則情報の分だけ記憶域を多くとる。ＮＲＤＦ１はこの欠点を明らかにした。

これに対しＮＲＤＦ２ではinterpreter方式をとることにした。

interpreter方式　とは次のようなものである。（図３－１－１・Ｂ）

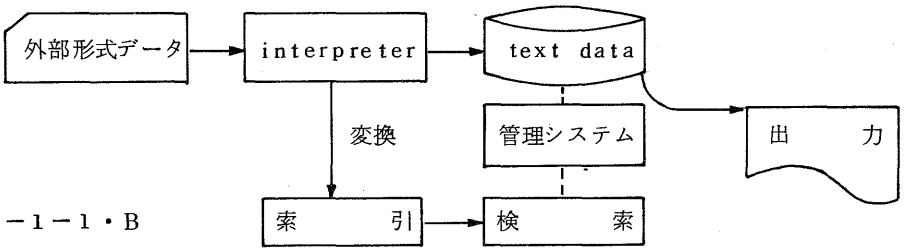


図３－１－１・Ｂ

一定の外部形式で与えられたデータをinterpreterが読み込み、与えられた文法にしたがって内容を解釈し、指定されたkey項目に関する情報のみを抽出しこれを＜key項目－key値＞の対の形をした内部形式に変換して項目ごとの索引（ inverted table ）へ登録する。また読み込んだデータは文法チェックのあと変換なしにtext dataとしてファイルに追加登録する。interpreterのもつ情報は外部形式（文法）とkey項目の種類及び索引のための内部形式である。

interpreter方式は、必要最小限の情報のみを「 compile 」して検索のための内部形式に変えるため、フォーマットの異なるデータの受理に際して、key情報の抽出さえできればよく、compiler 方式のようにすべてのデータを内部形式に変換するという強い条件がつかず抵抗が少なくなる。また必要に応じて部分的に他の内部形式へ変換していく自由度を残している。これは実行時にtext dataの解釈をしながら判断したりデータ操作をしたりすることを意味し、実行速度の低下を無視すれば、interpreterの媒介によって、すでに内部形式で情報が存在しているcompiler 方式の場合のような処理手順と表面的には同じ手続きを実行することになる。

interpreter 方式とcompiler 方式をまとめて比較すると次のようになる。

	interpreter方式	compiler 方式
使用する２次記憶域	少ない	多　い
情　報　操　作	遅　い	早　い
key項目以外の検索	遅　い	早　い
処理プログラムの大きさ	小さい	大きい
異種のフォーマットに対して	抵抗が小さい	抵抗が大きい

結局ＮＲＤＦ２で格納されている情報は、 comment 文として補足的に使うfree text（自然語）と、syntax と code で規制されたcoded informationおよびその一部をexplicitなデータ構造に展開したcompiled informationの3段階の様相をもっている。

free textはman-orientedな情報で、現実世界のできごとを記述するための強力な表現力をもっている。しかしまた反面,一意性や情報密度が犠牲にされている。coded inoformation は人工語であるという点で人間とのよいinterfaceをもち、同時に定義が厳密であるという点で機械にもよくなじむ。compiled informationはcoded information において文法によって一次元的なひろがりの中に圧縮されている「意味」をtableやlistなどのデータ構造の形に変換したもので、機械によるデータ操作のための最適化が可能である。これは machine-oriented な情報である。

free text 型の情報(あるいは学術論文)からcoded informationへの変換は,研究者が入力データをいかに作成するかという人間の側の技術的問題であり、coded informationの型式すなわち入力フォーマットを定める場合にはこの変換による情報洩れや人工語としての表現力の問題が重要となる。これに対してcoded informationからcompiled informationへの変換はシステム開発の技術上の核心である。coded informationからどの範囲の、どの位の深さの内容をどのようなデータ構造に変えるのかということが、システムの性格を決定する。interpreter方式とは、この問題に対してcoded informationの全部ではなく,keyとなるべき一部の情報を索引構造の中に転写しようという一つの方針をとることを示すものである。したがってkey項目の選択、索引構造の詳細については別途これを考慮する必要がある。compiled informationの型式の設定は、データ構造としてシステム内での情報検索戦略の裏付けとなるべきもので、質問文からはじまって目的データに至るまでの情報から情報への経路を準備することである。

