40. VLSI 用の正規集合認識法†

安浦　寛人††

1. まえがき

正規集合は，形式言語理論の中でも最も基本的でかつ実用的な言語のクラスとして，古くからその性質やそれを取り扱うシステムの研究が数多く行われている．正規集合に関連する問題は，形式言語としての側面以外にも，**正規表現**，**有限オートマトン**，**順序機械**等種々の側面から研究されている[1),2)]．これらの研究の成果は，ソフトウェアの分野では**構文解析器** (lexical analyzer) やテキストエディタの**パターンマッチング**等に利用され，正規集合に関連する種々のアルゴリズムが開発されている．ハードウェアの分野でも，論理回路の基本回路である順序回路の構成理論が古くから研究され，実際の論理設計の中で広く利用されている．

正規集合の**認識問題**は，「与えられた系列が，指定された正規集合の要素であるか」という判定問題である．この問題は，正規集合に関連する諸問題の中でも最も基本的かつ実用的な問題の一つである．

VLSI 技術の進歩に伴い，正規集合を認識する論理回路の設計が重要になってきた．例えば，「与えられた正規表現で表わされる正規集合を認識する VLSI 回路を構成する問題」は，**シリコンコンパイラ**と呼ばれる VLSI の自動設計（回路の仕様から回路を自動的に生成する手法）の中で重要な位置を占めると考えられる[3),4)]．また，回路の大規模化に伴い，従来の論理設計手法では効率良く設計できない問題が数多く出てきているが，このような問題の中でも，正規集合の認識問題に帰着できるものについては効率の良い回路設計法が開発されている[5),6)]．

ここでは，VLSI 用の正規集合認識法として，有限オートマトンや正規表現から直接回路を構成する手法や，入力系列が並列に加えられる時の高速認識回路の構成法を紹介する．

2. 正規表現と有限オートマトン

正規集合は，形式言語の生成文法を与えることによっても定義できるが，ここでは次のような正規表現によって表わされる系列の集合として定義する．

［定義］ Σ をアルファベットとする．Σ 上の**正規表現** (regular expression) とそれが表わす Σ 上の系列の集合は，次のように帰納的に定義される．

1. ϕ は正規表現であり，空集合を表わす．
2. ε は正規表現であり，集合 $\{\varepsilon\}$ を表わす．ここに ε は空系列である．
3. $a\in\Sigma$ ならば a は正規表現であり，集合 $\{a\}$ を表わす．
4. p と q がそれぞれ集合 P, Q を表わす正規表現であるとき，$(p)+(q)$, $(p)(q)$, $(p)^*$ も正規表現で，それぞれ集合 $P\cup Q$, PQ, P^* を表わす．ここに PQ は P 中の系列と Q 中の系列の連接からなる集合を表わし，P^* は集合 $\{\varepsilon\}\cup P\cup PP\cup PPP\cup\cdots$ を表わす． □

以後，演算の順位を $*$，連接，＋として，正規表現中の括弧は省けるものは省いて記す．

正規表現で表わされる Σ 上の系列の集合を Σ 上の**正規集合** (regular set) と呼ぶ．

［定義］ **非決定性有限オートマトン** (nondeterministic finite automaton) M は 6 字組 $(\Sigma, Q, \delta, q_0, F)$ により定義される．ここに，Σ は入力アルファベット，Q は内部状態の集合，δ は状態遷移関数で $Q\times(\Sigma\cup\{\varepsilon\})$ から Q の部分集合の集合への写像，$q_0\in Q$ は初期状態，$F\subseteq Q$ は最終状態の集合である．δ が $Q\times\Sigma$ から Q への写像となっているとき，特に M を**決定性有限オートマトン** (deterministic finite automaton) と呼ぶ． □

正規集合のクラスと非決定性または決定性有限オートマトンで認識（受理）される言語（系列の集合）のクラスは一致することが知られている[1)]．また，正規

† Regular Set Recognition Methods for VLSI by Hiroto YASUURA (Department of Information Science, Faculty of Engineering Kyoto University).

