解　説

機能メモリのアーキテクチャとその並列計算への応用

5. 文字列照合処理への応用†

高　橋　恒　介††

1. まえがき

初めに文字列照合の必要とされるテキスト検索について説明する．テキストは各種文書に含まれる文字情報部分である．コンピュータで検索される場合のテキストは電子化（コード化）された文字情報をいう．これらのテキストはワードプロセッサやオフィス機器の普及にともなって増加の一途にある．テキスト検索はこのようなテキストから欲しい情報を見つけ出すときに使われる．このテキスト検索の高速化に対する要望がテキスト量の増加にともない，年々，強まってきている．

テキスト検索の基本はテキスト文字列と検索（キー）文字列との間の照合処理である．これは文字コード比較を繰り返すソフトウェアで実現されるが，逐次処理ベースの汎用計算機を使うかぎり，高速化が容易でなかった．したがって，照合ソフトウェアでの文字比較回数を極小にする文字列照合アルゴリズムの改良が 1980 年代に入っても続いている[1),3),4)]．

同じ 1980 年代に，ソフトウェアでなく，ハードウェアでテキスト検索を高速化する研究が半導体デバイスの LSI 回路技術の進歩に合わせるように盛んになった．LSI 回路を活用する文字列照合のハードウェアアルゴリズムとそれに従った文字列照合プロセッサの開発が行われ，高速テキスト検索への応用が試みられるようになった．その中に，機能メモリに分類される文字列照合プロセッサが含まれる．

機能メモリの利用に期待することは照合の高速化である．一般に，機能メモリ LSI 内部においては，メモリマトリックスが行デコーダ，列デコーダで選択駆動されるとも，その過程で記憶データが並列に読出し・書き換えされる．並列に読み出されるデータを並列処理するように演算回路を付加すると，高並列処理が実行される．しかも，メモリマトリックスの大容量化とともに並列処理能力が高められる．詳細な説明に入る前にテキスト DB 検索の必要性にも触れておく．

文書の内容の検索に関してはデータベース (DB) 技術が知られているが，DB 化できる部分は文書の書誌データ（著者名，題名，出版社名，発行年月日など）やキーワードなどである．極端な言い方をすれば，文書の本体を検索せず，文書に付加された荷札を検索する．関係 DB が扱うインデックステーブルはこの荷札の集合である．

このテーブルの中で検索条件にマッチする荷札を選択する，ソートする，ジョインする，プロジェクションする，などが関係 DB の集合演算となっている．これを高速化する研究は，関連文書の検索時間を短縮する点で重要であるが，文書の中味（テキスト部分）を直接に読んで，文字列照合によって内容検索を行うフルテキスト DB の研究も重要な課題である[1),2)]．

その理由はテキストの内容に合ったキーワードや書誌データを作成し，荷札の検索だけで特定文字列を含むテキストを見つけられるように荷札を作成することの難しさにある．人間にはあまりに大きな負担となる．

キーワードや抄録の付加されないテキストの内容は，題名の中の全文字列と検索文字列との照合によってある程度推察されるが，十分でない．多くの場合，本文のテキスト文字列の中での各検索文字列の位置と生起回数を求めるテキスト検索が必要となる．

このような文書データのテキスト検索で要求される文字列照合の処理機能を以下に述べておく．

† Application of Functional Memories to String Matching by Kousuke TAKAHASHI (C&C Systems Research Labs., NEC Corp.).

†† 日本電気(株) C&C システム研究所

2. 文字列照合処理機能

文字列照合の中で最も単純な処理機能は二つの文字列が完全に一致するか否かを判別すること，すなわち，テキスト文字列（たとえば ABABAB-ABCBABB）から検索文字列（たとえば BABC）の含まれる位置を見つけることである．

当然，文字コードは，英語の場合なら ASCII の8ビットコードなどに，日本語なら JIS の16ビットコードなどに統一されるとする．その上での上記の文字列照合は，たとえば，上記文字列テキストの6文字目から9文字目までの文字列が検索文字列 BABC に一致することを見つけ出すことである．このような文字列照合処理機能をもっと細かく分類すると，以下のようになる．

a） 可変長完全マッチ (Variable-length exact match)

