人 工 知 能 77－9
（1991. 7. 18）

シャフル結合プロセッサによる並列言語認識

河村明展 (+) ， 梅尾博司 (+)

(+) 大阪電気通信大学

シャフル結合プロセッサによる高速言語認識アルゴリズムを提案する．並列言語認識に関しては1次元セルオートマトンにより数多くの言語がO（n）時間で認識可能なことが知られている．本稿では，シャフル結合を持つオートマトン上で同様な認識問題を考察し，1次元セルオートマトン上でO（n）時間で認識された言語がシャフル結合プロセッサ上ではO（log n）時間で認識されることを示す．

O(log n)-Time Language Recognition Algorithms on Shuffle-Exchange Networks

Akinobu Kawamura,(+) and Hiroshi Umeo(+)

(+) Osaka Electro-Communication University

18-8 Hatu-cho, Neyagawa-shi, Osaka, 572, Japan

Abstract

Shuffle-exchange networks have wide applications in the area of parallel processing, including FFT, matrix operations, polynomial evaluation and sorting. In this paper, we propose parallel language recognition algorithms on shuffle-exchange networks. Using shuffle-exchange networks efficiently, we develop O(log n)-time recognition algorithms for some languages which have been shown to be recognized in linear time by one dimensional cellular automata.

１．はじめに

シャフル結合はその結合が複雑でデータの流れが理解しにくいが，その特異な結合をうまく利用すればセルオートマトンよりも強力な動作をさせることが可能である．Stone[1]はシャフル結合プロセッサ上でフーリエ変換,多項式評価,転置行列,ソーティングのアルゴリズムを提案した.フーリエ変換,多項式評価,転置行列の時間計算量はＯ（ｌｏｇｎ）,ソーティングの時間計算量はＯ（ｌｏｇ^{2}ｎ）である.Awerbuch and Shiloach [8] はシャフル結合プロセッサ上で時間計算量Ｏ（ｌｏｇｎ）のminimum spanning forest と connected components を求めるアルゴリズムを提案した.[2]-[7],[9]-[12]ではシャフル結合ネットワークそのものが議論されている.本稿ではセルオートマトン上でＯ（ｎ）で認識できる言語を，シャフル結合プロセッサでより速く認識することを考え，シャフル結合プロセッサの能力を調べる．

まず２．でシャフル結合の性質を示し，３．でシャフル結合上での基本操作を説明する．４．では５つの言語をそれぞれＯ（ｌｏｇｎ）で認識するアルゴリズムを提案する．

２．シャフル結合の性質

シャフル結合は図１のような結合を持つ．離れたプロセッサと結合を持つグローバル結合と，上から２つずつのプロセッサ間にあるローカルリンクを持つ．ｎ個のプロセッサはプロセッサ番号０からｎ－１までを持つ．ｎはある正数ｍに対してｎ＝2^mである．シャフル結合方式は，以下の条件を満足するプロセッサP_jとP_kを１方向リンクで結合する．

$$k=\begin{cases} 2i & (0\leqq i\leqq n/2-1\text{の時}) \\ 2i+1-n & (n/2\leqq i\leqq n-1\text{の時}) \end{cases}$$

この結合はある時点の下半分のプロセッサが次のステップで，上半分の間にうまく挿入される形となっている．

プロセッサ番号を２進数表示した時に，その２進数を左に１ビット巡回シフトした数値が次のステップに接続されているプロセッサ番号となる．このことより，次の２つの性質が得られる．

性質１　シャフルをｍ回適用するとデータ項目は元の位置にもどる．この事は定義にあるように，ｍ回巡回 シフトを行なうと元のプロセッサ番号となることによる．

性質２　ある２つのプロセッサ番号を２進数表示したときに左からｃ番目のビットだけが異なるとき，ｃ回の完全シ ャフルの後これらのデータ項目は隣接プロセッサ上に存在する．ｃ回巡回シフトをする とプロセッサ番号が最下位ビットのみ異なるからである．これはプロセッサが隣接していることを表わす．

