情報学基礎 6－4
（1987. 9. 21）

テキストデータベース管理システム SIGMAについて

有川 節夫* 武谷 峻一** 篠原 武*** 大島 一彦+
原口 誠* 白石 修二+ 川崎 洋治++ 井上 仁++
湯浅 寛子++ 酒井 浩++ 山本 章博++ 宮原 哲浩++

*：九州大学理学部基礎情報学研究施設
**：九州大学工学部中央計数施設
***：九州工業大学情報工学部知能情報工学教室
+：福岡大学理学部応用数学教室
++：九州大学総合理工学研究科情報システム学専攻

著者らは，テキストデータベース管理システムSIGMAを1981年に開発し，九州大学大型計算機センターで公開している．近年著しい日本語テキストの増大，テキストファイルの大規模化・大量化に対処するため，その改訂を行い第2版を開発した．本稿では，新システムの概要について述べる．検索，置き換えコマンドや，ソーティングでのキーの切り出しに活用している高速パタン照合アルゴリズムを，日本語テキストに対応できるように改良した．このアルゴリズムは，1バイト文字と2バイト文字が混在する日本語テキストをそのまま処理するので，日本語処理のためのオーバーヘッドは特にない．ファイル・システムの改良により，大規模なテキストファイルを効率よく維持できるようになった．今回の改訂により，大規模テキストデータベースの管理や，自然言語の解析，文献検索などへの応用範囲が拡大される．

ON A TEXT DATABASE MANAGEMENT SYSTEM SIGMA

Setsuo ARIKAWA*, Shun-ichi TAKEYA**, Takeshi SHINOHARA***, Kazuhiko OSHIMA+
Makoto HARAGUCHI*, Shuji SHIRAISHI+, Yoji KAWASAKI++, Hitoshi INOUE++
Hiroko YUASA++, Hiroshi SAKAI++, Akihiro YAMAMOTO++, Tetsuhiro MIYAHARA++

* Res. Inst. Fundamental Information Science, Kyushu Univ. 33, Fukuoka 812, JAPAN
** Computation Center, Kyushu Univ. 36, Fukuoka 812, JAPAN
*** Dept. Artificial Intelligence, Kyushu Institute of Technology, Iizuka 820, JAPAN
+ Dept. Applied Mathematics, Fukuoka Univ., Fukuoka 814-01, JAPAN
++ Dept. Information Systems, Kyushu Univ. 39, Kasuga 816, JAPAN

A general-purpose text database management system SIGMA was developed by the authors , and it is open to the public at Computer Center, Kyushu University since 1981. We have revised the system in order to cope with remarkable increases of the scale and number of machine readable Japanese texts in recent years. This paper describes the outline of the revised system. We have replaced the fast pattern matching algorithms, which are utilized in realizing many commands for searching, replacing, extracting keys of sorting, etc., by new ones to deal with any texts including such Japanese texts. Since the algorithms process any texts containing 1-byte and 2-byte characters as they are, there is no additional overhead caused by the replacements. We have also revised the file system to maintain large text files efficiently. The new SIGMA system can be used for the management of large scale text database, the analysis of natural languages, the retrieval of bibliographic data, and so on.

1. はじめに

パーソナルコンピュータやワードプロセッサの急速な発達に伴って，われわれの身のまわりには機械可読な文書が大量に作られ，テキストファイルとして蓄えられるようになっている．このようにして集積されるテキストファイルをもっと有効に活用し，また，ワープロを打つ感覚で手軽に（数メガバイト程度の）大規模データベースを作成したいという要望もでてきている．

著者らは，1981年に SIGMAという名前の情報検索システムを開発し，九州大学大型計算機センターで公開している（[3], [4]）．このシステムは，陽には構造を持たない一次元の文字列データ，すなわち，テキストファイルを対象としたものであり，上で述べた要望を満足するものである．

SIGMA は，ファイル全体を一度だけ先頭から一字ずつ走査するという一方向逐字処理に基本を置いており，文献データベースの作成や検索，自然言語の解析などに活用されている．また，SIGMA は，高度の汎用性を有しており，多数のテキストファイルを管理・検索・編集する機能を持っているため，システムが管理するファイルの総体は，データベースとして機能する．このようなデータベースをテキストデータベースという（[9], [10]）．この意味で SIGMAはテキストデータベース管理システムであるといえる．

著者らは最近，日本語テキストの増大や，テキストファイルの大規模化・大量化に対処するために，SIGMA システムの改訂を行い，第２版を九州大学大型計算機センターのFACOM M-780 上に実現した．本稿では，２章でSIGMA システムの特徴について，３章で改訂の要点を述べ，４章で新システムの実行例を示す．

2. SIGMAシステムの特徴

2．1 研究者向き情報システム ＝テキストデータベース管理システム

SIGMA システムは，文献の蓄積・検索，論文の作成，自然言語の解析，書類の整理，その他データの収集・加工など，研究者の日常的な活動を支援する目的で開発した研究者向き情報システム（[2], [11]）である．現在では１章で述べたように，テキストデータベース管理システムとして位置づけられるものである．

また，個人用のデータベース以外に，グループで共有するためのデータベースを構築することもできる．さらに，層籠の考え方によるバックアップファイルのスタック化や，ファイル化された交信記録をコマンドプロシジャとして実行する機能などを持つ．

2．2 利用例

SIGMA は，きめの細かい検索，特に自然言語の解析などに威力を発揮している．例えば，九州大学教養部の樋口忠治教授はトーマス・マンの研究において，膨大なドイツ語のテキスト（約２６メガバイト）を入力して，テキストデータベース「トーマス・マン・ファイル」[9] を作成し，種々の文脈中における語の用法の解析に活用している[10].