任意の長さの検索文字列に完全に一致する文字列をテキスト文字列の中から見つけて，マッチ信号を発生することである．文字列長は，ほとんどの場合，16文字以下である．

b） あいまいマッチ (Approximate match)

a）の完全マッチと対比される．完全マッチでない文字列を見つけ出す機能をいう．文字列照合のときに，検索文字列またはそれに対面するテキスト文字列の中の 1〜2 文字の抜け，混入，置き替わりの誤りを許容してマッチ信号を出す．これは，超伝導と超電導，ファジイとファジー，インデックスとインデクスなどの用語の不揃いや，テキスト文字列作成時のタイプミスや用語表記の不統一に対応するために必要となる．

c） アンカーマッチ (Anchor match)

テキスト文字列に含まれる文字列が，英語のスペースのような区切り文字（アンカー）で区切られている場合に求められる．検索文字列はアンカーの後の文字列とだけ，またはアンカーの前までだけ，あるいはアンカーに挟まれた文字列とだけ照合される．

d） ノンアンカーマッチ (Non-anchor match)

テキスト文字列の中の文字列が，日本語のように，アンカーで区切られていない場合に求められる．検索文字列はテキスト文字列の中の全ての区間の文字列と照合されなければならない．

e） 固定長ドントケアマッチ (FLDC match)

検索文字列の中の一定個数の文字列がドントケア (Don't care : DC) 文字列である場合の文字列照合を FLDC (Fixed Length Don't Care) マッチという．たとえば，！千！百万とか日本!!株式会社などの検索文字列における！が DC 文字である．これらの検索文字列はテキスト文字列の「8千5百万」とか「日本土地株式会社」にマッチする．

f） 可変長ドントケアマッチ (VLDC match)

検索文字列に含まれるドントケア文字の個数を指定したくない場合には可変長 (Variable Length) ドントケア (VLDC) マッチが必要になる．たとえば，VLDC 文字列の記号を＊とすると，日本＊会社はテキスト文字列の中の日本土地株式会社にも日本不動産株式会社にもマッチする．

g） ワイルドカードマッチ (Wild-card match)

ワイルドカード文字はドントケア文字と同じ役目をもっている．テキスト文字列の中での不定部分を表すドントケア文字は検索文字列のどの文字にも一致する．たとえば，ワイルドカード文字を？で表すと，検索文字列 BABC はテキスト文字列 ABA?ABC にも ABAB?B にもマッチする．

h） 近接マッチ (Proximity match)

二つの検索文字列が順序を問わず，1行以内または数 10 文字以内に近接して存在するテキスト文字列を検出する機能をいう．英語の場合には単語数で範囲が指定される．

i） パラレルマッチ

検索文字列が数 10 個に及ぶとき，それらとテキスト文字列との照合を一度に行えるようにすることをパラレルマッチという．ATGTTTCAGA-CTTTTATTT などのテキスト文字列に ATG, TGTT, GACTT, TATT, AGAC などの検索文字列が何回含まれるかを調べるときに必要になる．

j） そのほか

以上の文字列照合のほかに，テキスト文字列から数値文字列の切出し，数値の大小符号比較，VLDC 文字列が数値に限られる正規表現の文字列照合，検索文字列に一致する文字列の別文字列や記号への置換などが検討されている．**表-1** は代表的な照合機能の比較例である．

表-1 基本的な文字列照合機能

Matching Mode	String Pattern	Text Data/ Matched String
Anchor	DEFG	ABC DEFG HDEFGHI
Non-anchor	DEFG	ABCDEFGHDEFGHIJK
FLDC	DE?G	ABCDEFGABCDEXFG
VLDC	DE*FG	ABCDEFCGDEABCFGH
Wild-card	DEFG	ABCDE?GHABD?EFGH
Approximate	DEFG	ABCDEGDECFGHDEAG

3. 照合アルゴリズム

文字列照合を従来からある汎用コンピュータを使いソフトウェアで実行するためのアルゴリズムの研究は非常に多い[3)~5)]. テキスト文字列と検索文字列を主記憶に格納し，各文字列の成分を比較演算回路を含む CPU に読み出して，文字列照合を行う.

3.1 ソフトウェア向き

BF 法 (Brutas Forth) またはナイーブ法と呼ばれる最も単純な文字列照合アルゴリズムはテキスト文字列と検索文字列を先頭から順次比較して，検索文字列の全文字コードで一致があるかを調べる. テキスト文字列長さを n とし，検索文字列長さを m とすると，$(n-m+1)m$ 回の文字比較が必要となる[3)].

mBF (modified BF) 法は検索文字列をそれに対面するテキスト文字列と先頭から比較し文字コードで不一致があるとその先の比較を中止する. したがって比較回数が減少する. しかし，検索文字列の前半に同一文字の繰り返しが多く含まれるテキスト文字列に対しては比較回数が減少しない.