結合が複雑で動作が理解しにくいので図２以後は時間軸を横にとった図とする．

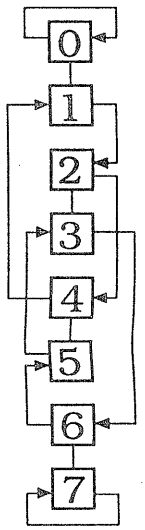

図１ シャフル結合

３．基本操作

３．では次の５つの基本操作について考える．

基本操作１　１のデータを全プロセッサにブロードキャストする．

基本操作２　各プロセッサのデータが全て１かどうか判断する．

基本操作３　全プロセッサの内，１のデータを持つプロセッサ数を計数する．

基本操作４　プロセッサのデータ列を反転する．

基本操作５　プロセッサのデータ列をｘ個下のプロセッサに巡回シフトする．

（但しｘは０≦ｘ≦ｎ－１の任意の整数）

以下の条件を満たすプロセッサを使用する．

- データの入出力が行える．
- 四則演算ができる．
- それぞれのプロセッサは０からｎ－１までのアドレスを持っている．
- 以上のプロセッサをシャフル結合にする．
- 上から２つずつのプロセッサ間にローカルリンクがあり比較，交換ができる．

・入力データはｎ個で，ある正数ｍに対してｎ＝2^mと表現される．
・一番上のプロセッサをリーダーと呼ぶことにする．
・プロセッサ番号をｉとすると，(i mod 2)=0のプロセッサを上のプロセッサ，(i mod 2)=1のプロセッサを下のプロセッサと呼ぶことにする．