3-1-2　データ間の関連とdata set

核反応データ、とりわけ荷電粒子核反応データは、反応様式の多様性を反映して多くの種類のデータをその中に含む。したがってこれを収容するファイルには適応性と記述能力が特に要求される。また、多種多様なデータでありながら、これを計算機独自の表現によらずに、しかも簡単にとりあつかえる形で保持するということは、研究活動の自然な延長としての情報活動を考える時、学術情報にとって基本的な要求である。しかし核反応データの場合、それが多様であっても、単位となる情報はいつもひとつのデータテーブルであって文献情報などと同じように比較的はっきりしている。異る点は、実際上の単位情報はこのデータテーブルに各種の(非数値的な)情報を加えたもので、この付加情報とデータテーブルの、場合によってはネットワーク的な、結びつきが一種の情報構造を与えているということである。この点を糸口としてNRDF2のファイル構成が行なわれる。

一回の核反応実験から得られるデータは、いくつかのデータテーブルの形にまとめられる。これらのデータテーブルおよび関連する情報の集まりをdata streamと呼ぶ。data stream

はいくつかのdata setの集合である。data set とはひとつのデータテーブルとそれに関連する情報の集合であって、その要素はsection と呼ばれる。section には、データテーブル本体のDATA section、書誌情報をもつBIB sectionおよび実験方法に関する情報をもつEXP sectionの3種類がある。data streamの中で複数個のdata set が sectionを共有してもよい。たとえば、BIB sectionは通常 1 data stream中にひとつしかなく、data stream中のすべてのdata setに共有される。入力にあたって各sectionは その種別と帰属するdata set の識別番号を先頭に表示する。sectionはさらに sub section の集合である。sub sectionにはdescription sub sectionとdata sub section の2種類があり、各々その記述のformat を異にしている。

description sub section は、

<項目>=<値>；又は<項目>=(<値>, <値> ,……)；

という形式で非数値的情報を記録している。この形式を statementという。いくつかのstatementを(と)でくくることや、associator(logical pointer)を用いて相互に関連づけをすることができる。また、／*と*／ではさまれたfree text を comment として記録することもできる。

data sub sectionはcolumn free な数値データテーブルで、各数値は可変長であって、空白で区切られていればよい。

このように、データファイルとは表(データテーブル)の集まりであって、検索はその中から希望する一部の表をとり出すことにほかならない。(図3－1－2・A)しかし厳密には数値だけからなる表では、その表の意味内容が何であるのかを示すという自己説明性に欠けるため、表に、書誌事項などの付加情報を合わせてひとつの集合としたものをdata set としてとりあつかいの単位にする。したがって同じ条件の下で実験が行なわれた複数個の表は、実験条件のsection、EXP sectionを互いに共有する複数のdata setとなる。これは、intersection をもちうる集合の集まりとしてデータファイルをとらえるものである。このようなデータファイルでは、異なる表の間の関連は、それらの表を核とするdata set がその外縁に何らかの共通部分(intersection)をもつという形で表現される。(図3－1－2・B)

図3－1－2・A

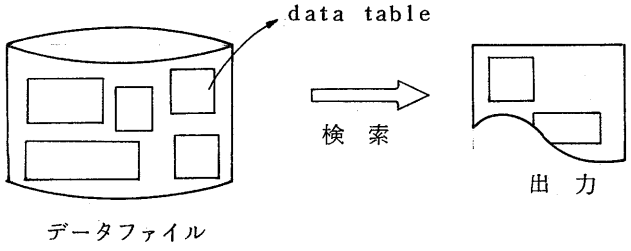
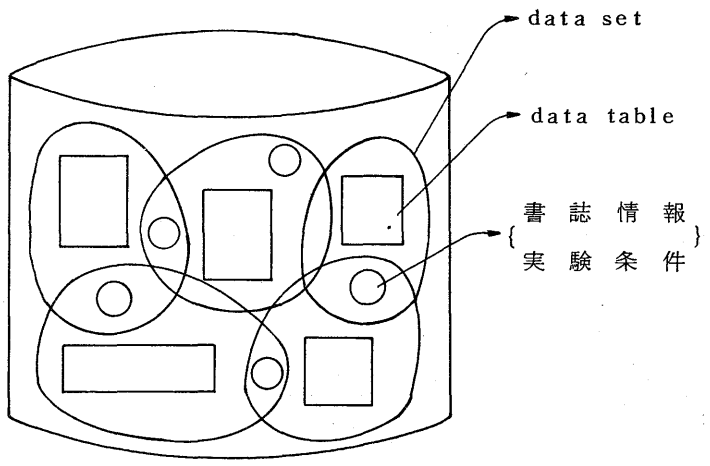