KMP (Knuth-Morris-Pratt) 法は mBF 法を改良し，検索文字列とそれに対面するテキスト文字列との照合を先頭から行う. 文字不一致があるとその位置に対してあらかじめ用意されたシフト制御表を呼び出す. 表から与えられるシフト量に従ってテキスト文字列をシフトさせると，文字比較回数が減る. シフト制御表は検索文字列が与えられた時点で，その中の部分文字列の繰り返しの有無を考慮し，無駄な文字比較を省くように計算される. したがって，文字比較回数が $0(n+m)+rm$ 回に減る. ここに，r は文字列マッチ回数である[4)].

BM (Boyer-Moore) 法は検索文字列とそれに対面するテキスト文字列を検索文字列の最後の文字から順に比較する. 文字不一致があると，KMP 法の場合と同様に，シフト制御表を呼び出し，テキスト文字列をシフトする. KMP 法と違う点は，シフト制御表を 2 種用意し，二つのシフト量の中の大きいほうを選ぶ点，2 種のシフト制御表が検索文字列の最後から始まる部分文字列の繰返しの有無とともに検索文字列の全成分文字を考慮する点である. アルゴリズムが複雑になるが，文字比較回数がベストケースでは $0(n/m)+rm$ に減る. ただし，BM 法は英文の文字列照合に適し，日本語のように文字種類が多いとシフト表算出に時間がかかる.

FSA (Finite State Automaton) 法は検索文字列と対面するテキスト文字列とを先頭から順に比較する. 文字一致があると状態が開始ノードから終端ノードに向かって遷移する. 状態が最終ノードに到達できるか否かで文字列照合を行う. 不一致があるとテキスト文字列をシフトし，開始ノードからの状態遷移を繰り返す.

検索文字列の中に部分文字列の繰返しが含まれる場合に KMP 法と同じようにシフト制御表を使う. したがって，文字不一致のときには状態は開始ノードに戻らないで途中のノードから続きの状態遷移を再開できる. 文字比較回数は KMP 法と同じになる.

AC (Aho-Corasic) 法は複数個の検索文字列の照合を行う FSA を goto 関数と output 関数で記述し，シフト制御表を failure 関数で記述する. それらの関数を検索文字列から計算で求め，テキスト文字列によって起こる状態遷移をトレースして文字列照合を行う. この方法の特徴は一つの FSA と 1 回のテキスト文字列走査でパラレルマッチを可能にする点である[5)]. しかし，failure 関数の作成手順は検索文字列が長くなるにつれ複雑で，その個数が多くなると厄介になる.

以上をまとめ，要求処理機能への対応能力を比較すると，表-2 のようになる. ソフトウェア向きのアルゴリズムの良い点は，汎用のコンピュータであればどこのメーカのものにおいても実行できることである. 欠点は AC 法を除き検索文字列の個数の増加に比例して文字比較回数が増える点である.

ソフトウェア向きアルゴリズムは英語やドイツ語のテキスト検索には実用化されているが，文字種類の多い日本語の文書におけるテキスト検索には性能面で不十分と評価されている.

表-2 ソフトウェア向き文字列照合アルゴリズムの機能比較

モード ＼ 照合方式	mBF	KMP	BM	FSA/AC
可変長完全マッチ	○	○	○	○/○
あいまいマッチ	△	△	×	×/×
アンカーマッチ	○	△	△	○/○
ノンアンカーマッチ	○	○	○	○/○
FLDC マッチ	○	○	○	○/○
VLDC マッチ	○	△	△	○/○
ワイルドカードマッチ	○	△	△	△/△
パラレルマッチ	×	×	×	×/△

○は実現が容易，△はある程度実現可能，
×は実現が困難

しかし，CAM または機能メモリを KMP 法，BM 法でのシフト制御表の記憶や，FSA 法，AC 法の状態遷移表の記憶に利用できる環境が作られれば評価は大きく変わるかもしれない．

3.2 ハードウェア向き

ソフトウェアの場合の文字列照合アルゴリズムは文字比較回数を最小にすることをベストにしたが，ハードウェアの場合は文字照合機能を最大にすることを目指す．そのためのハードウェアアルゴリズムとしては AM 法，CA 法，FSA 法や AC 法などが知られている[6]．

