数理解析研究所講究録
353巻 1979年 276-288

逐次検索システムと言語について

九大理　情報研　有川節夫

1. はじめに

逐次サーチに基づく情報検索システムは走査するデータの量に少くとも比例する時間を検索時に必要とする．このために多数の利用者を対象にする一般的なIR システムとしては不向きである．しかし最近の廉価で高速な外部メモリの開発と普及によって、利用者に新しい問題意識が生れてきた．そして効率的な逐次サーチに関する理論的研究の進展によって、こうした問題の解決が可能になりつつある．

本稿では、数年来筆者等が開発して使用している一方向逐次サーチに基づく情報検索システム TEXTIR (Text Information Retrieval System) について、その概要及び能力について考察する．

2. 準備

Σを字母とする．レコードはΣ^*の要素であり，ファイルはΣ^*の部分集合であるといえる．ファイルはある規則(方針)に従って集められるので、その規則を文法Gと考えると、ファ

イルは文法 Gによって生成される言語 $L(G)$ということにする。通常ファイルは $L(G)$ の有限な部分集合である。

逐次検索システムにおいて質問Qを満たすレコードの集合を検索することは、ファイル $L(G)$ と質問 Qが定義する文法 G'によって定まる言語 $L(G')$ との共通部分 $L(G) \cap L(G')$ を作ることに対応する。質問が $L(G)$ 又は Σ^*の或る明確な部分族を定義できれば、システムの能力を知る上で都合がよい。システムと質問文によって定義される言語族が正規集合族と一致するとき、システムを R-万能ということにする。逐次検索システムをこのように形式言語との対応関係で把えることによって、システムの能力の定性的評価が可能になる。

さて、TEXTIR は逐次サーチの部分に Aho-Corasick [1] によるパタン・マッチングマシンを使用している。パタン・マッチングマシンは Σ^* の要素としてのテキスト系列を左から右へ一方向に一度だけ走査して、与えられたキー（Σ^+の元）の有限集合 K の要素の存在場所をことごとく知らせてくれる一種の有限オートマトンである。従って、検索システムには、このように出来上ったオートマトンを系列上に走らせる部分と、与えられた K から有限オートマトンを作る部分が必要である。後者は goto 関数構成部と failure 関数構成部から成りたっている。

3. TEXTIR の機能と特徴

TEXTIR はパタン・マッチング・マシンを土台にして、次のことに留意して構成した，即ち，

a)パタン・マッチング・マシンを効率良く実現すること．

b)機能をできるだけ拡張追加すること．

c)しかし，そのために速さと効率を犠牲にしないこと．

その結果，結局もとの Aho-Corasick システムのもつ機能に加えて次の機能と利点をもつものになった。

1) $\Sigma^* x_1 \Sigma^* x_2 \Sigma^* \cdots \Sigma^* x_n \Sigma^*$ 形の言語の認識．この機能はキーの出現順序が意味をもつような検索において特に有効である．例えば，x followed by y の形の検索は，いわゆるトリプル・ドットを含むキー$A = x \cdots y$と論理式 $Z = A$ なる単純な質問で可能である．

2) 自由なレコード形式．このシステムでは検索の対象となるレコード形式を特定のものに限定していない。ファイル全体を1つの長い文字列とみて検索を行う．必要なレコードは，出力区切り語を通常のキーと同様に登録して，2つの出力区切り語で囲まれた区間として、検索時に自動的に取出すことができる．

3) 複数質問の同時処理．TEXTIR は一方向の逐次サーチに基づいているために，検索時間は必然的に走査対象のファイ

ルの長さに比例する。そこで，一連の検索における全体の検索時間を短縮するために，最高15個までの質問を同時に処理できるようにした。この場合も作成されるパタン・マッチングマシンは唯1個である。

4) ユニバースの制限．質問は Σ^* のある部分集合を定義するわけであるが，字母 Σ は実際には，計算機で使用できる全ての文字や記号からなる．質問によっては，このユニバース Σ^* を適当な部分集合，例えば，Σ_0^* ($\Sigma_0 \subset \Sigma$) に制限したくなる場合もある．これは理論的にはいつでも可能であるが，実際には質問において多数のキーを登録しなければならないため実用的ではない．そこで TEXTIR では *no reset* というモードを指定することによって，初期状態への *failure transition* を，次の出力区切り語を検出するまで禁止することにした。これによって，キー及び入力・出力区切り語に使用された語の集合を X とするとき，ユニバースをほぼ X^* に制限することが可能になった．このモードの選択による検索時間への影響はない．

5) メモリの節約．Aho-Corasick の技法を直接的に採用すると，パタン・マッチング・マシンを実現するには，goto 関数を状態図で表わした場合の「ノード数」+「終点ノード数」$= n$ とするとき，サイズ $3n$ の遷移表が必要である．しかしこの表の

goto関数の行は，任意の状態 k に対して $goto(k)=k+1$ となるように出来るので，除去できる．したがって表のサイズを $2n$ に縮めることができる．