図３－１－２・Ｂ



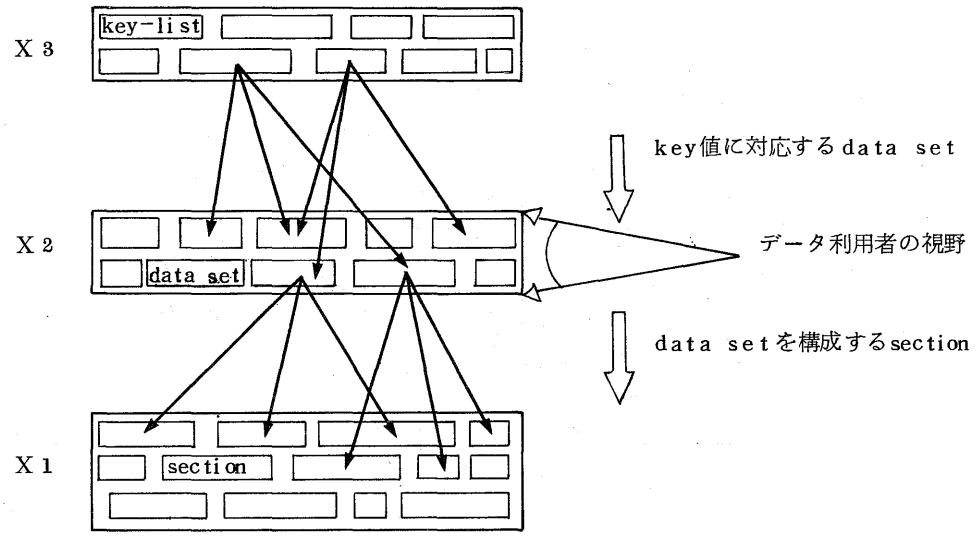
データファイル

３－２　ファイルの構成

ＮＲＤＦ２では、Ｘ１、Ｘ２、Ｘ３という３種のファイルをもっている。Ｘ１は、入力された情報をsection単位で、input imageのまま格納したもので、可変長レコードである。Ｘ２は、data setの集合である。実際には、各data setがsectionの集合であるため、対応するＸ１の要素（ section ）へのpointerの組でひとつのdata setを論理的に表わしている。data setは検索のtargetとなるものであって、データ利用者は、data setの全集合であるこのＸ２ファイルのみを意識すればよい。Ｘ３はＸ２を検索するための各種索引である。索引は、data set中の非数値的情報の中から、システムの定める項目（key項目）についての対応する値にもとづいて作られる。Ｘ３を構成する要素は、key値とそれに対応するＸ２の部分集合である。この部分集合は、実際にはＸ２へのpointerの組で表わされている。Ｘ３はＸ２をその内容にもとづいて分類した情報という意味で、Ｘ２に対する部分転置ファイルである。同様にＸ２はＸ１に対する部分転置ファイルとしてとらえることができる。Ｘ１、Ｘ２、Ｘ３の関係を図３－２に示す。

すなわち、sectionの集合Ｘ1を基底集合とした部分集合族がＸ２であり、Ｘ２を基底集合とした部分集合族が索引Ｘ３である。データ利用者の論理的な視野を規定するＸ２を中心にとらえれば、Ｘ１はデータの実体を収容しており、Ｘ３はデータの分類を与えているのである。このようなファイル構成に対してシステムは、データ入力時には、入力されたデータを適宜Ｘ１、Ｘ２、Ｘ３にふり分ける。またデータ検索時には、データ利用者のＸ２上での探索のために、必要な時点でシステムが自動的にＸ１、Ｘ３を参照して、要求されたデータを取り出す。