文献データベースを構築するときには，図１のような形式のテキストファイルを，エディタかワープロで作成して，SIGMA システムへ取り込むだけでよい．従って，データベースの作成・維持は極めて容易であり，パソコン（と時間か予算）さえあれば，個人で数万件程度の文献データベースを構築し，他の利用者に提供することも簡単にできる．しかも，検索は逐字的に行うため，転置ファイル方式の情報検索システムで用いられているストップワード（キーワードに指定できない語）を特に設定する必要はないので，検索の精度は高い．よって，（もし，あるなら）標題にofが４回以上現れる文献を検索することなども可能である．また，ファイルを一度読む間に複数の質問を同時に処理できるため効率はよい．

```
$
(TI) Efficient String Matching: An Aid to Bibliogr
aphic Search
(AU) Aho,A.V., Corasick,M.J.
(SO) Comm.ACM,Vol.18,pp.330-340,(1975)
$
(TI) テキストデータベース管理システムＳＩＧＭＡについ
て
(AU) 有川節夫，武谷峻一，篠原武，大島一彦　他
(SO) 情報処理学会研究報告，87-FI-6,(1987)
$
(TI) A Fast String Searching Algorithm
(AU) Boyer,R.S. and Moore,J.S.
(SO) Comm.ACM,Vol.20,pp.762-772,(1977)
$
```

図１．　テキストファイル

2．3 一方向逐字処理

テキストからキーを検出する手法をパタン照合と呼ぶ[7]．その手法の代表的なものとして,Knuth-Moris-Pratt[13]やBoyer-Moore[8],Aho-Corasick[1]のものなどが知られている．前の２つは１度に１個のキーを検出するものであるが，Aho-Corasickの手法は同時に複数のキーを検出するものである．SIGMA では，この Aho-Corasick のパタン照合機械を処理の基本に置いた．それは単純な一次元文字列という構造の上で高度な処理が可能だからである．

Aho-Corasick のパタン照合機械とは,具体的には与えられたキーの集合からそれらを検索する一種の有限オートマトンを構成し，このオートマトンで入力テキストを先頭から１回だけ走査する一方向逐字処理により，キーの存在をチェックするものである．従って，検索には入力テキストの長さに比例した時間が必要なのはやむをえない．

しかし,この手法を SIGMA システムで実現するにあたっては，オートマトンの遷移表のサイズの最小化や，文字コード分割（英字１文字を２つに分割）と表一瞥による高速化などを行い，アルゴリズムを大幅に改良している[5]．また，SIGMA ではこのパタン照合機械を，複数キーの同時検索だけでなく，ソーティング用キーの切り出し，複数文字列の同時置き換えなど多くの場面で活用している．

2．3 領域とファイル構成

SIGMA は，ファイルの高速処理および共有データベースの管理のため，独自のファイルシステムを有し，それによってファイルを管理している．また，オペレーティ

ングシステムが管理しているテキストファイル（外部ファイルと呼ぶ）も簡単に参照できる．SIGMA が扱うファイルは，検索コマンドの索引付け用ファイルを除いてすべてテキストファイルである．

ファイルは用途に応じて数種類に分けられ，種類別にまとめて保管されている．同種類のファイルの集まりを領域と呼ぶ．領域には，個人用のファイルを置くMEMO領域，データベースを共有するための SIGMA領域，外部ファイルの集まりである外部領域がある．各領域間でファイルを転送することができる．

MEMO領域は，図２に示すように，５個の部分領域（区域と呼ぶ）から構成され，各区域間でファイルを転送することができる．メモ区域には，個人用の名前つきのファイル（メモファイルと呼ぶ）が置かれる．他に，いろいろな作業の中間結果を置くための作業用区域，システムとの交信記録（ログファイルという）を置くためのログ区域，検索コマンドの索引付け用ファイルを置くための索引用区域，MEMO領域のファイルのバックアップを置くためのバックアップ区域がある．

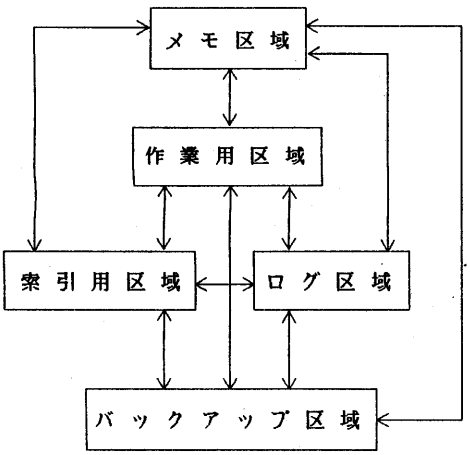

図２．　MEMO領域のファイル構成

SIGMA では，ふだんは作業用区域のファイルを用いて，検索・置き換え・ソートなどを行う．個人用に整理したければ，名前をつけてメモファイルとすればよい．メモ区域を除く全ての区域は，底のあるスタックとして構成され，常に最新のファイルがそのトップに置かれ，除去される場合はそのスタックの底にある最古のファイルがバックアップ区域のトップに移される．こうして丁度屑籠のように，容量に余裕がある限り，除去されたファイルも保存される．これによって，操作ミスによるデータの損失を防いでいる．

