並列プロセッサシステムにおける d-順序ベクトルに対するメモリープロセッサ結合†

牧野 武則††

並列プロセッサシステムでは，メモリ競合問題と，メモリープロセッサ結合方式が重要な課題である．この論文では，第 i 番目の要素の位置が $(d \cdot i+b) \bmod M$，ここで M はメモリモジュール数，であるようなベクトルを，d-順序ベクトルと呼び，メモリアクセスを，この d-順序ベクトルに限定することで，大規模化可能な，RSN (Rotation and Skip Network) と呼ぶ，簡素な結合方式を実現できることを示す．

基本的には，Swanson が提案した k-apart 相互結合を基にしており，メモリモジュールの数を素数とし，d-順序ベクトルではメモリ競合が生じないこと，k を法 M に関する原始根から選ぶことで，すべての d, $1 \leq d \leq M-1$, について 1-順序ベクトルに置換できることをベースにしている．

RSN は，b に対応して回転置換する部分と，d に対応して距離 d のスキップ置換を行う部分から構成される．双方とも回転置換 R_j で表わせ，$R_b T_k R_i T_k^{-1}$ と書ける．ここで T_k は $ki \bmod M \rightarrow j$ の置換を表わし，T_k^{-1} は T_k の逆置換である．また，$d=k^i \bmod M$ である．一方，回転置換 R_j は，ここで導入する巡回置換スイッチにより実現され，2入力セレクタを単位素子とすると，RSN は，$2M \log_2 M$ のオーダの素子で構成でき，LSI 化も容易になる．

メモリアクセスを d-順序ベクトルに限定したことから，RSN ではサポートできない，あるいは非効率な応用も存在する．しかし，制御が簡単でハードウェア量も少ないため，大規模並列プロセッサシステムを実現する1つの方向を示している．

1. まえがき

複数のメモリモジュールを共有する並列プロセッサシステムのアーキテクチャを構築する上で，メモリ競合に関する問題と，メモリープロセッサ相互結合方式が重要な課題であり，すでに多くの研究が，これらの問題に対しなされている[2)~4),6)]．

メモリ競合を避ける方式として，Budnic ら[1)]は，データアレイの行，または列を単一メモリサイクルでフェッチするため，メモリモジュールの数を，アレイの次数と互いに素となるよう，素数から選ぶ方法を提案している．

その後，Swanson[2)]は，N 個の要素をもつベクトルの第 i 番目の要素位置が，$p \cdot i \bmod N$ と表わせるようなベクトルを p-順序ベクトル (p-ordered vector) と呼び，k-飛び相互結合 (k-apart interconnection) をルーティングすることにより，p-順序ベクトルを1-順序ベクトルに変換する方式を提案している．

Stone[3)]は，結合セット $\{s_i\}$，ここで s_i は，要素 a を要素 $(a+s_i) \bmod N$ に移す結合を表わす，で表わされる circulator と呼ばれる相互結合を提案している．この相互結合は，要素間での回転置換を実行する．

これらの結合方式では，k-飛び相互結合や結合 s_i をルーティングすることにより置換が行われるが，与えられた結合に対して，ルーティング回数が異なりワーストケースが存在する．k-飛び相互結合の場合，要素数 M に対し，ワーストケースでは，$(M-2)$ 回のルーティングが必要となる．ルーティング回数を少なくするため，k_1-飛びと k_2-飛びの2つの相互結合をそれぞれルーティングする方式[2)]が提案されているが，要素数 257 の場合，ルーティング回数は 27 回と報告されており，ルーティングによる方式は要素数が大きくなると，その実行時間が問題となる．また，ルーティングによる結合は，レジスタ群が必要であり，ルーティング回数をカウントしなければならない等，実現上，実行時間を遅くする要因をもつ．大規模な並列プロセッサシステムを実現するためには，さらに簡素な結合方式が望まれる．

この論文では，p-順序ベクトルを拡張し，メモリアクセスが (b, d) のペア，ここで b はベクトルのオフセット，d はベクトル要素の距離，で表わせる d-順序ベクトルを定義し，このベクトルにメモリアクセスを限定することで，簡素な相互結合機構が実現できることを述べる．

はじめに，想定している並列プロセッサシステムの

† Memory-Processor Interconnection for d-Ordered Vectors in Parallel Processor Systems by TAKENORI MAKINO (Computer System Laboratory, C&C Systems Laboratories, Nippon Electric Co. Ltd.).