並列比較法または AM 法

文字列照合の最も単純な方法は検索文字列とそれに対面するテキスト文字列をシフトレジスタに貯え，図-1 のようにレジスタ間に文字比較器を並べ，テキスト文字列を 1 文字ずつシフトし，全文字比較器の出力の論理積を出力する並列比較法である．アンカーマッチのときには，1 文字単位のシフトでなく文字比較で不一致があると，次のアンカー文字まで読み飛ばしのできるシフト制御が求められる．

図-1 の中の検索文字列レジスタと比較検索器の部分を図-2 に示すように連想メモリ (Associative Memory : AM，または Content Addressable Memory : CAM) に置き換えた並列比較法は AM 法と呼ばれる．AM 法の特徴は，シフトレジスタを介して与えられるテキスト文字列が CAM に貯えられた複数個の検索文字列と一斉に照合される並列処理にある．エンコーダはマッチ信号の出力端子を減らすために使われている．図-2 は CAM が長い可変長文字列の記憶に対し，大きなデータ幅をもち，短い検索文字列に対し余った部分に DC 文字を記憶できるならば，実用的であるといえる[2),7)]．

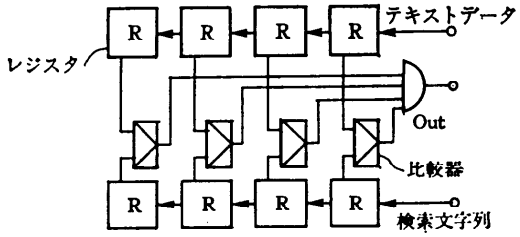

図-1 並列比較法の文字列照合回路例

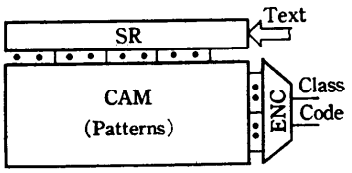

図-2 AM 法の文字列照合回路

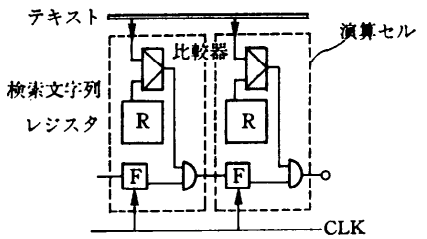

図-3 ブロードキャスト型 CA 法の文字列照合回路

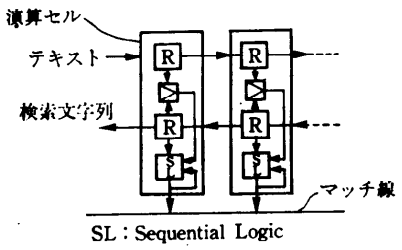

図-4 シストリックアレイ型 CA 法の文字列照合回路

CA (Cellular Array) 法

セルラーアレイ (CA) 法は文字比較の演算セルをアレイ状にカスケード接続して文字列照合を行う．この方法を実現する文字列照合回路は図-3 に示すブロードキャスト方式と図-4 に示すシストリックアレイ方式に分かれる．連結したセルアレイを並列に配列すると，パラレルマッチが可能になる[6]．

演算セルはいずれも文字コードレジスタのほかに文字比較結果をフラグビット信号の形で記憶するフラグレジスタを含んでいる．ブロードキャスト方式 (図-3) の場合，i 番目のセルのフラグ信

号は検索文字列の先頭から i 番目までの文字列がマッチしたことを示す．フラグ信号“1”が最後のセルに到達するか否かで文字列マッチが判定されている．シストリックアレイ方式の場合，循環する検索文字列の先頭のアンカー文字で“1”にセットされたフラグビット信号が次のアンカー文字の到着時まで“1”であり続けるか否かで文字列マッチの有無が判定される．

ブロードキャスト方式のほうがセルサイズが小さくなるが，短い可変長検索文字列の登録時に空きセルをDC文字！で詰める必要がある．シストリックアレイ方式の場合，大き目のセルが2倍必要で，かつ，照合時間が2倍になるが，結合配線が隣接セル間に限られる．