図３－２



３－３　検　　索

ＮＲＤＦ２において検索操作とは、希望する条件を満たすデータテーブルを探し、とり出すことである。ここでデータテーブルとは実際には、表とその表に関係する非数値的情報をワンセットにしたdata setであって、このdata setの集合であるファイルＸ２が検索の主な舞台となる。

検索の操作はしたがってＸ２からひとつの部分集合をとりだすことであり、このとき望みの部分集合を得るために利用されるのがＸ３である。また最終的にとり出すべきデータ本体はＸ１に格納されている。Ｘ３は検索する者にとってはファイルの分解能を与える役割をもっている。すなわち、大雑把な分類しかもたないｉndexでは粗っぽい検索しかできず、詳細にわたる情報にもとづくｉndexは精密な検索操作を保障する。

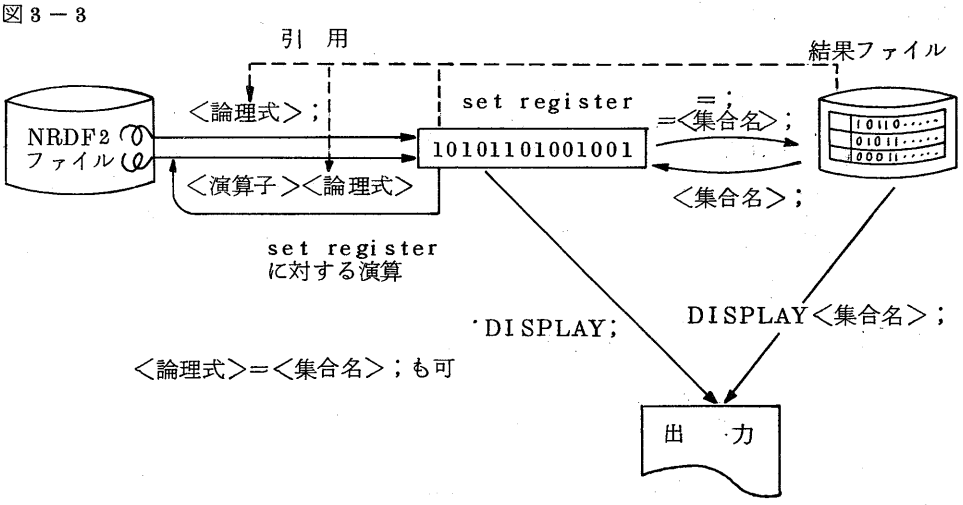
ファイルに対する質問は：

＜論理式＞＝＜集合名＞；

の形式で与えられ、＜論理式＞は（ＡＴＨ＝Ａ．ＢＣＤ）のようにkey項目とその指定する値を関係演算子＝，＞＝ｅｔｃでつないだものを要素として任意の論理条件を組合わせることができる。＜集合名＞は＜論理式＞で指定された条件にもとづいて検索を行ない、その結果として得られるdata setのひとつの集合に付ける名前で、この名前の下に一時記憶域に結果が格納される。＝や＜集合名＞は省略可能で各々省略時解釈が適用される。

この質問から得られる結果は一時記憶域とは別に常に、set registerと呼ばれる特別の作業領域に保持され、これを媒介にデータファイルと検索結果ファイルとの間の情報の受けわたしが行なわれる。（図３－３）

図３－３

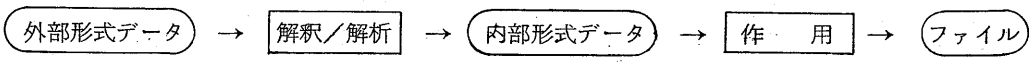


なお検索操作の間、論理式などによってとり出されたデータの集合は常にデータのaddressの集合であってデータそのものではない。このデータ本体は、最終的な出力命令DISPLAYが発せられた時、はじめてＸ２とＸ３を参照してとり出すことができる。また、論理式中で与えられるＡＮＤ、ＯＲ等の演算はそのつど集合演算としてとりあつかわれる。すなわち、set registerの上ではもっぱら集合演算が行なわれその結果として集合が残る。