†† 日本電気(株) C&C システム研究所コンピュータシステム研究部

構成と，d-順序ベクトルを実際の問題に対応づけた (base, distance) で表わされるメモリアクセスパターンを設定し，プログラムコードとの関係を述べる．そして，bに対応して，要素並びを回転置換する部分と，dに対応して，スキップ置換する部分から構成される Rotation and Skip Network (RSN) と呼ぶ相互結合方式を提案する．このスキップ置換部は k-飛び相互結合と同じアイディアによるものだが，回転置換部と同様，回転置換により実現できることを示す．また，回転置換は，ここで導入する巡回置換スイッチにより構成することができ，RSN は $2 \cdot M \log_2 M$，ここで M はメモリモジュール数，のオーダで素子数で実現でき，LSI 化が容易なことを示す．最後に，d-順序ベクトルにメモリアクセスを制限したことで生じる応用プログラムに対する制限について述べる．

2. 並列プロセッサシステムとメモリ参照パターン

並列プロセッサシステムのモデルとして，Swanson[2] が想定したモデルと同様，相互結合機構により，並列メモリモジュールと並列プロセッサエレメントが結合されるモデルを考える．メモリモジュールの数は，メモリ競合を避けるため，Budnic[1] が提案したように，素数とする．このようなシステムは，すでに提案されており，バローズ社の BSP[4] や NASF 計画における FMP[5] は，それぞれ，17 メモリモジュール，521 メモリモジュールを有する並列プロセッサシステムである．

これらのシステムでは，相互結合機構として，クロスポイントスイッチ[4] や OMEGA ネットワーク[6] の変形が採用されている．しかし，メモリモジュール数，プロセッサエレメント数をNのオーダとすると，クロスポイントスイッチでは，スイッチ量は N^2 のオーダとなり，大規模化は困難である．一方，OMEGA ネットワークでは，スイッチ量が $N \log_2 N$ のオーダとなるが，スイッチノードで競合が生ずることがあり，制御も複雑になる．しかし，これらは，ランダムなメモリアクセスを許容することを目的にしている．

メモリへのアクセスがランダムではなく，d-順序ベクトルへのアクセスに限定できるならば，すでに紹介した k-飛び相互結合のアイディアを導入することができ，相互結合機構をより簡素にすることができる．

d-順序ベクトルは，Swanson[2] が与えた p-順序ベクトルの定義に，ベクトルの先頭要素がモジュール 0 だけではなく，任意のモジュールであるような一般的な場合を表わすため，ベクトルのオフセットの考慮を追加したもので次のように定義される．

[定義 1] あるベクトルの第 i 番目の要素の位置が

$$(d \cdot i + b) \bmod M,$$

ここで，$0 \leq i \leq M+1$，$0 \leq d$，$b \leq M-1$ と表現できれば，このベクトルを d-順序ベクトルと呼ぶ．

この d-順序ベクトルとプログラムコードの関係について述べる．並列計算の表現として，つぎのステートメントと考える．

do parallel for $i=1$, n

ここで，i を並列表現インデックスと呼ぶ．このステートメントは，オブジェクトレベルでは，つぎのように解釈される．

do parallel for each processor element
  do i=pe#+1, n, #pe

ここで，#pe はプロセッサエレメント数，pe# はプロセッサエレメント番号を示す．並列計算部に現われるベクトル $X(j)$ に対し，第 i 回目のループで，各プロセッサエレメントが参照するベクトル要素は，

$X((i-1) \cdot \#\text{pe}+1)$, $X((i-1) \cdot \#\text{pe}+2), \cdots,$
$X(i \cdot \#\text{pe})$

である．このベクトルが，メモリモジュールに沿って，$X(1)$ がメモリモジュール 0 に，$X(2)$ が 1 にというように格納されているとすると，1-順序ベクトルであり，第 i 回目のループでのベクトルの先頭要素は $X((i-1) \cdot \#\text{pe}+1)$ である．もし，ベクトル要素が，d モジュール飛びに格納されている場合，d-順序ベクトルであり，この d を距離と呼ぶ．ここで，各メモリサイクルでアクセスされるベクトルの先頭の要素のアドレスを (base) と呼び，ベクトルの要素間の相対アドレスを (distance) と呼び，参照されるベクトル要素を (base, distance) のペアで表わす．すなわち，第 i 番目のプロセッサエレメントが参照するベクトル要素のアドレスは，

(base)+(distance)$\cdot i$

と表わされるとする．

プログラムコードに現われるデータアレイの参照は，一般に，要素間の相対アドレスが一定なベクトルの参照とみることができる．例として，次の次元のアレイ Xを考える．

dimension $X(n_1, n_2, n_3)$

このアレイは，1 次元ベクトルに変換され，メモリシステムにストアされる．ここで，$X(1, 1, 1)$ がメモ

リモジュール0に，$X(2, 1, 1)$ がモジュール1に，$X(i_1, i_2, i_3)$ がモジュール $\{(i_1-1)+n_1\cdot(i_2-1)+n_1\cdot n_2\cdot(i_3-1)\} \bmod M$，$M$ はメモリモジュール数，にメモリモジュールに沿ってストアされるとする．

参照されるアレイの要素間の相対アドレス，すなわち，(distance) は並列表現インデックスがどの次元に現われるかにより，つぎのように決定される．

$X(i, -, -)$: distance$=1$,
$X(-, -, i)$: distance$=n_1\cdot n_2$,
$X(i, i, -)$: distance$=n_1+1$,
$X(-, a\cdot i, -)$: distance$=a\cdot n_1$,
$X(-, -, -)$: distance$=0$,
⋮