FSA (Finite State Aoutomaton) 法

FSA 法はソフトウェアでなくハードウェアでも実現される．ハードウェアの場合，**図-5** のメモリ回路で文字列照合を行う．IC メモリに状態遷移表が格納され，メモリの出力の一部（状態ノード番号）はアドレス入力端子側にフィードバックされ，出力の残りが文字列クラスを示す．

図-5 における IC メモリのサイズはアドレス側が文字コード幅とノード数の対数の和で，ビット側がノード数の対数とクラス数の対数の和で決められる．文字種類の多い日本語ではメモリサイズがきわめて大きくなる．ハードウェアが大きくなるだけでなく遷移表作成処理が困難となる．

NFSA 法は Nondeterministic FSA 法の略であり，非決定性オートマトンによる複数個の検索文字列照合を可能にする．二つ以上のノードへ分岐させる文字が入力される場合に対し状態遷移図を分岐ノードの前の部分と分岐ノード部分とその後の部分に分割し，それぞれの状態遷移表を三つの IC メモリに分けて記憶する．各遷移表の記憶には図-5 と同じメモリ回路が使われる．これによってパラレルマッチが達成されるが，設計は複雑となる．FSA 法，NFSA 法のいずれの回路も IC メモリに専用処理回路を付加した点で機能メモリとなるが，ベースに逐次処理の思想が残っている．

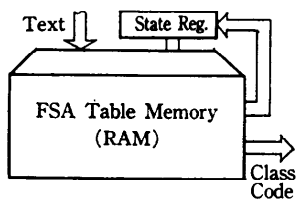

図-5 FSA 法の文字列照合回路

4. 機能メモリでの照合

テキスト検索のための文字列照合プロセッサは論理処理機能とともに，検索文字列のメモリ機能を必要とする．このプロセッサを機能メモリによって実現することを考える場合，CAM のような文字列メモリセルを先に考え，後で多様な照合機能のロジックを加えるアプローチと，文字列照合のロジックを先に考え，後で機能をプログラマブルに変えられるようにメモリを付加する逆のアプローチがある．

4.1 CAM による文字列照合

CAM は AM 法の文字列照合回路の実現に役立つだけでなく，FSA 法の状態遷移表の記憶にも役立つが，可変長完全マッチや FLDC/VLDC マッチやあいまいマッチへの対応に難点がある．図-2 に示したように CAM の入力段に水平シフトレジスタを挿入するには，CAM マトリクスのデータ幅をきわめて大きくしなければならないが，現実の CAM のデータ幅はマッチ線のプリチャージ能力で約 64 ビット（4 文字）以下に制限される．その場合には拡張性が問題になる．

4 文字より長い文字列の照合には次の二つの方法がある．一つ目は，CAM の連結である．出力部にエンコーダ (ENC) を用いない CAM の多数のマッチ線をチップ間で AND 結合することである．二つ目は長い検索文字列をあらかじめ 4 文字以下の部分文字列に分解し，複数の CAM の出力を外部でソフトウェアを使って処理し，マッチする文章を限定することである．

4 文字より短い文字列の登録や FLDC の登録には，DC 文字の記憶が CAM セルに要求されることはすでに述べた．DC 文字を記憶するような CAM セルは単純な CAM セルに較べサイズが大きくなる．セルサイズが小さくならないと，チップ当たりのセル数が制限され，並列度が上がらなくなる．このことは，機能要求が全セルに分散されるような CAM を用いて文字列照合プロセッサを実現することが困難になることを示す．

4.2 プログラマブル順序論理による文字列照合

先に文字列照合のための順序論理回路を与え，

その機能をプログラマブルに変更できるように後でメモリを付加するアプローチを説明する．このような文字列照合法を PSL (Programmable Sequential Logic) 法と呼ぶ．結果は CAM にロジックを付加した機能メモリになる[8]．

図-6 は文字コード列を検出する順序論理回路である．フラグレジスタでのフラグ信号伝ぱんが文字コードに対するマッチ信号Yで制御され，左端レジスタのフラグ“1”が右端レジスタに伝達されるかどうかで文字列マッチか否かを判定する．**図-7** はこの回路での順序論理動作の状態遷移図である．

図-6 の回路のハードロジックな文字コード検出部分を**図-8** (a)(b) に示すように CAM や RAM に置き換えると順序論理回路はプログラマブルになる．k ビット文字コード m 個の記憶に RAM では 2^k の m 倍のメモリセルが必要になり，CAM では mk 個のセルが必要である．