３－４　システムの構成

ＮＲＤＦ２においてのシステムの基本動作とは、図３－４・Ａのように、人間側から与えられた外部形式データを受けとり、これを解釈・解析し内部形式データに変換し、この内部形式データによってファイルへの各種の作用を及ぼすというものである。

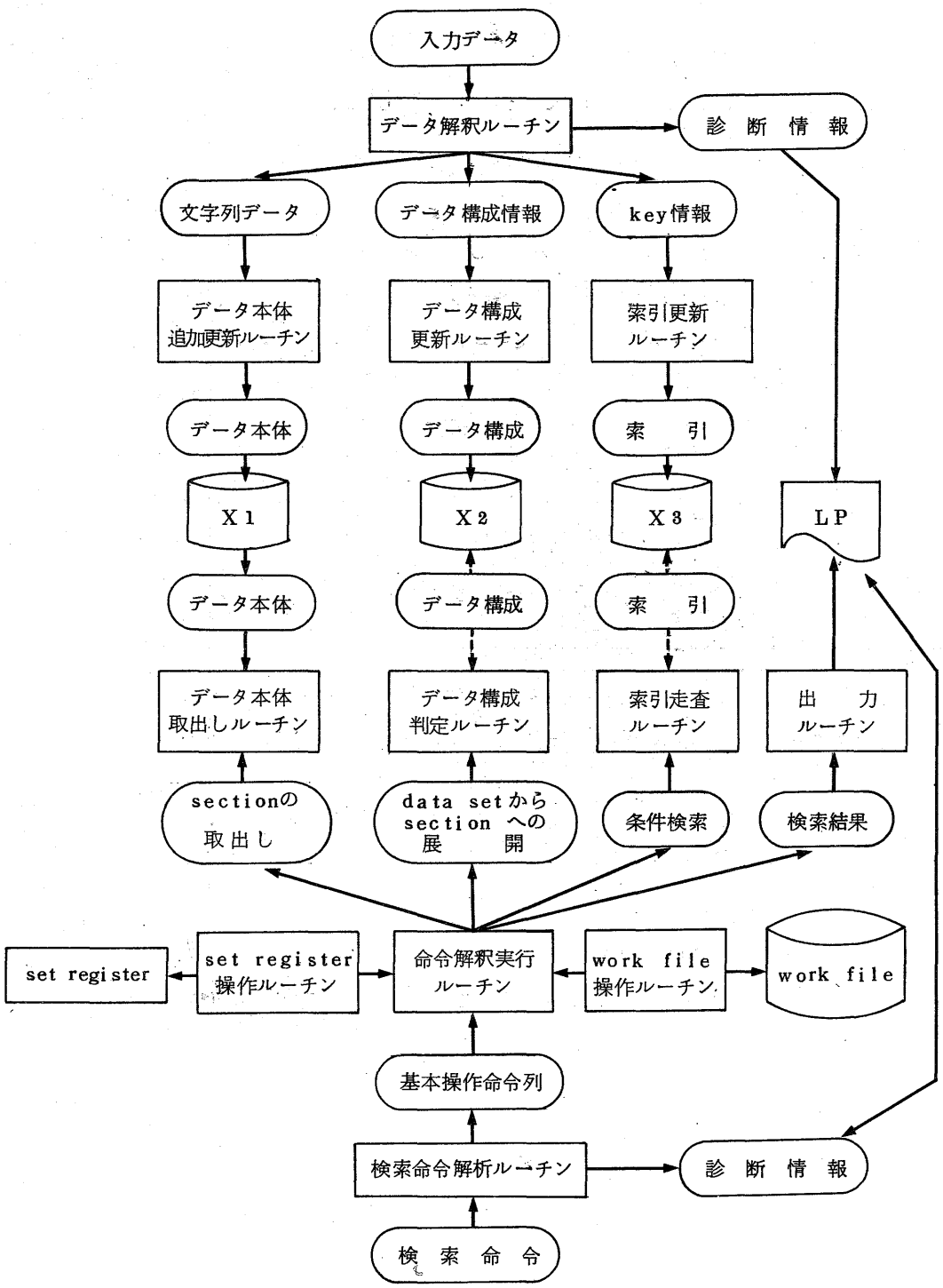
図３－４・Ａ



ここで、データ入力時においてはこの基本的動作は、入力データの解釈と分類、各種内部データの生成、さらにＸ１、Ｘ２、Ｘ３の各種ファイルの更新となる。また、データの検索時には、検索命令の解析と基本命令への変換、そしてその基本命令の実行によるファイルの参照・とり出しとなる。

図３－４・Ｂにデータ入力およびデータ検索のためのシステム構成と、主なデータの流れを示す。

図3－4・B　システム構成


入力データ
データ解釈ルーチン
診　断　情　報
文字列データ
データ構成情報
key情報
データ本体
追加更新ルーチン
データ構成
更新ルーチン
索引更新
ルーチン
データ本体
データ構成
索　引
X1
X2
X3
LP
データ本体
データ構成
索　引
データ本体
取出しルーチン
データ構成
判定ルーチン
索引走査
ルーチン
出　力
ルーチン
sectionの
取出し
data setから
sectionへの
展　開
条件検索
検索結果
set register
set register
操作ルーチン
命令解釈実行
ルーチン
work file
操作ルーチン
work file
基本操作命令列
検索命令解析ルーチン
診　断　情　報
検　索　命　令

3－4－1　データ入力

データ入力時においてはまず、データ解釈ルーチンが起動され、computer readable formatで書かれた入力データを読み込む。この時に、文法上のcheckを行ない、もしデータに文法エラーがあれば、それについての診断情報を出力する。また、システム内で指定されたkey項目リストを参照しながら、key項目についての値の記述があれば、それを抽出してkey情報リストを生成する。さらに、入力データの各sectionごとに表わされたdata setの構成についての情報を抜き出して、sectionとdata setとの間のn：mの（多対多の）対応関係情報リストを作る。最後にシステムは、入力データをsectionごとにちぎって分解し、データ本体の登録にそなえる。