2.4　コマンド一覧

SIGMA では表１のコマンドが使用可能である．コマンドを入力する際には，下線部まで入力すれば十分である．

表１．　SIGMA のコマンド一覧

CATENATE	:ファイルを連接する．
COMMENT	:ファイルにコメントをつける．
COPY	:ファイルの複製を作る．
DDIRECTORY	:ディレクトリ（ファイル名一覧）を詳細情報とともに表示する．
DELETE	:ファイルを除去して（消去ではない），バックアップ区域のトップに置く．
DIRECTORY	:ディレクトリを表示する．
ECHO	:ログファイルをコマンドプロシジャとして実行中に，実行内容を表示するかどうかを設定したり，端末にメッセージを出力したりする．
END	:SIGMAシステムを終了する．
GET	:指定するファイルを作業用区域のトップに置く．
HELP	:SIGMAで使用できるコマンドや用語の説明を表示する．
HEXDUMP	:ファイルの16進ダンプをとる．
KEYIN	:作業用区域のトップにあるファイルに，キーボードから直接入力する．
LIST	:ファイルの内容を表示する．
LOAD	:指定するファイルの内容を，作業用区域のトップにあるファイルに読み込む．
LOOK	:作業用区域のトップにあるファイルの内容を表示する．
MOVE	:ファイルを移動する．
NICKNAME	:ファイルに別名をつける．
PROFILE	:システム定数を表示したり設定したりする．
PUT	:作業用区域のトップにあるファイルを指定する所に置く．
REFILE	:検索結果を再ファイル化する．
REPLACE	:複数文字列の同時置き換えをする．
SAVE	:作業用区域のトップにあるファイルの内容を指定するファイルへ書き出す．
SEARCH	:複数キーの同時逐字検索を行う．
SETMEMO	:MEMO領域を初期化する．
SETSIGMA	:SIGMA領域を初期化する．
SORT	:レコードのソーティングをする．
SPACE	:MEMO領域の使用可能量を表示する．
TSS	:TSSコマンドを呼び出す．

2．5　主要なコマンド

SIGMA の特徴となっている検索・ソート・置き換えコマンドについて説明する．

(1)　区切り語と仮想レコード

コマンドの説明の前に，区切り語の説明をしておく．SIGMA が対象とするのは，陽には構造を持たない一次元の文字列（テキストファイル）であるから，検索やソートを行う際に，そのファイルをどのようなレコードの集合とみなすかは，利用者がそのつど指定することになっている．区切り語に指定できるのは空でない文字列であり，複数の文字列を指定することもできる．レコード区切り語は，仮想的なレコードを定めるものであり，２つのレコード区切り語にはさまれた部分をレコードとみなして，検索・ソートを行う．アイテム区切り語は検索における質問式の評価を行う範囲を指定するものであり，

アイテム区切り語を検出した時点で，質問式の評価が行われる．これによりきめの細かい検索を行うことができる．

区切り語の指定は各処理の先頭で行うため，同一のファイルを，利用の目的や利用者の見方によって自由に扱うことができる．例えば，図１に挙げたようなファイルを，ある時は文献の書誌事項が記載されたレコードの集合と見なして文献検索を行ったり，またある時は単語単位をレコードとして単語調査を行ったりするということができる．

(2) 検索（SEARCH）

SIGMA の最も特徴ある機能は検索にある．キーワードとしては，通常のすべての文字列の他に，ワイルドカードを表すトリプル・ドット ... つきの文字列も使用できる．すなわち，X...Yは，Xの後に何か（空でもよい）文字列があってYで終るような文字列を示す．

図１のテキストファイルを例にとる．このファイルを，文献の書誌事項（標題(TI)，著者(AU)，出典(SO)）が記載されたレコードの集合と見なすときには，$ をレコード区切り語とすればよい．また，このファイルでは，$ の直後と各書誌事項の終わりに復帰改行コード（<CR>で表す）が入力されている．そこで，<CR>をアイテム区切り語として，(TI)...String をキーワードにすれば，標題にStringを含む文献を検索することができる．このような設定では，アイテム区切り語は，書誌事項（フィールド）の終わりを指定するという意味を持つ．

キーワード指定の後は，キーワード変数(A1,A2,...)，式変数(Z1,Z2,...) と論理演算子を使って論理式の指定を行う．これらの変数はキーワードの出現の有無ではなく出現回数を記録しているため，さらに算術演算子や比較演算子を使用して，例えばキーワードＡの出現がキーワードＢの出現より多いレコードの検索などもできることになる．

こうして，一次元文字列という単純なファイル構造ではあるが，区切り語の採用とトリプル・ドットを含むキーワードの指定，さらに論理式の記述能力により，作成したファイルの隠された構造を熟知している利用者には，多種多様な検索が可能である．

(3) ソート(SORT)

レコード区切り語で区切られた仮想レコードから，指定された字種のキーを切り出し，そのキーの順序でレコードを並べ換えるコマンドである．SIGMA のSORTコマンドは，固定されたフィールドからキーを切り出すのではなく，例えば，図１のような文献データや論文の参考文献一覧のような構造のないファイルからキーを切り出すところに特徴がある．例えば著者名の昇順で並べ，次いで発行年の降順に並べるなど，優先度の付いた複数のキーを指定できる．また，キーの指定にあたっては，字種のパタンの他にも，切り出しの範囲などの指定も可能である．さらに，空白を区切り語とすれば単語単位のソートを行うこともでき，その結果を頻度表としてまとめる機能もある．

(4) 複数文字列の同時置き換え（REPLACE）

ファイル中の指定された文字列を，全て指定された他の文字列で置き換えるコマンドで，ファイルの１回の走査により，複数組の置き換えを同時に行う．これもパタン照合機械で実現した．この機能は，エディタなどの置き換え機能を一般化したものであり，複数組の置き換えが一度にできるだけなく，巡回する置き換えなどの際の煩わしさを解消できる．例えばBOYS LOVE GIRLS.という文を GIRLS LOVE BOYS. に変更する場合，一般の置き換えでは BOYS→GIRLS として GIRLS→BOYS とすると，結果は BOYS LOVE BOYS.となるため，一度他の使用されない文字列へ待避する操作が必要となる．しかし，SIGMA では BOYS→GIRLS と GIRLS→BOYS の２組を同時に指定するだけで，ファイルの１回の走査により行うことができる．

複数文字列の同時置き換えでは，キーワードの重なり具合いに関する新しい問題が生じるが，SIGMA ではテキストを先頭から走査し，なるべく長い文字列を検出して置き換えるという自然な方法によって解決している[6].