6)検索時間の短縮．遷移表のサイズを固定して，無駄な ε-動作を出来るだけ減少させるようにした．（文献[4]参照）

7) R-万能性．式変数と ε-変数@ を論理式に許すことによって R-万能性を得た．したがって，このシステムで少くとも理論的にはすべての正規言語を定義することができる．更に正規文法で書かれたレコードの構文チェックも可能となった．

4. 質問文と検索

TEXTIRでは，区切り語やキー，論理式の登録に先立って，トリプル・ドットを表わす記号，復改用の記号，エラー訂正用の記号，reset/noreset の指定を行うことが出来る．勿論システムには標準的な記号，モードが仮定されているので，これらの指定を省略することもできる．

1)区切り語．区切り語は Σ^+ の要素であり，出力区切り語と入力区切り語とがある．出力区切り語は長いテキスト系列の中に仮想的なレコードを設定して，入力区切り語は論理式を評価すべき区間を指定する．「通常は」出力区切り語は入力区切り語としても働く．システムは入力区切り語を検出するたびに論理式を評価し，出力区切り語を検出した時点での論理

式の値が1(true)であるような質問に対する検索結果として、最近の2つの出力区切り語で囲まれた仮想的レコードを出力する. 図1の場合には、「通常の」検索においては、出力区切り語 u_3 を検出した時点で論理式Zの論理和が

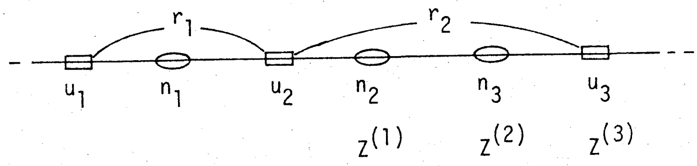

図1　出力／入力区切り語
□ : 出力区切り語 (u_1, u_2, u_3)
○ : 入力区切り語 (n_1, n_2, n_3)
$z^{(i)}$: 論理式の値

$$Z^{(1)}+Z^{(2)}+Z^{(3)}=1$$

であるときに、出力区切り語 u_2 と u_3 で囲まれるレコード r_2 をこの論理式に対する検索結果として出力する.

キー、入力区切り語、出力区切り語には、この順序で優先順位がある. したがって、ある記号列がキーでもあり、入力区切り語でもあり、更に出力区切り語でもある場合には、システムは、先ずキーであることをマークして、論理式を評価し、仮想的レコードの切り出しを行う.

2) キー. キーには通常のキーとトリプル・ドット(これを∘で表わす)を含む $x_1 \circ x_2 \circ \cdots \circ x_n$ の形のキーとがある. キー、区切り語の登録はシステムが発行する変数 A1, A2, … に に続いて行う.

3) 論理式. 「通常の」検索における論理式は式変数 Z1, Z2, … に続いてキー変数 A1, A2, … と論理和(+), 論理積(・), 否定(−)及び括弧 [,] を用いて登録する. 一般に TEXTIR では論理

式に式変数 R_i や ε-変数@ を使うこともできる. ε-変数@は特別な働きをする. @の値はシステムが出力区切り語をまさに離れようとしているときだけ1で、その後は次の出力区切り語を検出するまで0である.

論理式の評価は $z_1, z_2, \cdots$ の順序で行われ、出力区切り語を検出して次の仮想レコード上を走査する前に, $z_i := ivz(i)$ によって初期化される. ここに $ivz(i)$ は

$$z_i = f_i(@; A_1, \cdots, A_n; z_1, \cdots, z_m)$$

とするとき,

$$ivz(i) = f_i(1; 0, \cdots, 0; ivz(1), \cdots, ivz(i-1), 0, \cdots, 0)$$

である.

質問リストに@が使用されていないで, どの論理式 $z_i = f_i$ にも式変数 $z_j\ (j \geq i)$ が使用されていないときには,「通常の」方式で検索する.

5. システムの能力

これまでに Aho-Corasick のパタン・マッチング・マシンに論理演算 (+, ・, −) だけを使えるシステムを始めとして,「通常の」検索をする TEXTIR, 一般の TEXTIR の3種類の逐次情報検索システムを考えてきたことになる. 本節ではこれらの能力について考えてみる.