図-7 の状態遷移図のノードと遷移パスが1対1に図-6のレジスタRとANDゲートに対応していることを利用すると，1文字の抜け，混入，置換の誤りを許容する**図-9** の状態遷移図から，それに対応するあいまい文字列照合の順序論理回路を容易に設計できるようになる．

図-10 は LSI 化された PSL 法の文字列検索プロセッサ ISSP (Intelligent String Search Processor) の基本回路構成を示す．順序ロジック (SLC) 部があいまい文字列照合を可能にするように2列のレジスタ (SR, AR) とそれらを結合する AND ゲートで構成されている．

文字列マッチ信号を発生する連想メモリ (CAM) 部はメモリセル数を減らせるように，そしてドントケア文字の記憶を可能にするように，分割 RAM のワイヤド AND 結合を採用している．ISSP では，16 Kb の SRAM を使って16ビット文字コード512個を記憶できる．

可変長文字列照合を可能にするには，SLC 部の各段にデリミタレジスタ (DR と略する) を設けている．DR のピッチに合わせ，順序論理回路 SLC を一定の文字列長 (図では4文字) に区切り，対応する CAM 部の各区間に検索文字列の先頭が登録されたか，途中が登録されたかを，先頭 DR に“1”が書き込まれたか否かで指示する．

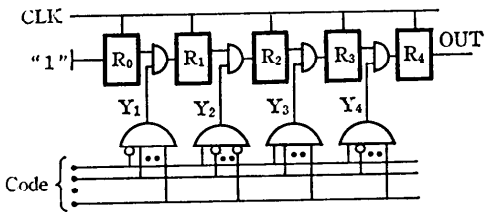

図-6 順序論理回路

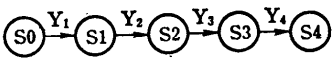

図-7 順序論理回路の状態遷移図

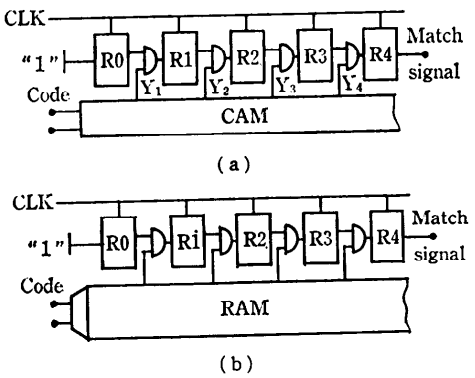

図-8 CAM(a) と RAM(b) を使うプログラマブル順序論理回路

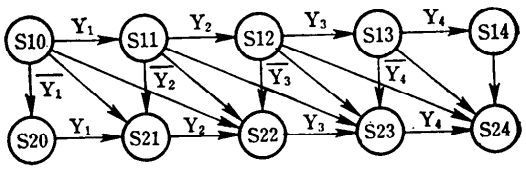

図-9 あいまい文字列照合の状態遷移図

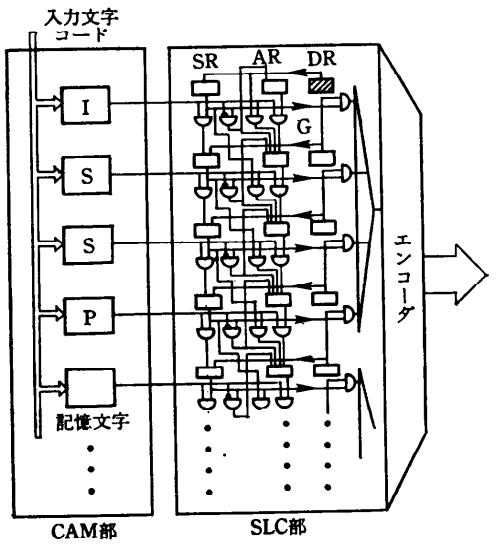

図-10 文字列検索照合プロセッサの LSI 回路構成

そのDRからSLCのフラグレジスタにフラグ“1”をクロックごとにセットするか区切り文字ごとでセットするかで，ノンアンカーマッチとアンカーマッチの文字列照合機能を選択できる．

任意長さの文字列の最後を検出するには，最後の文字位置に対応する段のDRに“1”をセットすればよい．その段のSLC部から出力される文字列マッチ信号とDRの内容とのAND論理結果をORゲートを介してエンコーダに伝えると，可変長イグザクトマッチの登録アドレスをエンコーダから出力できる．