†† 京都大学工学部情報工学教室

表現 R から，R の表わす正規集合を認識する非決定性または決定性有限オートマトンを得る方法もよく知られている[1),2)]．以下，状態数 $|Q|$ を r，入力アルファベットの要素数 $|\Sigma|$ を s と記す．また，状態遷移関数 δ は状態遷移グラフによって与えることもある．

3. 正規集合の直列認識法

ここでは，入力系列が逐次入力される場合の論理回路による正規集合認識法について述べる．

(1)　決定性有限オートマトンによる方法

$M=(\Sigma, Q, \delta, q_0, F)$ を正規集合 S を認識する決定性有限オートマトンとする．S を認識する回路は，**図-1** のような同期式順序回路として M から実現することができる．入力は，$\lceil \log_2 s \rceil$ ビットで符号化され，クロックに同期して逐次入力される．Q 中の各状態は，$\lceil \log_2 r \rceil$ ビットで符号化される．C_1 は状態遷移関数 δ を計算する組合せ回路で，C_2 は計算された状態 q が F に含まれるか否かを判定する組合せ回路である．

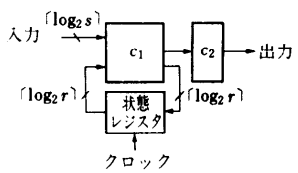
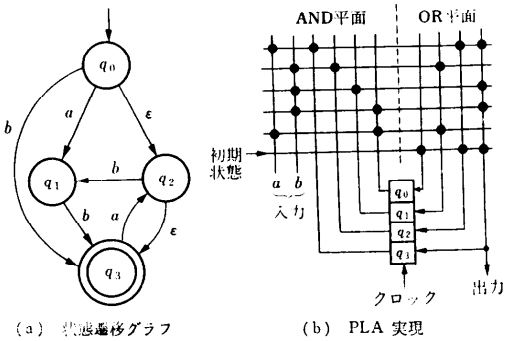


図-1　決定性有限オートマトンを実現する順序回路



(a)　状態遷移グラフ　　(b)　PLA 実現

図-2　非決定性有限オートマトン M の実現

回路の実現に，**入次数 (fan-in) 制限**のある素子を用いると，実現に必要な**素子数**の上界は $O(rs \log r/\log(rs))$ となり，**クロック周期**は $O(\log(r\cdot s))$ で実現できる．

(2)　非決定性有限オートマトンによる方法

非決定性有限オートマトン M から直接に同期式順序回路を構成することもできる．Q 中の各状態に 1 ビットずつ状態変数として割り当てることにより，図-1 と同様の形の各期式順序回路が構成できる．入次数制限のある素子で実現すると，回路の素子数は $O(rs\,2^r/r+\log s)$，クロック周期の上界は $O(\log(rs))$ となる．

一方，実現に入次数制限のない **PLA** (programmable logic array) を用いると，VLSI 上に実現したときの**面積**が $O(sr(r+\log s)$ となるような回路の構成法がある[3)]．

[例]　非決定性有限オートマトン $M=(\{a,b\}, \{q_0, q_1, q_2, q_3\}, \delta, q_0, \{q_3\})$ を考える．M の状態遷移グラフを**図-2**(a) に示す．M の PLA による実現を**図-2**(b) に示す．PLA の左半分は AND 平面，右半分は OR 平面であり，両平面の縦の線は入力および同状態に対応している．□

(3)　正規表現による方法

正規表現の基本演算に対応して回路を構成することもできる．この方法では，よく知られた McNaughton-Yamada の正規表現から非決定性有限オートマトンを得るアルゴリズム[7)]を利用する．**図-3** に，正規表現の各基本演算に対する回路の構成規則を示す．この規則を帰納的に用いて回路を得る．

[例]　正規表現 $(\varepsilon+(\varepsilon+a+b)b)(b+abb)^*$ に対する回路の構成例を**図-4** に示す．長方形の各モジュールは，**図-3**(c) の回路で，そこに記された記号と入力記号の一致をとり，その結果と前段からの状態信号の論理積をとって，フリップフロップに記憶する働きをする．入力記号とクロックは同モジュールに一斉に与えられる．

この方法では，素子数は与えられた正規表現の長さ m に比例する．Floyd と Ullman は，この方法による回路が面積 $O(m)$ で実現できることを示している[3)]．

さらに，Trickey は，面積最小の回路を得るための動的プログラミングを利用したアルゴリズムを示している[4)]．ここに述べた方法では，入力記号を全モジュールに一斉に供給する必要があるが，回路規模が大きくなると，そのための信号遅延が大きくなる問題がある．Foster と Kung は，このような信号の大域通信を行わないで同様の機能を実現するシストリック回路を提案している[8)]．