「通常の」TEXTIR においては, キー $A = x$ は言語

$$\|A\| = \|x\| = \Sigma^* x \Sigma^*$$

と定義し，キー $A = x_1 \circ x_2 \circ \cdots \circ x_n$ は言語

$$\|A\| = \|x_1 \circ x_2 \circ \cdots \circ x_n\| = \|x_1\| \cdot \|x_2\| \cdots \|x_n\|$$

を定義すると考えられる．この解釈 $\| \; \|$ は一般の論理式にも

$$\|\alpha + \beta\| = \|\alpha\| \cup \|\beta\|$$

$$\|\alpha \cdot \beta\| = \|\alpha\| \cap \|\beta\|$$

$$\|-\alpha\| = \Sigma^* - \|\alpha\|$$

によって拡張できるから，論理式 $Z_i = f(\alpha_1, \cdots, \alpha_n)$ は言語

$$\|Z_i\| = \| f(\alpha_1, \cdots, \alpha_n \|$$

を定義することになる．

一般の TEXTIR の質問に対しては多少工夫を要するが，同様に $\|Z_i\|$ でもって，論理式 Z_i が規定する言語を表わすことにする．そこで，次のような3種類のシステムによって定義される Σ 上の言語族を考えてみよう．

$A(\Sigma)$ = Aho-Corasick システムによって定義される言語族，

$R(\Sigma)$ = 「通常の」TEXTIR による言語族，

$W(\Sigma)$ = 一般の TEXTIR による言語族．

さらに $Reg(\Sigma)$ で Σ 上の正規言語の族を表わすことにする．こうすると，族 $A(\Sigma)$ や $R(\Sigma)$，$W(\Sigma)$ の大きさ，或いは $Reg(\Sigma)$ との関係でもって，システムの検索能力を知ること

ができる．ここでは $\Sigma := \Sigma - \{\cdot\}$ とし，変数の個数，キーや区切り語の長さに関しては，それが有限であることを除いて特に制限はないものと仮定する．

〈定理1〉 (1) $A(\Sigma) = R(\Sigma) \subsetneq W(\Sigma)$ （$\#\Sigma = 1$ のとき），

(2) $A(\Sigma) \subsetneq R(\Sigma) \subsetneq W(\Sigma)$ （$\#\Sigma \geq 2$ のとき），

(3) $W(\Sigma) = Reg(\Sigma)$.

略証．定義から $A(\Sigma) \subset R(\Sigma) \subset W(\Sigma) \subset Reg(\Sigma)$ であることは明らかである．また $\#\Sigma = 1$ のときは，

$$\| x_1 \circ x_2 \circ \cdots \circ x_n \| = \| x_1 x_2 \cdots x_n \|$$

であるから，$A(\Sigma) = R(\Sigma)$ である．更に $\Sigma = \{a\}$ とするとき、言語 $L_1 = (aa)^*$ は $R(\Sigma)$ に属さないことが次のようにして示される．L_1 が $R(\Sigma)$ に属すと仮定して，キーのリスト

$$A_i = a^{r_i} = aa \cdots a \quad (r_i \text{個}, \ 1 \leq i \leq n)$$

をもつ質問 $Z = f(A_1, A_2, \cdots, A_n)$ が L_1 を定義するとする．論理式 Z は積和形式

$$Z = f(A_1, A_2, \cdots, A_n) = \sum_i A_1^{e_{i_1}} \cdot A_2^{e_{i_2}} \cdots A_n^{e_{i_n}}$$

で表わされる．ここに $e_{i_j} = 0$ 又は 1 であり，$A_j^0 = -A_j$，$A_j^1 = A_j$ である．いま $e_{i_j} = 0$ とすると，

$$\| A_1^{e_{i_1}} \cdots A_j^{e_{i_j}} \cdots A_n^{e_{i_n}} \| \subset \| A_j^0 \|$$
$$= a^* - \| A_j \|$$

$$=\{\varepsilon, a, aa, \cdots, a^{r_j-1}\}$$

となる．L_1 は無限集合であるから，少くとも１つの項 $A_1^{e_{k1}} \cdots A_n^{e_{kn}}$ に対しては，すべての j に対して $e_{k_j}=1$ でなければならない．そうすると

$$\|A_1^{e_{k1}} \cdots A_n^{e_{kn}}\| = \|a^{\max(k_1, \cdots, k_n)}\|$$
$$= a^{\max(k_1, \cdots, k_n)} \cdot a^*$$

となり，$m > \max(k_1, \cdots k_n)$ に対して，

$$a^m,\ a^{m+1} \in \|A_1^{e_{k1}} \cdots A_n^{e_{kn}}\| \subseteq L_1$$

となり，Σ が L_1 を定義することに反する．したがって L_1 は $R(\Sigma)$ に属さない．L_1 が $W(\Sigma)$ に属すことは L_1 が正規であることと (3) に依る．これで (1) が証明された．

(2) は $\Sigma=\{a, b\}$ とするとき，$L_2=\Sigma^* a \Sigma^* a \Sigma^*$ が $R(\Sigma)$ に属すが，$A(\Sigma)$ には属さないことに依る．実際 L_2 はキー $A=a \circ a$ と論理式 $Z=A$ によって定義される．$A(\Sigma)$ に属さないことは上の L_1 に関する議論とほぼ同様にして示される．