VLDCマッチやワイルドカードマッチの機能も，SLC部にラッチ回路やORゲートを挿入することで実現できる．このことは，SLC部が多様な文字列照合機能を分担し，CAM部で多数の文字列の記憶機能を分担することによって達成されたと言える．

文字列検索プロセッサLSI (ISSP) は512文字の記憶容量をもち，8文字以下の文字列64個を並列に照合し，テキスト文字列を毎秒1000万字の速度 (20 MB/S) で検索すると報告されている[9]．

CAM部では入力文字が常に512文字と並列に比較されるので，ISSPでの毎秒の文字比較回数は51億回に及ぶ．あいまい文字列照合のためにゲートアレイ部で文字比較結果が各ノードからの状態遷移を4つのパスで制御しているので，逐次処理CPUの処理能力に換算すると演算処理速度はさらに4倍高い毎秒約200億回 (20 GIPS) に相当する．このような処理能力はCAMとSLCを一つのチップに搭載し，並列処理を行えるようにした機能メモリのアーキテクチャによって達成できている．**表-3** はハードウェアアルゴリズムの文字列照合機能に対する実現能力の比較結果である．PSL法が最も良い．

表-3 ハードウェア向き文字列照合アルゴリズムの機能比較

照合方法 / モード	並列比較とAM法	B-CA S-CA	FSAとNFSA	PSL
可変長完全マッチ	○／△	○／○	○／○	○
あいまいマッチ	△／×	×／×	△／×	○
アンカーマッチ	○／○	○／○	○／○	○
ノンアンカーマッチ	○／○	○／○	○／○	○
FLDCマッチ	△／△	○／○	○／○	○
VLDCマッチ	△／△	△／△	○／○	○
ワイルドカードマッチ	△／△	△／△	○／○	○
パラレルマッチ	△／○	△／△	△／△	○

○は実現が容易，△は実現不可能でないが，複雑，
×は実現が困難

5. そのほか

過去の文字列照合アルゴリズムでは文字比較回数を最小限にすることに目標が置かれていたが，機能メモリLSIを使える時代にはその目標がもっとあいまいな文字列のもっと高度な並列照合に変わった．

たとえば，PSL法による文字列プロセッサISSPを改良したDISPと呼ばれる辞書検索プロセッサのLSIチップがすでに開発されている[10]．

CAM部はISSPの場合と同様，RAMを組み合わせた回路になっているが，16エリアに分割され，選択されたエリアのCAMだけがSLC部へ接続される．CAM部の記憶容量はISSPの場合の16倍，160 Kb (8192文字) である．

SLC部は2文字誤りを許容するあいまいマッチ機能のほかにアンカーマッチや近接マッチなどの多様なマッチ機能を備えている．また，検索条件を検索文字列別にSLC部に設定できる．すなわち，別々の検索条件による検索結果を出力する．エリヤの切替りに対しては，検索条件をCAM部の各エリヤに書き込めるようにしている．

このプロセッサに登録される検索文字列やプロセッサによって検索されるテキストはフィールドに分類されることを前提にしている．フィールドの異なる検索文字列をCAM部の別々のエリヤに登録すると，各検索文字列が同じフィールドのテキストデータとのみ照合される．このためのフィールドマッチ機能を与える小型のCAMがDISPチップに内蔵されている．

このプロセッサの検索速度が毎秒3000万文字に及ぶ．これはエリヤ当たり毎秒150億の文字比較の処理速度に相当する．記憶容量増加に比例した性能向上になっていないが，フィールドマッチによるテキスト選択を含めた総合性能は上記処理速度の16倍に増える．

CAMの大容量化はDRAMセルを用いるとさらに進展する．それを裏づけるものに，4 Mbitのソーティング機能を含むワードパラレル・ビットシリアルなCAMチップの提案がある[11]．4 Mbは16ビット文字コード25万6千個に相当する．かなり広範囲の用語辞書の記憶を可能にする．検索速度が3 MW/sに低下しても，毎秒の文字比較

処理が約4千億に及び，並列照合処理が性能を高く保つことになる．

文字列照合プロセッサを多数個並列に動作させると，毎秒の文字比較数はLSIチップ数に比例して増加する．しかし，検索時間の短縮と検索文字列の増加は両立できない．登録する検索文字列を増やすとテキスト検索時間は減少しないし，テキストを分割し，並列入力して検索時間を減らそうとすると，チップが複数でも，検索文字列数を増やせない[12]．