一般に，正規集合 S が与えられたとき，ここに述べた 3 種の方法によって構成した S を認識する回路

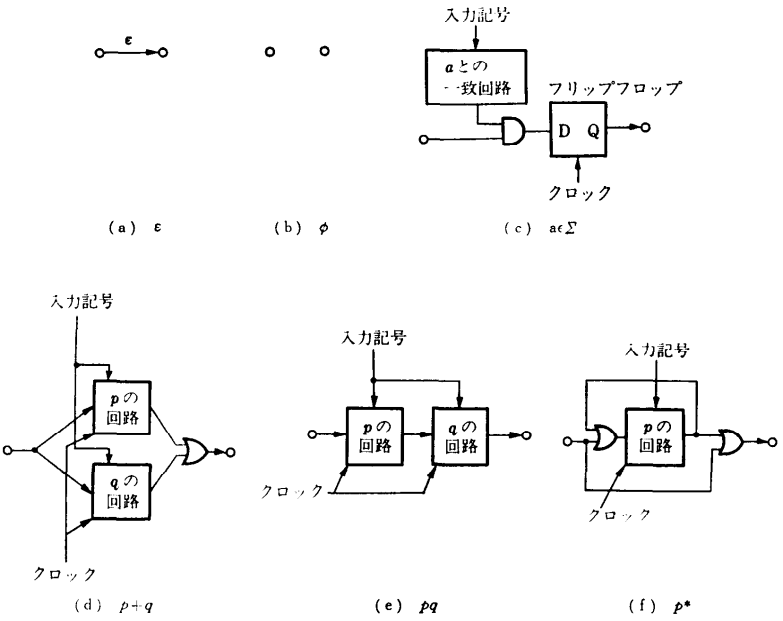


図-3 正規表現から回路への変換規則

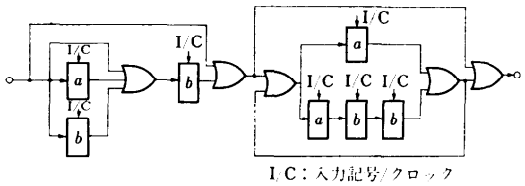


図-4 $(\varepsilon+(\varepsilon+a+b)b)(a+abb)^*$ に対する回路

の素子数，面積，クロック周期等は S によって異なる．各回路に対する素子数，面積，クロック周期の上界は，設計時にいずれの方法を選ぶかを決めるのに利用できる．

4. 正規集合の並列認識法

入力系列が一斉に与えられたとき，それをより高速に認識するための回路の構成法も知られている．ここでは，入力系列長が固定されていると仮定する．すなわち，与えられた正規集合の中の長さ n の系列を認識する回路を考える．

(1) 一次元一方向繰り返し組合せ回路

図-5(a)のような同期式順序回路の計算は，**図-5**(b)のような**一次元一方向繰り返し組合せ回路** (combinational unilateral one-dimensional iterative circuit **CUODIC**) によって模倣できる．CUODIC は組合せ回路であるから，順序回路より高速に計算が可能である．**3.** で考えた3種の順序回路から，長さ n の系列を並列に認識する CUODIC を構成できる．

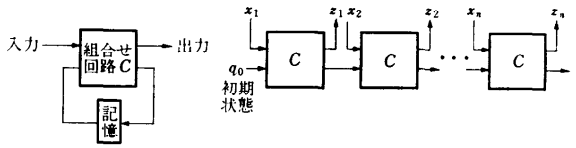
［例］ 有限オートマトン $M=(\{00, 01, 10, 11\}\{q_0, q_1\}, \delta, q_0, \{q_1\})$ を考える．ここに，

$$\delta(q_0, 00)=\delta(q_0, 11)=\delta(q_0, 10)=\delta(q_1, 10)=q_0$$

$$\delta(q_1, 00)=\delta(q_1, 11)=q(q_1, 01)=\delta(q_0, 01)=q_1$$

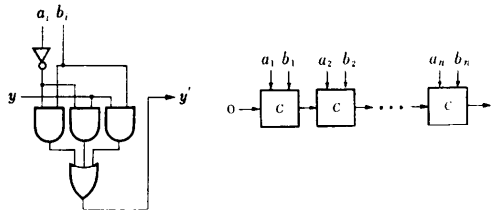
とする．これは，入力系列 $(a_1b_1)(a_2b_2)\cdots(a_nb_n)$ (但し，$a_i, b_i\in\{0,1\}$) を2つの n ビット2進数 $A=\sum_{i=1}^{n} a_i2^{i-1}$，$B=\sum_{i=1}^{n} b_i2^{i-1}$ と見て，$A<B$ となる系列を認識する比較器と考えることができる．この比較器を CUODIC で実現すると，**図-6** のようになる．素子数は $5n$，段数は $2n+1$ となる． □