３．改訂の要点

SIGMA システム第１版の公開からほぼ６年を経過した．１章で述べたような環境の変化に対応するため，改訂を行い第２版を開発した．

３．１ 日本語テキストへの対応

今回の改訂の最も重要な点は，日本語テキストへの対応である．そのため，検索・ソート・置き換えでの逐字処理の基本に置いた Aho-Corasick 型のパタン照合機械を日本語テキストに対応させた．日本語テキストは英文テキストと異なり字種が多く，そのため計算機内部では通常１文字が２バイト（英字２字分）で表現されている．また，多くの場合１文字１バイトの文字との混在も許されている．こうした字種が多く異なる符号長の文字が混在するという状況を考慮して，著者らは新しく日本語テキストのためのパタン照合アルゴリズムを開発した[14].

このアルゴリズムは，先に開発した英文テキスト用の高速アルゴリズム[5]で用いた，英字１文字を２つに分割する方法を拡張することで実現できた．具体的には，２バイト系文字と１バイト系文字の切り替え用のシフトコードに関する遷移を，オートマトンの構成時に遷移表

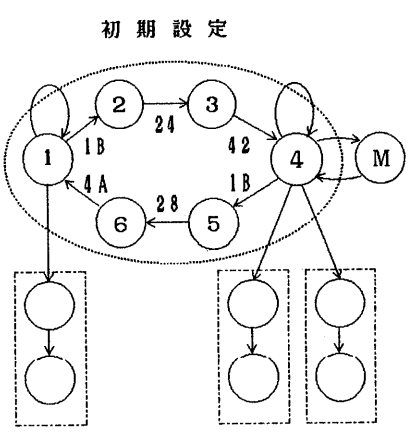

図３.日本語テキストのためのパタン照合機械の構成

の核として初期設定しておくことである．構成の概要を図３に示すが，図中の２桁の文字はシフトコードを構成する文字で，16進数である．図中のコード系は，JIS C 6226に従った．また，遷移は１バイト単位で行うことになっているが，Mは２バイト系文字が遷移する場合の中間状態に対応する．実際の第２版の開発は,FACOM OS IV 上の JEFコードに従い，高速化のため４ビット単位で遷移する手法によった．

実現したアルゴリズムは，１バイト文字と２バイト文字が混在する日本語データを，文字分割の方法を用いてそのまま４ビット単位で照合するものであり，全ての文字を２バイト化するなどの前処理（正規化，[16]pp.124-137）を必要としない．つまり，日本語処理のためのオーバーヘッドは特にないといってよい．よって，実行速度は英文テキスト用のパタン照合アルゴリズムと変わらない．実際，第１版と同じくFACOM M-382 上でも，第２版のSEARCHコマンドは，英文テキスト専用の第１版SEARCHコマンドとほぼ同じ速度で，日本語テキストを処理することを確かめた．

新しい SIGMA では,この日本語用パタン照合機械を逐字処理の基本に置く．

３．２　ファイルシステムの改良

日本語対応以外の改訂点としては，ファイルシステムの改良がある．

まず，個人用のMEMO領域を大幅に拡大した．第１版では，当時の種々の制約から，個人用MEMO領域は４メガバイトとし，その内利用者が利用可能な分は約３メガバイトであった．今回はまずこのサイズを利用者が必要に応じて設定できるようにし，その範囲も１～92メガバイトに拡張して，システム管理用の部分を除き利用者には最大約80メガバイトまで使用できるようにする．また，ブロック長の拡大によって，ディスクへのアクセス回数を減らし，ファイル処理の高速化を達成した．

３．３　ソーティングの機能強化

機能に関する改良としては，ソーティングのキーの切り出し機能の強化が挙げられる．第１版では英文のみを対象としていたため，字種のパタンとしては英字と数字の２種しか用意していなかった．第２版では日本語対応となるため，字種パタンも１バイト系と２バイト系に分けて，さらに漢字，かな，カナ，ギリシャ文字，ロシア文字，記号などきめ細かく用意した．また，英文の単語の調査や解析を行う場合は空白を区切り語として逐字処理を行えばよいが，日本語文では単語が空白で区切られていないため困難が生じる．そのため，キーの切り出しに際しては，細かく切り出しパタンを設定できるように改良した．

さらに，右揃えのソート，右からの逆順ソート（逆引き辞書の作成に有効）などの機能を付加した．結果の表示方法にも多様性を持たせて，索引作成に役立つ形式や指定した付帯情報つきの形式での表示が可能となった．

なお，逐字処理が難しい字種パタンの検出に関しては，第１版では手続的な手法を採用したが，第２版ではこれもパタン照合機械で行えるように工夫した．これによって，SIGMA の文字列走査機能は全てパタン照合機械を用いることになり，逐字処理として統一されたシステムになった．

３．４　開発環境

プログラムの開発には FORTRANを使用した．多人数で作成するため，ソースプログラムにＣ言語のマクロを記述してプリプロセスする工夫をした．これよって，プログラムはより見やすくなり，また共通ルーチンの定義や組み込みも容易になって，開発の作業効率がかなり向上した[15]．

４．　新システムの実行例

SIGMA の特徴となっている検索(SEARCH)・置き換え(REPLACE)・ソート(SORT) コマンドを使った論文作成支援の例をあげる．対象としたテキストは，本稿の草稿であり，ワープロで作成して大型計算機センターへ転送し，SIGMA システムのメモファイルとした．このファイルDRAFT1.TEXT は，全角で１万３千字程度の大きさを持つ．まず，SEARCHコマンドで受身を多用した文を抽出し推敲する（４．１）．次に，REPLACE コマンドで文体を統一する（４．２）．草稿では，引用した順に参考文献を挙げて，この順番で文献番号を打っている．SORTコマンドを用いて著者名と発行年順に参考文献を並べかえ，REPLACE コマンドで参考文献の新しい順に従って，草稿全体の文献番号を打ち直す（４．３）．