３．１　基本操作１　*１のデータを全プロセッサにブロードキャストする*

（図２参照　ｎ＝８の時）

１　ローカルリンクでつながっている上下２つのプロセッサの内，上のプロセッサのデータを下のプロセッサに書き込む．

２　グローバル結合にデータを出力する．

３　１～２の動作をｌｏｇ ｎ回行う

基本操作１はリーダーのデータをブロードキャストさせる．少し変形させるだけでどのプロセッサからでもブロードキャストすることが出来る．

３．２　基本操作２　*各プロセッサのデータが全て１かどうか判断する*

（図３参照　ｎ＝８の時）

１　入力データを上から２つずつローカルリンクで比較する．上下のプロセッサのデータが１であれば，上のプロセッサにそのデータを書き込み，もし違うならば０を書き込む．

２　グローバル結合にデータを出力する．

３　１～２の動作をｌｏｇ ｎ回行なった時にリーダーが１であれば入力データは全て１である．

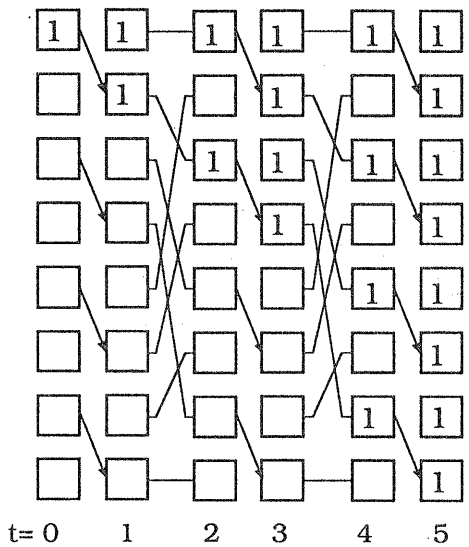

図２　基本操作１

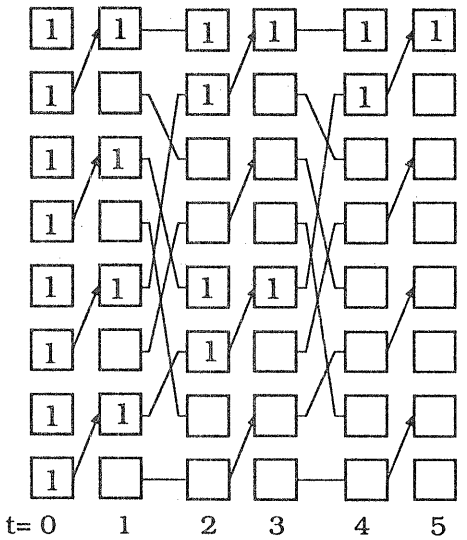

図３　基本操作２

3．3　基本操作3　*全プロセッサの内，1のデータを持つプロセッサ数を計数する*

（図4参照　n＝8の時）

1　ローカルリンクで接続されている2つのプロセッサのデータの合計を上のプロセッサに書き込む．

2　グローバル結合にデータを出力する．

3　1～2の動作をｌｏｇ ｎ回行うとリーダーに合計値が存在する．

3．4　基本操作4　*プロセッサのデータ列を反転する*

（図5参照　n＝8の時）

1　入力データを上から2つずつローカルリンクで交換する．

2　グローバル結合にデータを出力する．

3　1～2の動作をｌｏｇ ｎ回行うとデータ列は反転される．

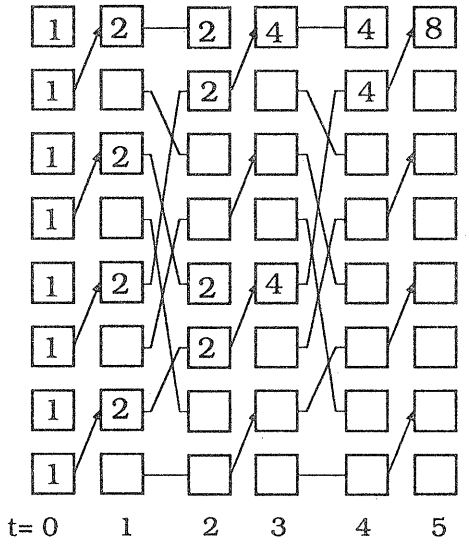

図4　基本操作3

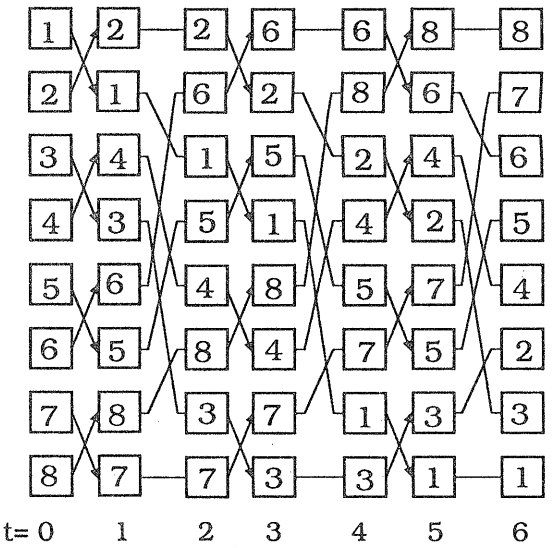

図5　基本操作4

３．５　基本操作５　*プロセッサのデータ列をｘ個下のプロセッサに巡回シフトする*

（但しｘは０≦ｘ≦ｎ－１の任意の整数）

（図６参照　ｎ＝８，ｘ＝３の時）

まず，現アドレスにｘを加算して新アドレスを求める．オーバーフローする場合はその値を新アドレ　スとする．各プロセッサは，現アドレス，新アドレス，データの３つを持つ．それら３つ全部を　まとめて転送データと呼ぶことにする．

１　新アドレスと現アドレスの最下位ビットを比較し，異なっていればローカルリンクでつながっているプロセッサの転送データと交換する．

２　グローバル結合に転送データを出力する．

３　新アドレスと現アドレスをそれぞれ左に１ビット巡回シフトする．

４　１～３の動作をｌｏｇ　ｎ回行なう

この動作は以下の理由で正しく動作する．

ローカルリンクでつながっているプロセッサはアドレスの最下位ビットのみが異なる．この事より，新アドレスと現アドレスの最下位ビットのみ異なれば，ローカルリンクでデータを交換するだけで新アドレスのプロセッサに移動できる．最下位ビット以外も異なるなら１回シャフルすればアドレスが１ビットシフトしたプロセッサに移動するので，異なるビットが最下位ビットになった時にローカルリンクでデータを交換する．この動作を全ビットに対して行えば新アドレスにデータを移動出来る．新アドレスと現アドレスの全ビットを比較すればよいのでｌｏｇ　ｎ回の動作で終了する．