CUODIC による実現では，入力系列 n に対し，素



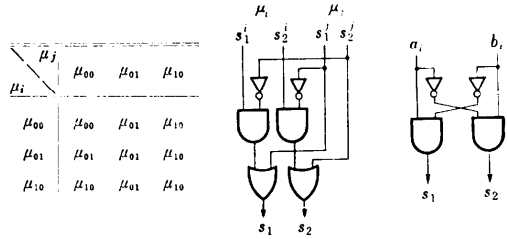
(a) 同期式順序回路 (b) 1次元1方向繰り返し組合せ回路

図-5 同期式順序回路と1次元1方向繰り返し組合せ回路

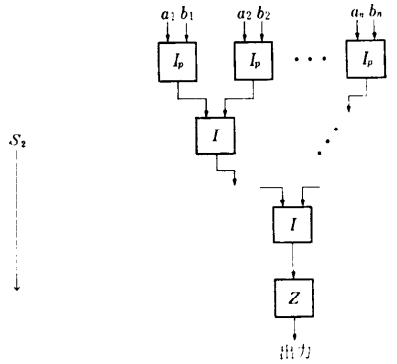


（a） 基本回路C　　（b） 比較器

図-6 CUODIC による n ビット比較器



（a） 演算表 ($\mu_i * \mu_j$)　（b） I 回路　（c） I_p 回路



（d） Z 回路　　（e） 木状回路C

図-7 $O(\log n)$ 段高速比較器

子数 $O(n)$，段数 $O(n)$ となる．

（2） $O(\log n)$ 段高速認識回路

Unger は，CUODIC と同じ計算を行う $O(\log n)$ 段の組合せ回路の構成法を与えた[5]．以下，与えられた決定性有限オートマトン M から，M が認識する正規集合中の長さ n の系列を並列に認識する段数 $O(\log n)$ の組合せ回路の構成法を説明する．

［並列高速認識回路の構成法］[5),6),9)]

（i） 決定性有限オートマトン $M=(\Sigma, Q, \delta, q_0, F)$ を考える．今，Q から Q の中への自己写像 μ について考える．自己写像 $\mu_a(a\in\Sigma)$ を $\mu_a(q)=\delta(q,a)(q\in Q)$ としてすべての Σ 中の記号に対して定義する．

（ii） Q から Q の中へのすべての自己写像の集合の中で，$\{\mu_a | a\in\Sigma\}$ から生成される自己写像の集合 G は半群をなす．すなわち，自己写像間の演算を，$\mu_i * \mu_j(q)=\mu_j(\mu_i(q))(q\in Q)$ と定義すれば，$\{\mu_a | a\in\Sigma\}$ の要素同士に有限回この演算を施して得られる自己写像の集合 G は，演算$*$について閉じており，$*$は結合則を満たすから，G は半群となる．

（iii） G中の自己写像を適当に2値符号化して，次のような回路を構成する．

I 回路：G中の2つの自己写像 μ_i, μ_j を入力し，$\mu_i * \mu_j$ を計算する回路．

I_p 回路：入力記号 $a\in\Sigma$ から自己写像 μ_a を生成する回路．

Z 回路：自己写像 μ を入力とし，初期状態q_0に対し，$\mu(q_0)\in F$ ならば1，$\mu(q_0)\notin F$ ならば0を出力する回路．

（iv） これら3種の基本回路を用いて木状の回路を構成する．まず，葉数 n，高さ $\lceil\log_2 n\rceil+1$ の2分木を構成し，その葉節点に I_p 回路を1つずつ対応させる．各内部節点には I 回路を対応させる．同回路の出力は2分木の親節点の入力と結ぶ．I_p 回路の入力に左から順に入力変数 $x_1, x_2, \cdots, x_n$ を割り当てる．根節点にあたる I 回路の出力を入力とする Z 回路を用意し，その出力を回路全体の出力とする．I 回路，I_p 回路，Z 回路は入力系列長 n に無関係に M から構成されるから，回路全体の素子数 $O(n)$，段数は $O(\log n)$ となる．□

［例］ 前の例題で考えた比較器をこの方法で構成してみる．入力記号から直接得られる自己写像は，

$\mu_{00}(q)=\delta(q,00)=\delta(q,11)=\mu_{11}(q)$

$\mu_{01}(q)=\delta(q,01)$，$\mu_{10}(q)=\delta(q,10)$

の3種である．μ_{00} は $\{q_0\to q_0, q_1\to q_1\}$，$\mu_{01}$ は $\{q_0\to q_1, q_1\to q_1\}$，$\mu_{10}$ は $\{q_0\to q_0, q_1\to q_0\}$ という写像を表わしている．$\{\mu_{00}, \mu_{01}, \mu_{10}\}$ は演算$*$について閉じているので，半群Gとなっている．Gの演算表を**図-7**（a）に示す．今，$\mu_{00}, \mu_{01}, \mu_{10}$ を2ビット (s_1, s_2) で $(0,0), (0,1), (1,0)$ とそれぞれ符号化する．I 回路，I_p 回路，Z 回路は**図-7**（b），（c），（d）のように設計される．これを**図-7**（e）のように木状に組み合わせて比較器を構成する．素子数は $10n-6$，段数は $3\lceil\log_2 n\rceil+2$ となる．□