最後の２ページはその実行例である（図４）．実行例において，下線を施した部分が利用者による入力を表す．利用者による入力が復改キーのみである場合は，記号↙を用いて表している． SIGMAでは，プロンプトに対して復改キーのみを入力すると，次の状態のプロンプトが出される．説明のために左に番号つけ，実行例の省略は・を縦に３つ続けて表した．

４．１　SEARCH コマンドによる受身文の抽出

日本語テキストを解析し推敲に役立つ情報を使用者に提供する日本語文章推敲支援ツール「推敲」がある[16]．受身文は意味が曖昧になることが多いので，受身文を多用することはよくないと「いわれている」[12]．

「推敲」は，多くの機能を持っているが，その中にテキスト中の受身文の候補をすべて抽出するという機能がある． SIGMAのSEARCHコマンドでも，句点をレコード区切り語に指定して，複数キーワードの同時検索を行うことにより，このような機能を実現することができる．

SIGMA システムを使用するには，公開後は九州大学大型計算機センターを呼び出しREADY状態で，SIGMAと投入すればよい． 終了するには，ENDコマンドを使用する．

(1)まず，このテキストDRAFT1.TEXTを出力してみる．(2)SEARCHコマンドを投入する．(3)レコード区切り語として，全角のピリオド(D1)，半角のピリオド(D2)，記号!(D3)を指定する． !は代用の復帰改行コードある．この指定により，見出しの文などピリオドで終らない文も正確に切り出すことができる． (4)アイテム区切り語として，全角のコンマ(D4)，半角のコンマ(D5)，全角の丸括弧(D6, D7)，半角の丸括弧(D8, D9)，カギ括弧(D10, D11)を指定する．読点や括弧で区切られた部分は，文に次ぐ意味単位を構成しているので，この指定は自然である．読点や括弧で区切られた部分（アイテム）に，「されていると考えられる」などのように，受身が２回以上出現する文は，おそらく推敲を要する文である． (5)キーワードの登録を行う．

受身文を判定するために次の条件（[16]pp.138-143）

を採用する．

「かれ，され，たれ，まれ，られ，われ，がれ，ばれ，しなれ，死なれ」のうちのいずれかに含まれ，かつ「現われ，われわれ」などの明らかに受身ではない語(A11からA23に登録したような語)に含まれない「れ」は，助動詞「れる，られる」の使用を示している可能性が高い．よって，受身・可能・自発・尊敬を区別しない立場に立てば，このような「れ」を含む文は，受身文の候補としてよい．

キーワード変数 A1,A2,...はそれぞれ登録したキーワードの出現回数を記録しているため，

(A1+...+A10)-(A11+...+A22+A23*2)

は，このような「れ」の出現回数を表す．(6)よって，論理式Z1は，受身の候補を含む文を検索せよという意味を持ち，論理式Z2は，読点や括弧で区切られた１つの部分に２つ以上の受身の候補が出現する文を検索せよという意味を持つ．(7)検索するファイル名を投入すると，(8) 検索が行われて質問ごとにヒットしたレコード数（ここでは，条件を満たす文の個数）が表示される．(9)検索結果を表示するように指定する．(10)２番目の質問に対する結果を，(11)新しいレコード区切り語を !#! にして，(12)ファイル T（端末装置を表す特殊ファイル名）に出力する．このようにして，受身を多用した可能性のある文をすべて出力することができる．（この程度なら許容範囲に入っているということで，ここでは修正をしない．）

４．２ REPLACE コマンドによる文体の統一

長い文書を作成したり，あるいは数人で分担した文書を編集する際に，文体や用語，送り仮名を統一する作業にわずらわされることが多い．最近では，漢字を使った難しい表記法は避けて，かなで表記するのがふつうとなってきている．すなわち，下記のリストの左の記法は右のように書き直すほうがふつうである[12]．

「或は→あるいは，普通→ふつう，即ち→すなわち，出来る→できる」

SIGMA システムの REPLACE コマンドは,複数文字列を同時に置き換える機能を持っているので，このような書き直し作業を簡単に行うことができる．

草稿DRAFT1.TEXT 全体を対象にして，上のリストに従い書換え作業を行う．(13)対象となるテキストDRAFT1.TEXT をもう１度出力してみる．難しい表記法が残っているのがわかる．(14)REPLACE コマンドを投入する．(15)上のリストに出てくる置き換えを順次指定する．(16)リストの中の文字列自身は置き換えられないように，指定を追加している．(17)入力ファイルをDRAFT1.TEXT とし，(18)出力ファイル名を DRAFT2.TEXTとして，置き換えた結果に名前を付ける．(19)置き換えた結果を表示する．

４．３ SORT, REPLACE コマンドによる参考文献の並べかえと文献番号の同時打ち直し

論文を書く際に，参考文献の並べかえや引用の仕方に悩まされることが多い．引用する文献を増やしたために，本文中の文献番号を十数個ほど打ち直すようなことは，よく経験することである．SORT, REPLACE コマンドを使えば，このような作業を効率よく行うことができる．

DRAFT2.TEXT を対象にしてこの作業を行う．(20)SORT コマンドを投入する．(21)レコード区切り語を登録する．参考文献一覧では，各文献は ').<CR>' で終わっているので，D1のように登録すればよい．本文の最初から参考文献一覧の前まで（ '参考文献] <CR>' の前まで）の部分もレコードとしてソートして，そのまま参考文献一覧の前にくるように，D2の指定を追加した．(22)キーの切り出し仕様を指定する．'] 'を見つけて(Find)その後に続く半角の英字(Alphabet)を10文字とってきて(Get) 第１キーとし，'(19' の後に続く２桁の数字(Number)をとってきて第２キーとする．K1には第１著者の英字表記がはいり，K2には年代の下２桁が入ることになる．(23)表示はレコード単位とし，(24)新しいレコード区切り語をD1と同じにし，(25)切り出したキーを表示しないようにして，(26)，(27)入出力ファイルを指定する．