(3) は任意の有限オートマトンを TEXTIR の中でシミュレートできることを示せばよい．$M=(K, \Sigma, \delta, s_1, F)$ を有限オートマトンとする．$K=\{s_1, s_2, \cdots, s_m\}$, $\Sigma=\{a_1, \cdots, a_n\}$ として，$a_1, a_2, \cdots, a_n$ を入力区切り語として登録して，各キー変数にキー a_i を $A_i=a_i$ $(1 \leq i \leq n)$ と割り当ててお

く．勿論モードは noreset を選んでおく．そして質問リストを

$$Z_1 = \sum_{(j,k):\delta(s_k,a_j)=s_1} A_j \cdot Z_{n+k} + @ ,$$

$$Z_i = \sum_{(j,k):\delta(s_k,a_j)=s_i} A_j \cdot Z_{n+k} \qquad (1 < i \le m),$$

$$Z_{m+i} = Z_i \quad (1 \le i \le m),$$

$$Z_{2m+1} = \sum_{s_i \in F} Z_i ,$$

とする．$\| Z_{2m+1} \| = L(M)$ であることは明らかであろう．

定理1(3)はTEXTIRがR-可能であることを示している．したがってこれを正規文法で書かれたレコードの構文チェックにも用いることができる．即ち文法を表現する質問リストを作り，テキスト系列としてそのレコードを与え，それが検索結果として出力されるならば，文法に適っていて，そうでなければ文法的誤りがあると判定できる．

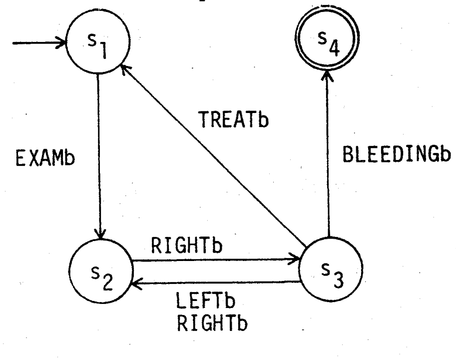

図2　不完全有限オートマトンN
（正規文法）

実際、例えば図2で与えられる文法Nに対しては，次のようなTEXTIRの質問が対応する．モードをnoresetに選び，入力区切り語としてはスペース b と登録し，キーを $A_1 = TREAT$, $A_2 = BLEEDING$, $A_3 = LEFT$, $A_4 = RIGHT$, $A_5 = EXAM$ とする．そして論理式を

$Z_1 = Z_2 \cdot A_2$, $Z_2 = Z_3 + A_4$, $Z_3 = Z_4 \cdot A_5 + [A_3 + A_4] \cdot Z_5$, $Z_4 = Z_5 \cdot A_1 + @$, $Z_5 = Z_2$ とする。そうすると $\| Z_1 \| = L(N)$ となり，Z_1 でこの文法の構文チェックができる。

さて、この定理からトリプル・ドットの機能は他の論理演算では表わせないことも分った。トリプル・ドットに関しては更に次の定理が成立する。

$R^{(k)}(\Sigma)$ ＝「通常の」TEXTIR において、トリプル・ドットを1個のキーの中で高々k個しか使用できないような質問によって定義される言語族とする。

〈定理2〉 $A(\Sigma) = R^{(0)}(\Sigma) \subsetneq R^{(1)}(\Sigma) \subsetneq \cdots \subsetneq R(\Sigma)$.

略証．$L_n = \| a \circ a \circ \cdots \circ a \|$（n 個）は $R^{(n)}(\Sigma)$ に属すが，$R^{(n-1)}(\Sigma)$ には属さないことが示される。

6. おわりに

パタン・マッチング・マシンを利用する一方向逐次サーチに基づく情報検索システム TEXTIR について概説し、その能力を考えてみた。システムは「通常」版は FORTRAN で一般の方は Coral 66 で書かれている。今後の方向として，ここで導入した区切り語の概念を一般化して、更にアクションも導入して、検索だけでなく、一般性のあるテキスト編集のためのシステムにする問題がある。

7. 参考文献

[1] Aho, A.V. and Corasick, M.J. Efficient string matching: An aid to bibliographic search, CACM 18 (1975), 333-340.

[2] 有川, 武谷, 石橋. パタン・マッチング・マシンを用いる例文検索システム, 情報処理学会論文集 (1976), 119-120.

[3] Takeya, S. and Arikawa, S. TEXTIR — A text information retrieval system using pattern matching machines, Res. Rept. Res. Inst. Fund. Inform. Sci., Kyushu Univ. 83 (1978), 1-17.

[4] Arikawa, S., Hill, A.G. and Townsend, H.R.A. An information retrieval system based on one-way sequential search, Res. Rept. MIRU, Univ. of Edinburgh (to appear).