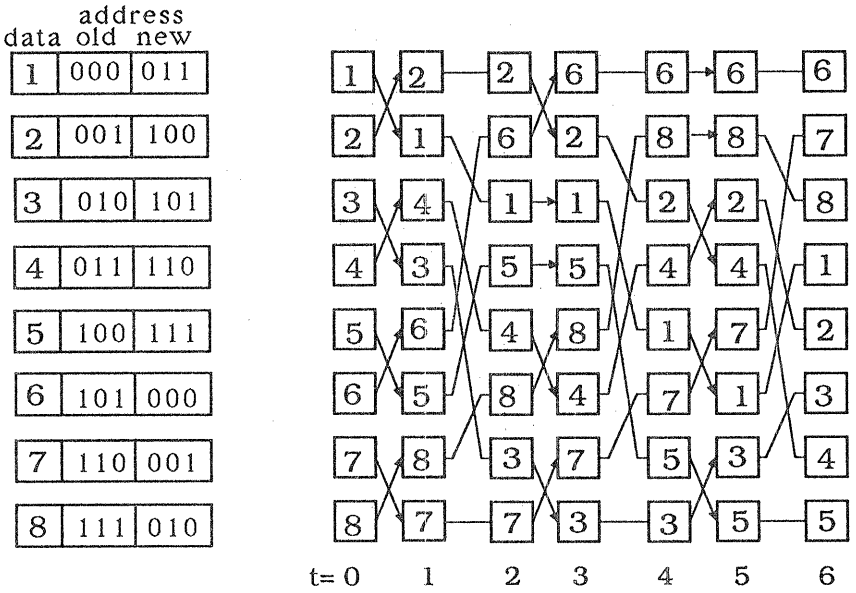

図６　基本操作５

３．６　時間計算量

基本操作の時間計算量は定数ステップの動作をｌｏｇ ｎ回行うのでＯ（ｌｏｇ ｎ）である.

３．７　プレフィクス

以上の基本操作を使ってプレフィクスを求めるアルゴリズムを示す. プレフィクスとはプロセッサ番号 ｉのプロセ ッサがプロセッサ番号０からｎ－１までのデータの合計値を求めることをいう.よってプロセッサ番号ｎ－１のプロセッサの計算結果は全データの合計値を持つ.そのアルゴリズムを以下に示す.図７参照（ｎ＝８の時）.

Ｔ＝０のデータが入力データである.

1　Ｔ＝０のデータと,そのデータを下に2^0シフトとたデータとの合計値をＴ＝１のデータとする.

2　Ｔ＝１のデータと,そのデータを下に2^1シフトとたデータとの合計値をＴ＝２のデータとする.

3　Ｔ＝２のデータと,そのデータを下に2^2シフトとたデータとの合計値をＴ＝３のデータとする.

4　ｎ＝８の場合Ｔ＝３のデータがプレフィクスとなる.

このように時刻ｔのデータとそのデータを下に2^tシフトしたデータとの合計値を時刻ｔ＋１のデータする.この動作をtが　０からｌｏｇ ｎまでｌｏｇ ｎ回行なう.シフトさせるには基本操作５を使用する.ただし基本操作５は巡回シフトを行なうので巡回するデータは０にしておく必要がある.このアルゴリズムの時間計算量は,Ｏ（ｌｏｇ ｎ）の動作をｌｏｇ ｎ回行なうのでＯ（ｌｏｇ^{2}ｎ）となる.

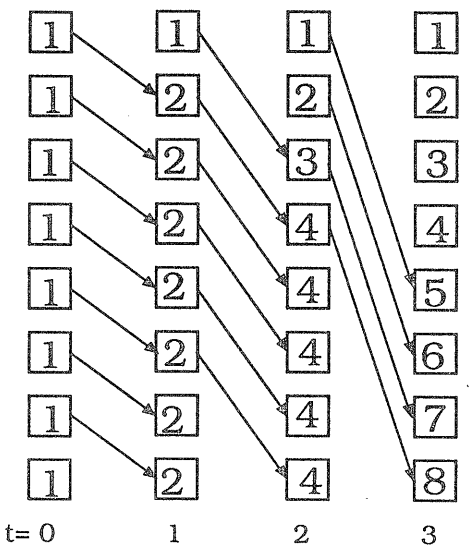

図７ プレフィクス問題

4．言語認識

4．では入力データが以下の5つの言語であるかそうでないかを認識するアルゴリズムを示す.認識するとは，入力文字列がある規則にしたがうか従わないかを判断することを言う.Σはアルファベットの集合とし，Σ上の全ての語の集合をΣ*と表す．w^r は文字列wの逆系列とする.