(28)並べかえた結果（DRAFT3.TEXT）表示してみると，参考文献一覧の文献が，意図したとおりに並べかえられているのがわかる．今度は，(30)で表示されている順に文献番号を打って，DRAFT3.TEXT 全体の文献番号を打ち直す．(32) REPLACEコマンドを投入し，(33)文献番号の置き換えを指定する．続いて，DRAFT3.TEXT の(29)のところと，(31)のように著者名の英字が残っているところを，(34),(35)で置き換えておく．(36),(37)入出力ファイルを指定し,(38)最終結果（DRAFT4.TEXT）を表示する．

４．４ その他の利用法

上の実行例では，かなり複雑な入力をしたが，次からは，作業の交信記録（ログファイル）をコマンドプロシジャとして実行させればよい．指定の一部（例えば，検索の対象とするファイル名やキーワードの指定）だけを，入力し直すことや，ログファイルをエディタで修正して，使いやすいコマンドプロシジャを作ることなども可能である．

SEARCHコマンドでは，もちろん，外国語のデータと日本語のデータの混ざった図１のような文献ファイルに対しても，書誌事項を対象にきめの細かい検索を行うことができる．アイテム区切り語を指定しなければ，受身の候補が４回以上出現する文を抜き出せというような検索も簡単に実現できる．「ＡのＢのＣの...」のように，「の」が続く文も推敲を要する可能性が高い．読点や括弧で区切られた部分に，「の」が３回以上出現する文を抜き出せというような検索も同様に実現できる．

上で説明してきたように， SEARCH コマンドは，日本語テキストの解析に威力を発揮することが期待される．また，１ＣＰＵ秒で２メガバイト（全角で百万字）程度のファイルを検索するができるので，大規模な日本語テキストにも対応できるものと考えている．

５． おわりに

一方向逐字処理に基づいたテキストデータベース管理システムSIGMA について改訂の要点を述べ，第２版の概要を説明した．第２版では，ワードプロセッサなどでテキストファイルとして蓄えられている日本語テキストが広く扱えるようになり，検索やソーティング，置き換えなどの各種機能も強化された．第２版は，昭和62年10月頃，九州大学大型計算機センターを通じて公開する予定である．さらに多くの利用者に活用されることを期待したい．

[参考文献]