（3） 面積・時間積最小の回路

VLSI による実現の一つの評価尺度として，回路の占めるチップ上での**面積** A と計算の**遅延時間** T の重み付けされた積 AT^α がしばしば用いられる．

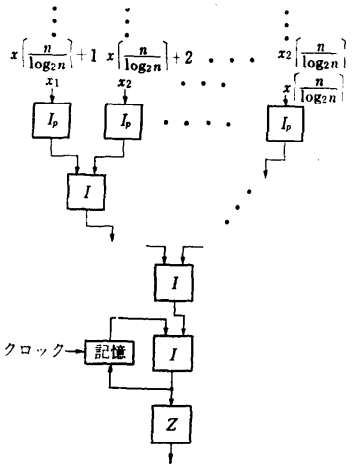


図-8 面積・時間積最小の回路

前項の並列高速認識回路の構成法を利用して，**面積・時間積** $AT^{\alpha}=O(n(\log n)^{\alpha-1})(\alpha\geqq 1)$ で与えられた正規集合 S 中の長さ n の系列を認識する回路を，次のようにして構成できる[10),11)]．まず，与えられた正規集合 S を認識する決定性有限オートマトン M に対し，前項の方法で I 回路，I_p 回路，Z 回路を構成する．次に，I_p 回路と I 回路により，入力数 $\lceil n/\log_2 n\rceil$ の木状回路を構成する．木状回路の出力を図-8 のように I 回路による順序回路の入力とする．回路全体への入力は，$\lceil n/\log_2 n\rceil$ 個ずつに分けて，クロックに同期して逐次入力する．この回路では，$A=O(n/\log n)$ となることが容易に示されるので，$AT^{\alpha}=O(n(\log n)^{\alpha-1})$ となる．これが漸近的に AT^{α} の最小値となることは文献10) に示されている．

本節で述べた3種の並列認識回路は，決定性有限オートマトンに限らず，非決定性有限オートマトンや正規表現から直接構成した順序回路から，各様の変換手順によって構成することもできる．

5. あとがき

VLSI による実現を考えた正規集合の認識法について，直列認識法および並列認識法の主なものを概説した．ここで取り上げた手法は，すべて，与えられた正規集合（正規表現）に対して，それを認識する回路を構成する方法であった．これに対して，正規集合や正規表現を使用時に指定できる万能認識回路も実用的には重要で，VLSI 化に適した方法の研究が今後の一つの課題と思われる．

参考文献

1) Hopcroft, J. E. and Ullman, J. D.: Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Massachusetts (1979).
2) Kohavi, Z.: Switching and Finite Automata Theory, McGraw-Hill, N. Y. (1978).
3) Floyd, R. W. and Ullman, J. D.: The Compilation of Regular Expressions into Integrated Circuits, J. ACM, Vol. 29, No. 3, pp. 603-622 (1982).
4) Trickey, H. W.: Good Layouts for Pattern Recognization, IEEE, Trans. Comput., Vol. C-31, No. 6, pp. 514-520 (1982).
5) Unger, S. H.: Tree Realizations of Iterative Circuits, IEEE, Trans. Comput., Vol. C-26, No. 4, pp. 365-383 (1977).
6) 安浦寛人：論理関数の複雑さの理論とその高速論理回路の構成法への応用，情報処理学会論文誌，Vol. 21, No. 4, pp. 268-278 (1980).
7) McNaughton, R. and Yamada, H.: Regular Expressions and State Graphs for Automata, IEEE. Trans. Comput. Vol. C-9, No. 1, pp. 39-47 (1960).
8) Foster, M. J. and Kung, H. T.: Recognizing Regular Languages with Programmable Building-Blocks, VLSI-81, Gray, J. P. Ed., pp. 75-84, Academic Press, N. Y. (1981).
9) Ladner, R. E. and Fischer, M. J.: Parallel Prefix Computation, J. ACM, Vol. 27, No. 4, pp. 831-838 (1980).
10) 和田，萩原，都倉：n 変数論理関数の面積複雑度，電子通信学会論文誌，Vol. J64-D, No. 8, pp. 676-681 (1981).
11) Brent, R. P. and Kung, H. T.: A Regular Layout for Parallel Adders, IEEE Trans. Comput., Vol. C-31, No. 3, pp. 260-264 (1982).

（昭和57年12月3日受付）