次に解釈ルーチンが生成した各種の情報にもとづいて各ファイルの更新に入る。

まず、抽出されたkey情報のリストにもとづいてこのリストの分だけ索引に追加すべく、索引更新ルーチンを起動して索引ファイルＸ３の更新を行なう。データ構成情報は各data setがどのsectionから構成されているかを表わしており、これにもとづいて、ファイルＸ２を更新する。さらに、sectionごとにちぎられた、文字列のままのデータ本体が、ファイルＸ１に追加される。

3－4－2　データ検索

検索命令が与えられた時には、検索命令解析ルーチンによって、命令の形式のcheckのあと、ひとつの検索命令をいくつかの基本的な操作命令の列に変換する。この時の基本操作命令には、key値指定に対する索引走査、work file操作、set register操作、data setの展開、および出力のためのデータ本体参照操作がある。

これらの基本操作命令列を、命令解釈実行ルーチンが、順次読み込んで実行する。索引走査の際にはファイルＸ３が調べられ、指定された条件にかなうdata setの集合（addressの集合）がとり出される。work file、set registerは、Ｘ３からとり出されたaddress集合の一時保存のための場所として使われ、各集合は名前をつけて参照される。

検索結果の最終的な出力を行なうために、各data setがどのsectionから構成されているのかを調べる必要がある。このために、データ構成情報の集まりであるファイルＸ２が参照され、data setごとに構成sectionのaddressが得られる。さらに、このaddressをもとにして、データ本体が文字のまま格納されているファイルＸ１の中から各sectionをとり出して、出力にそなえる。こうして、Ｘ３→Ｘ２→Ｘ１と順次たどることによって、指定した条件を満たすdata setをLine－Printer上に文字の形で出力することができるようになる。またwork file、set registerは、最終出力に至る途中でのさまざまな途中結果をメモがわりにとっておくために利用される。