（図４の(39) DRAFT4.TEXTの内容を参照．）

```
(1)DO: LIST DRAFT1.TEXT
   1.   はじめに

     パーソナルコンピュータやワードプロ
   セッサの急速な発達に伴って，われわれ
   ・
   ・
   ・
   [16] Kinoshita木下是雄: 理科系の作文
   技術，中公新書 624 (1981).

(2)DO: SEARCH
(3)  RECORD DELIMITERS
   D1:=.
   D2:=,
   D3:=!
   D4:=↙
(4)  ITEM DELIMITERS
   D4:=,
   D5:=,
   D6:=（
   D7:=）
   D8:=(
   D9:=)
   D10:=「
   D11:=」
   D12:=↙
(5)  KEYWORDS
   A1:=かれ
   A2:=され
   A3:=たれ
   A4:=まれ
   A5:=られ
   A6:=われ
   A7:=がれ
   A8:=ばれ
   A9:=死なれ
   A10:=しなれ
   A11:=現われ
   A12:=分かれ
   A13:=わかれ
   A14:=捕らわれ
   A15:=とらわれ
   A16:=表われ
   A17:=さわれ
   A18:=代われ
   A19:=変われ
   A20:=替われ
   A21:=換われ
   A22:=はまれ
   A23:=われわれ
   A24:=↙
(6)  LOGICAL FORMULAS
   Z1:=(A1+A2+A3+A4+A5+A6+A7+A8+A9+A10)
   -(A11+A12+A13+A14+A15+A16+A17+A18+A1
   9+A20+A21+A22+A23*2)>=1
   Z2:=(A1+A2+A3+A4+A5+A6+A7+A8+A9+A10)
   -(A11+A12+A13+A14+A15+A16+A17+A18+A1
   9+A20+A21+A22+A23*2)>=2
   Z3:=↙
(7)FILE:=DRAFT1.TEXT

(8)  RETRIEVED TEXTS

   QUESTION  1 (Z1 ) =    39
   QUESTION  2 (Z2 ) =     3
   TOTAL             =    39
   CPU (SEC/1000)    =    69
   FILE:=↙

(9)LIST OF RESULTS (N/Y)? Y
   SUMMARY
   QUESTION  1 (Z 1) =        39
   QUESTION  2 (Z 2) =         3
   TOTAL             =        39
(10)QUESTION:=2
(11)NEW RECORD DELIMITER:=1#1
(12)OUTPUT-FILE:=T
   QUESTION  2 (Z 2) =         3

   #
   メモ区域を除く全ての区域は，底のある
   スタックとして構成され，常に最新のフ
   ァイルがそのトップに置かれ，除去され
   る場合はそのスタックの底にある最古の
   ファイルがバックアップ区域のトップに
   移される
   #
   こうして丁度屑籠のように，容量に余裕
   がある限り，除去されたファイルも保存
   される
   #
   読点や括弧で区切られた部分（アイテム）
   に，「されていると考えられる」などの
   ように，受身が２回以上出現する文は，
   おそらく推敲を要する文である
   #
   QUESTION:=↙
(13)DO: LIST DRAFT1.TEXT
   1.   はじめに

     パーソナルコンピュータやワードプロ
   セッサの急速な発達に伴って，われわれ
   ・
   ・
   ・
     長い文書を作成したり，或は数人で分
   担した文書を編集する際に，文体や用語，
   送り仮名を統一する作業にわずらわされ
   ることが多い．最近では，漢字を使った
   難しい表記法は避けて，かなで表記する
   のが普通となってきている．即ち，下記
   のリストの左の記法は右のように書き直
   すほうが普通である[16].
       「或は→あるいは，普通→ふつう，
   即ち→すなわち，出来る→できる」
   SIGMA システムの REPLACE コマンドは，
   複数文字列を同時に置き換える機能を持
   っているので，このような書き直し作業
   を簡単に行うことが出来る．
   ・
   ・
   ・
   [16] Kinoshita木下是雄: 理科系の作文
   技術，中公新書 624 (1981).

(14)DO: REPLACE
(15)X01:=或は
   Y01:=あるいは
   X02:=普通
   Y02:=ふつう
   X03:=即ち
   Y03:=すなわち
   X04:=出来る
   Y04:=できる
(16)X05:=或は→
   Y05:=或は→
   X06:=普通→
   Y06:=普通→
   X07:=即ち→
   Y07:=即ち→
   X08:=出来る→
   Y08:=出来る→
   X09:=↙
(17)INPUT FILE:=DRAFT1.TEXT
(18)OUTPUT FILE:=DRAFT2.TEXT
    TIME FOR REPLACEMENT 41 CPU (SEC/10
   00)
   INPUT FILE:=↙
(19)DO: LIST DRAFT2.TEXT
   1.   はじめに

     パーソナルコンピュータやワードプロ
   セッサの急速な発達に伴って，われわれ
   ・
   ・
   ・
     長い文書を作成したり，あるいは数人
   で分担した文書を編集する際に，文体や
   用語，送り仮名を統一する作業にわずら
   わされることが多い．最近では，漢字を
   使った難しい表記法は避けて，かなで表
   記するのがふつうとなってきている．す
   なわち，下記のリストの左の記法は右の
   ように書き直すほうがふつうである[16].
       「或は→あるいは，普通→ふつう，
   即ち→すなわち，出来る→できる」
   SIGMA システムの REPLACE コマンドは，
   複数文字列を同時に置き換える機能を持
   っているので，このような書き直し作業
   を簡単に行うことができる．
   ・
   ・
   ・
   [16] Kinoshita木下是雄: 理科系の作文
   技術，中公新書 624 (1981).

(20)DO: SORT
(21)  RECORD DELIMITERS
   D1:=).!
   D2:=参考文献］!
   D3:=↙
(22)  KEY WORDS
   K1:=F/'］'/,G/10A/
   K2:=F/'(19'/,G/2N/
   K3:=↙
(23)DISPLAY:=↙
(24)NEW DELIMITER:=).!
(25)LISTING (Y/N):=N
(26)INPUT FILE:=DRAFT2.TEXT
(27)OUTPUT FILE:=DRAFT3.TEXT
       PMM RUN TIME = 113 CPU (SEC/1000
   )
    SORTER RUN TIME = 84 CPU (SEC/1000)
      REFILING TIME = 39 CPU (SEC/1000)
        TOTAL TIME = 236 CPU (SEC/1000
   )
     SORT COMPLETED
(28)DO: LIST DRAFT3.TEXT
   1.   はじめに

     パーソナルコンピュータやワードプロ
   セッサの急速な発達に伴って，われわれ
   ・
   ・
   ・
   開する予定である．さらに多くの利用者
   に活用されることを期待したい．
```