	例　n＝8の場合
言語1＝{w　w	w∈Σ*}	a b c d a b c d
言語2＝{a^n b^n	n≧1}	a a a a b b b b
言語3＝{w　w^r	w∈Σ*}	a b c d d c b a
言語4＝{a^n b^n c^n z	n≧1，z∈Σ*}	a a b b c c y z
言語5＝{w　w^r z	w∈Σ*，z∈Σ*}	a b c c b a y z

言語1は左から中央までの文字列が中央から右までの文字列と同じ.
言語2は左から1種類目の文字数と2種類目の文字数が同じ.
言語3は左から中央までの文字列の反転が中央から右までの文字列と同じ.
言語4は同数の文字列が3種類ある．プロセッサ数が3で割り切れないため余りの文字がある.
言語5は言語3に他の文字列が右側につながっている.
以上の5つの言語を認識するアルゴリズムを示す.ただし言語4～5の文字列zにはそれより左にある文字を含まないものとする．入力文字数をnとし，nはある整数mに対して$n=2^m$と表現される．認識結果はリーダーに出力される.

4．1　言語認識1

言語1＝{w　w　　|　w∈Σ*} を認識する．図8参照（n＝8の時）

1　グローバル結合にデータを出力する.

2　上から2つずつローカルリンクで比較し上下のプロセッサのデータが同じであれば上下のプロセッサに1を，異なるならば0を書き込む.

3　基本操作2で全てが1であるか調べる.

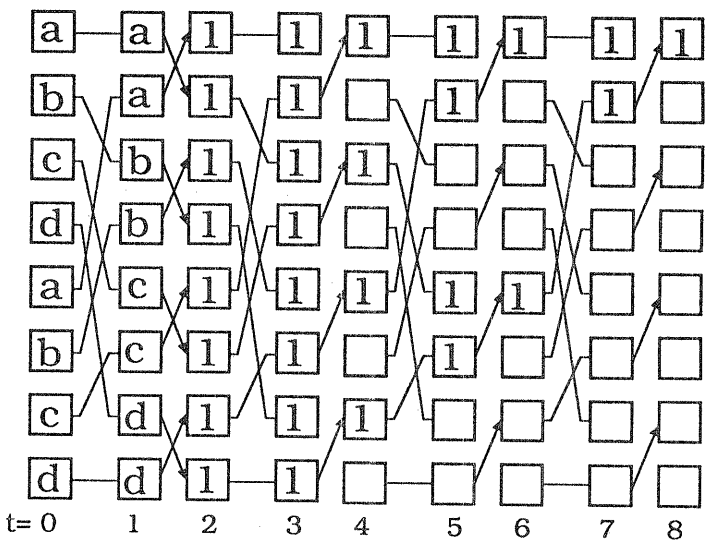

図8　言語認識1

４．２　言語認識２

言語２＝｛a^n b^n ｜ $n \geq 1$｝を認識する．図９参照（ｎ＝８の時）

このアルゴリズムではＡ，Ｂ２つのレジスタを使用する．入力データはＡレジスタに入力されるとする．

1　グローバル結合にデータを出力する．
2　上のプロセッサは下のプロセッサのデータをローカルリンクで読み込み，Ｂレジスタに書き込む．
3　グローバル結合にデータを出力する．
4　上から２つずつのプロセッサのＡレジスタ，Ｂレジスタどうしを比較し，同じでかつ０以外であれば上のプロセッサにそのデータを書き込む．もし異なるならば０を書き込む．そしてデータを出力しシャフルする．
5　４を$\log n$　－１回繰り返す．
6　リーダーが０以外ならば，言語２と認識できる．

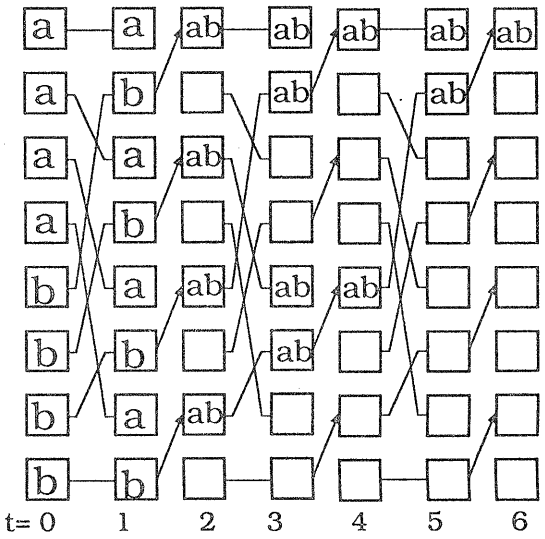

図９　言語認識２

４．３　言語認識３

言語３＝｛w　w^r　｜　$w \in \Sigma^*$｝を認識する．図１０参照（ｎ＝８の時）

この認識は，基本操作４で行なったように与えられたデータ列を反転し元のデータと同じであれば１を書き込みそれを基本操作２でチェックする．このアルゴリズムではＡ，Ｂ２つのレジスタを使用する．入力データはＡレジスタに入力されるとする．