なお，テキスト検索時に文字列照合によって得られるデータとしては次のようなものがある．

①テキストレコード当たりの文字列マッチ回数
②各レコードでのマッチ文字列位置
③レコード別マッチ文字列の生起分布

処理速度が高いほど，検索結果の処理量が多くなるので，出力段にバッファメモリを使う必要が起こる[12]．

このような文字列照合プロセッサはテキスト検索だけでなく，表形式に構造化された関係データベースの検索にも利用できる．たとえば，指定属性の8文字以下のレコードが10万件のときにISSP 1チップでの検索時間が80 msとなる．

数値のレンジ検索はFLDCマッチ機能を使うことで達成される．たとえば，479～512のレンジ検索を行うとき，ドントケア文字に！を使い，479，48!，49!，50!，511，512を検索文字列として登録すると，レンジ検索がテキストの検索と同じ要領で実行される．検索時間は3桁数字のレコードが10万件なら約30 msとなり，十分短い．

ISSPではソートやジョイン演算を実現できないが，4 Mbソーティングメモリのように，データをソートしながら書き込めるとソートも機能メモリで達成される．このような機能メモリがDB処理専用プロセッサとして汎用性を備えるには制御方式の改良が課題となる．

6. おわりに

文字列照合アルゴリズムとそれを実現するハードウェアの説明から機能メモリの文字列照合への応用を説明した．文字列検索プロセッサはCAM部に高並列のロジックを結合した機能メモリであって，CAM部のメモリ容量を増加させることでこれまでにない高速の文字列照合を可能にする．このプロセッサがテキストデータベースだけでなく構造化されたデータベースの検索の高速化にも応用できることも述べた．

参考文献

1) Faloutsos, C.: Access Methods for Text, ACM C. Surveys, Vol. 17, pp. 49-74 (1985).
2) Lee, D. and Lochovsky, F.: Text Retrieval Machine, in Sec. 14 of Office Automation, Springer-Verlag, pp. 338-376 (1985).
3) Sedgewick, R. (editor); Algorithms, Addison-Wesley Publishing Company, pp. 277-304 (1988).
4) Baeza-Yates, R. A.; Algorithms for String Searching: A Survey, SIGIR FORUM, Vol. 23 (3, 4), pp. 34-58 (1989).
5) 有川，篠原，その他：テキストデータベース管理システムSIGMAとその応用，情報処理研究報告 Vol. 87 (66), FI-14-7 (July 1989).
6) Hsiao, D. K., editor; Advanced Database Machine Architecture, Chapt. 9 (pp. 256-299), Prentice-Hall, Inc. (1983).
7) Ogura, T., Yamada, J., Yamada, S. and Tanno, M.; A 20-kbit Associative Memory LSI for Artificial Intelligence Machines, IEEE J. Solid State Circuits, Vol. 24(4), pp. 1014-1020 (Aug. 1989).
8) Takahashi, K. et al.: A New String Search Hardware Architecture for VLSI, Computer Architecture News, Vol. 14(2), pp. 20-27 (1986).
9) Yamada, H. et al.; A High-speed String Search Engine, IEEE J. Solid-State Circuits, Vol. 22, pp. 829-834 (Oct. 1987).
10) Motomura, M. et al.: A 1.2-Million Transistor, 33-MHz, 20-b Dictionary Search Processor (DISP) ULSI with a 160-kb CAM, IEEE J. of Solid State Circuits, Vol. 25 (5) (1990).
11) Okabayashi, I., Kotani, H. and Kadota, H.: A Proposed Structure of 4Mbit Content-addressable and Sorting Memory, Proc. of Sympo. on the VLSI Circuits, 10-8, pp. 109-110 (June 1990).
12) 井上，速水，福岡，鈴木，松永：データベースプロセッサRINDAの設計と実現，情報処理学会論文誌，Vol. 31 (3), pp. 373-379 (Mar. 1990).

(平成3年7月3日受付)

高橋 恒介（正会員）

1940年生．1964年慶應義塾大学理工学部計測工学科卒業．1966年大学院修士課程修了．工学博士．同年日本電気(株)入社．磁性膜メモリデバイス，高速ファイルメモリシステム，テキスト検索，VLSIアルゴリズムの研究．現在は同C&Cシステム研究所主管研究員．IEEE，電子情報通信学会，電気学会，応用磁気学会などの各会員．