図４．　実行例（続く）

```
(29)                    [).
(30)[10] Aho,A.V. and Corasick,M.J.: Ef
    ficient String Matching : An Aid to
    Bibliographic Search, Comm.ACM, Vol.
    18, pp.333-340 (1975).
    [ 5] Arikawa, S. and Kitagawa, T.:
    Multistage Information Retrieval Sys
    tem Based on Researcher Files, Proc.
     2nd USA-Japan Computer Conf., pp.14
    9-153 (1975).
(31)[ 1] Arikawa有川, 篠原, 白石, 玉越:
     研究者向き情報システムSIGMAについて,
    九州大学大型計算機センター広報, Vol.
    14, No.4, pp.550-573 (1981).
    [ 2] Arikawa,S., Shinohara,T., Shir
    aishi,S. and Tamakoshi,Y.: SIGMA - A
    n Information System for Researchers
     Use, Bull. Inform. Cybernetics, Vol
    .20, No.1-2, pp.97-114 (1982).
    [11] Arikawa,S. and Shinohara,T.: A
     Run-Time Efficient Realization of A
    ho-Corasick Pattern Matching Machine
    s, New Generation Computing, Vol.2,
    No.2, pp.171-186 (1984).
    [12] Arikawa,S. and Shiraishi,S.: P
    attern Matching Machine for Replacin
    g Several Character Strings, Bull. I
    nform. Cybernetics, Vol.21, No.1-2,
    pp.101-111 (1984).
    [ 7] Arikawa有川, 篠原: 文字列パター
    ン照合アルゴリズム, コンピュータソフ
    トウェア, Vol.4, No.2, pp.2-23 (1987
    ).
    [ 9] Boyer,R.S. and Moore,J.S.: A F
    ast String Searching Algorithm, Comm
    . ACM, Vol.20, pp.762-772 (1977).
    [ 3] Higuchi樋口, 篠原: 公用データベ
    ース トーマス・マン・ファイル／SIGM
    Aの公開, 九州大学大型計算機センター広
    報 Vol.16, No.4, pp.379-393 (1983).
    [ 4] Higuchi樋口: 「前置詞」bis の用
    法について, 独仏文学研究, Vol.36, pp
    .183-210 (1986).
    [ 6] Inose, H. (ed.): Scientific In
    formation Systems in Japan, North-Ho
    lland (1981).
    [16] Kinoshita木下是雄: 理科系の作文
    技術, 中公新書 624 (1981).
    [ 8] Knuth,D.E., Morris, J.E. and P
    ratt,V.R.: Fast Pattern Matching in
    Strings, TRCS-74-440, Stanford Univ.
     (1974).
    [13] Shinohara篠原, 有川: 日本語テキ
    スト用の Aho-Corasick 型パターン照合
    アルゴリズム, 情報処理学会研究報告,
    Vol.86, No.48 (NL-52), pp.52.4.-52.4
    .8 (1985).
    [14] Shinohara篠原, 川崎, 有川: マク
    ロ機能を用いたFORTRAN,昭和62年情報処
    理学会九州支部大会講演論文集(予定),
    (1987).
    [15] Ushijima牛島和夫編: 日本語処理
    のためのプログラミング支援環境の構築
    －日本語文章支援ツールの開発を中心と
    して, 九州大学工学部情報工学科 (1986
    ).

(32)DO: REPLACE
(33)X01:=[10]
    Y01:=[ 1]
    X02:=[ 5]
    Y02:=[ 2]
    X03:=[ 1]
    Y03:=[ 3]
    X04:=[ 2]
    Y04:=[ 4]
    X05:=[11]
    Y05:=[ 5]
    X06:=[12]
    Y06:=[ 6]
    X07:=[ 9]
    Y07:=[ 8]
    X08:=[ 3]
    Y08:=[ 9]
    X09:=[ 4]
    Y09:=[10]
    X10:=[ 6]
    Y10:=[11]
    X11:=[16]
    Y11:=[12]
    X12:=[ 8]
    Y12:=[13]
    X13:=[13]
    Y13:=[14]
    X14:=[14]
    Y14:=[15]
    X15:=[15]
    Y15:=[16]
(34)X16:=[).1
    Y16:=[参考文献] 1
(35)X17:=Arikawa有川
    Y17:=有川
    X18:=Higuchi樋口
    Y18:=樋口
    X19:=Shinohara篠原
    Y19:=篠原
    X20:=Ushijima牛島
    Y20:=牛島
    X21:=Kinoshita木下
    Y21:=木下
    X22:=↙
(36)INPUT FILE:=DRAFT3.TEXT
(37)OUTPUT FILE:=DRAFT4.TEXT
     TIME FOR REPLACEMENT 45 CPU (SEC/10
    00)
    INPUT FILE:=↙
(38)DO: LIST DRAFT4.TEXT
    1.  はじめに

      パーソナルコンピュータやワードプロ
    セッサの急速な発達に伴って, われわれ
    ・
    ・
    ・
    開する予定である. さらに多くの利用者
    に活用されることを期待したい.

(39)                    [参考文献]
    [ 1] Aho,A.V. and Corasick,M.J.: Ef
    ficient String Matching : An Aid to
    Bibliographic Search, Comm.ACM, Vol.
    18, pp.333-340 (1975).
    [ 2] Arikawa, S. and Kitagawa, T.:
    Multistage Information Retrieval Sys
    tem Based on Researcher Files, Proc.
     2nd USA-Japan Computer Conf., pp.14
    9-153 (1975).
    [ 3] 有川, 篠原, 白石, 玉越: 研究者
    向き情報システムSIGMAについて, 九州大
    学大型計算機センター広報, Vol.14, No
    .4, pp.550-573 (1981).
    [ 4] Arikawa,S., Shinohara,T., Shir
    aishi,S. and Tamakoshi,Y.: SIGMA - A
    n Information System for Researchers
     Use, Bull. Inform. Cybernetics, Vol
    .20, No.1-2, pp.97-114 (1982).
    [ 5] Arikawa,S. and Shinohara,T.: A
     Run-Time Efficient Realization of A
    ho-Corasick Pattern Matching Machine
    s, New Generation Computing, Vol.2,
    No.2, pp.171-186 (1984).
    [ 6] Arikawa,S. and Shiraishi,S.: P
    attern Matching Machine for Replacin
    g Several Character Strings, Bull. I
    nform. Cybernetics, Vol.21, No.1-2,
    pp.101-111 (1984).
    [ 7] 有川, 篠原: 文字列パターン照合
    アルゴリズム, コンピュータソフトウェ
    ア, Vol.4, No.2, pp.2-23 (1987).
    [ 8] Boyer,R.S. and Moore,J.S.: A F
    ast String Searching Algorithm, Comm
    . ACM, Vol.20, pp.762-772(1977).
    [ 9] 樋口, 篠原: 公用データベース
    トーマス・マン・ファイル／SIGMAの公開,
    九州大学大型計算機センター広報 Vol.1
    6, No.4, pp.379-393,(1983).
    [10] 樋口: 「前置詞」bis の用法につ
    いて, 独仏文学研究, Vol.36, pp.183-2
    10 (1986).
    [11] Inose, H. (ed.): Scientific In
    formation Systems in Japan, North-Ho
    lland (1981).
    [12] 木下是雄: 理科系の作文技術, 中
    公新書 624 (1981).
    [13] Knuth,D.E., Morris, J.E. and P
    ratt,V.R.: Fast Pattern Matching in
    Strings, TRCS-74-440, Stanford Univ.
     (1974).
    [14] 篠原, 有川: 日本語テキスト用の
     Aho-Corasick 型パターン照合アルゴリ
    ズム, 情報処理学会研究報告, Vol.86,
    No.48 (NL-52), pp.52.4.152.4.8 (1985
    ).
    [15] 篠原, 川崎, 有川: マクロ機能を
    用いたFORTRAN,昭和62年情報処理学会九
    州支部大会講演論文集(予定), (1987).
    [16] 牛島和夫編: 日本語処理のための
    プログラミング支援環境の構築－日本語
    文章支援ツールの開発を中心として, 九
    州大学工学部情報工学科 (1986).

DO:
```

図４．　実行例（続き）