1　Ａレジスタの値をＢレジスタに書き込む．Ａレジスタにおいて基本操作４を行なう
2　ＡレジスタとＢレジスタのデータを比較し，同じであれば１を，異なるなら０を書き込む．
3　基本操作２を行い，全てが１であれば言語３と認識できる．

４．４　言語認識４

言語４＝｛$a^n b^n c^n$　z　｜　$n \geq 1$，$z \in \Sigma^*$｝を認識する．

この認識は同数の文字列が３種類あるかを認識する．入力はＡレジスタにされるとする．

1　リーダーのＡレジスタを基本操作１でブロードキャストする．
2　受け取ったデータとＡレジスタのデータを比較し，同じであればフラグを立てる．基本操作３でフラグの数Ｆ１を計数する．
3　Ｆ１をブロードキャストする．アドレスがＦ１のプロセッサはＡレジスタを全プロセッサにブロードキャストする．
4　受け取ったデータとＡレジスタのデータを比較し，同じであればフラグを立てる．基本操作３でフラグの数Ｆ２を計数する．
5　Ｆ１＋Ｆ２をブロードキャストする．アドレスがＦ１＋Ｆ２のプロセッサはＡレジスタを全プロセッサにブロードキャストする．
6　受け取ったデータとＡレジスタのデータを比較し，同じであればフラグを立てる．基本操作３でフラグの数Ｆ３ を計数する．
7　リーダーはＦ１＝Ｆ２かつＦ２＝Ｆ３かを判別する．

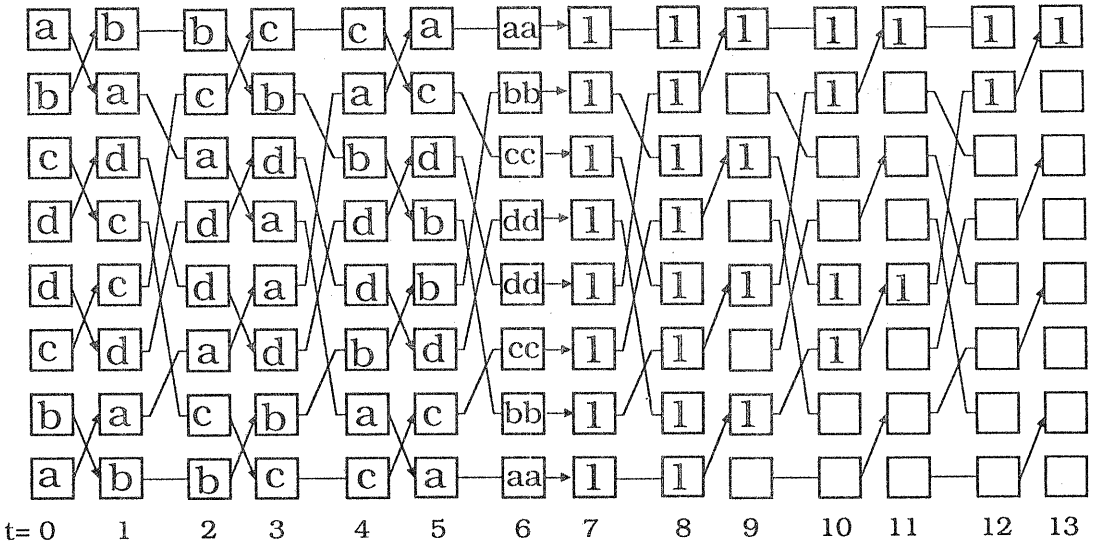

図１０ 言語認識３

４．５ 言語認識５

言語５＝｛ｗ ｗ^{r} ｚ ｜ ｗ∈Σ*, ｚ∈Σ*｝を認識する．

この認識は言語認識３の変形である．リーダーと同じデータを持つプロセッサの内，最もアドレスが大きいプロセッサを見つけ出し，そのデータを先頭とするように下にシフトさせて反転させたのが入力データと同じであるか調べる．Ａ，Ｂ２つのレジスタを使用し，入力はＡレジスタにされるとする．

１ リーダーのＡレジスタを基本操作１でブロードキャストさせる．

２ 受け取ったデータと同じデータを持つプロセッサはアドレスを出力する，異なるプロセッサは０を出力する．

３ ローカルリンクでそのデータを比較し大きい方を出力していくとｌｏｇｎ回の比較で最大のアドレスＦが求められる．

４ ｎ－Ｆ－１だけ基本操作５で下に巡回シフトし，Ｂレジスタに書き込む．基本操作４でＢレジスタを反転させる．

５ アドレスがＦ以下のプロセッサでＡとＢレジスタを比較し同じであればフラグを立て，基本操作３でフラグの数Ｆ’を数える．Ｆ’＝Ｆ＋１であれば言語５と認識できる．

４．６ 時間計算量

以上５つの認識アルゴリズムの時間計算量を考える．これら５つのアルゴリズムは定数ステップの動作とＯ（ｌｏｇｎ）で実行できる基本操作を定数回行っているだけなので時間計算量はＯ（ｌｏｇｎ）となる．

５． 結論

シャフル結合プロセッサ上で言語認識がＯ（ｌｏｇｎ）時間で出来ることが明らかになった．

参考文献

[1] Harold S.Stone;"Parallel Processing With the Perfect Shuffle",IEEE Trans.Computers,vol.C-20,pp.153-161,Feb,(1971).

[2] Tomas Lang and Harold S.Stone;"A Shuffle-Exchange Network with Simplified Control",IEEE Trans.Computers, v ol.C-25,pp.55-65,Jan,(1976).

[3] Tomas Lang;"Interconnections Between Processors and Memory Modules Using the Shuffle-Exchange Network",IEEE Trans. Computers,vol.C-25,pp.496-503,May,(1976).

[4] D.Stott Parker;"Notes on Shuffle/Exhange-Type Switching Networks",IEEE Trans.Computers,vol.C-29,pp.213-222, Mar,(1980).

[5] Chuan-Lin Wu and Tse-Yun Feng;"The Universality of the Shuffle-Exchange Network",IEEE Trans.Computers,vol. C-30,pp.324-331,May,(1981).

[6] Daniel M.Dias and Manoj Kumar;"Comments on "Interference Analysis of Shuffle/Exchange Network"",IEEE Trans. Computers, vol.C-31,pp.546-547,Jun,(1982).

[7] David Steinberg;"Invariant Properties of the Shuffle-Exchange and a Simplified Cost-Effective Version of the Omega Network",IEEE Trans.Computers,vol.C-32,pp.444-450,May,(1983).

[8] B.Awerbuch and Y.Shiloach;"New Connectivity and MSF Algolithms for Shuffle-Exchange Network and PRAM", IEEE Trans. Computers,vol.C-36,pp.1258-1263,Oct,(1987).

[9] Shing-Tsaan Huang and Satish K.Tripathi;"Self-Routing Technique in Perfect-Shuffle Networks Using Control Tags", IEEE Trans.Computers,vol.C-37,pp.251-256,Mar,(1988).

[10] Suchai Thanawastien and Pradip K.Srimani;"The Universality of a Class of Modified Single-Stage Shuffle/Exchange Networks",IEEE Trans.Computers,vol.C-37,pp.348-353,Mar,(1988).

[11] Charles.R Bisbee and Victor P.Nelson;"Failure Dependent Bandwidth in Shuffle-Exchange Networks" IEEE Trans. Computers,vol.C-37,pp.853-860,Jul,(1988).

[12] Wentailiu,Thomas H.Hilderandt and Ralph Cavin;"Hamiltonian Cycles in the Shuffle-Exchange Network",IEEE Trans. Computers,vol.C-38,pp.745-750,May,(1989).

[13] M.C.Pease;"An adaptation of the fast Fourier transform for parallel processing",J.ACM,vol.15,pp.252-262,April, (1968).

[14] S．アクル著;＂並列ソーティングアルゴリズム＂ 啓学出版 (1988).

[15] Alvy Ray Smith ;"Real-Time Language Recognition by One-Dimensional Cellular Automata", JOURNAL OF COMPUTER AND SYSTEM SCIENCES (1972).

[16] 宮島広美;"境界付きセルオートマトンの早い言語認識",電気情報通信学会論文誌D,Vol.J70-D,No5,pp.841-850,(1987).

[17] C.R.Dyer;"One-Way Bounded Cellular Automata",INFORMATION AND CONTROL 44,pp.261-281, (1980).

[18] A．サローマ著;＂計算論とオートマトン理論＂,サイエンス社,(1988).