ここで，$-$は並列表現インデックスを含まないインデックスを表わす．一般に，Xの第k番目の次元のインデックス関数を $a_k\cdot i+b_k$，ここで a_k, b_k は定数，デメンジョンをr次元とすると，

$$\text{distance}=\sum_{k=1}^{r} a_k \prod_{j=0}^{k-1} n_j, \quad \text{ただし } n_0=1.$$

参照される要素が一定の相対アドレスをもつには，インデックス関数が，並列表現インデックスの1次関数であることが要求される．このことは改めて後述する．

並列表現インデックスに関して，インデックス関数が1次ならば，変数のアクセスは，(base, distance) のペアで表わすことができ，プロセッサエレメントの番号を i, $i=0, 1, 2, \cdots$, とすると，プロセッサエレメント i が参照する要素のアドレスは

(base)+(distance)$\cdot i$

と表わせる．なお，distance が0の場合は，(base) で指定される要素の各プロセッサエレメントに対するブロードキャスティングを表わすが，本稿では議論しない．

つぎに，(base, distance) のペアで表わされる参照パターンは，メモリモジュール数 M が distance と互いに素，(M, distance)$=1$ の場合，メモリ競合が起らないことを，整数論[7]の完全剰余系の定義を使用して説明する．

[定義 2] $x\equiv y \pmod M$ ならば，y は M を法とするxの剰余という．集合 $x_1, x_2, \cdots, x_M$ は，すべてのyについて，$y\equiv x_j \pmod M$ となるような x_j がただひとつしか存在しなければ，完全剰余系という．

完全剰余系の1つの例は，集合 $0, 1, 2, \cdots, M-1$ であり，この集合はプロセッサエレメント番号に対応させることができる．

[定理 1] (a, M)$=1$ ならば，集合 $\{i \mid i=0, 1, \cdots, M-1\}$ が完全剰余系ならば，集合 $\{a\cdot i+b\}$ も，法 M に関して完全剰余系である．

証明：整理論で既知数，省略する□

[系] (base, distance) のペアで表わされるアクセスパターンは，distance とメモリモジュール数が互いに素であれば，競合は起らない．

(base, distance) のペアで表わされる各要素が格納されているメモリモジュールは，

$(b+d\cdot i) \bmod M$

と表わされる．ここで，

$b=$(base) $\bmod M$,
$d=$(distance) $\bmod M$.

このベクトルをフェッチする場合，$(b+d\cdot i) \bmod M$ で表わされるメモリモジュールに入っている要素をプロセッサエレメント i に送ればよい．このため，まず (base) を合わせるため，b ポジション左に回転置換を行い，つぎに，距離dだけ離れたベクトル要素を距離1のベクトルに置換するスキップ置換を行えばよい．ここで，上述の回転置換とスキップ置換に対する定義を与えておく．

[定義 3] 回転置換．要素並び $0, 1, 2, \cdots, M-1$ を左に i ポジション回転させる置換，すなわち，すべての j, $0\le j\le M-1$, についての置換

$(i+j) \bmod M \to j$

を回転置換 R_i という．

[定義 4] スキップ置換．ベクトル要素間の距離がdであるベクトルの距離1のベクトルへの置換を距離dのスキップ置換，S_d と呼ぶ．この置換 S_d は，すべての j, $0\le j\le M-1$, について

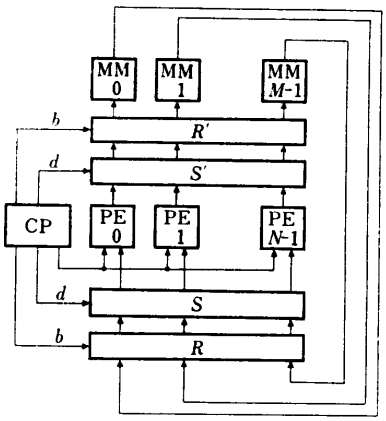
$(d\cdot j) \bmod M \to j$

を実行する．

これらの定義から，b ポジション回転置換を R_b と表わし，距離dのスキップ置換を S_d と表わすと，フェッチにおける置換は，$R_b\cdot S_d$ と表わされる．

逆に，プロセッサエレメントからのデータのストアやアドレス，コマンドの転送は，まず 1-順序ベクトルを d-順序ベクトルに置換し，そして，b ポジション右回転置換を行うことで達成できる．置換 R_b, S_d の逆置換を R_b^{-1}, S_d^{-1} と表わすと，ストアにおける置換は，$S_d^{-1}\cdot R_b^{-1}$ と表わされる．

図1に，RSN を導入した並列プロセッサシステムを示す．制御プロセッサ CP は，プログラムコードか



PE: Processor element　　R, R': Rotation permutation
MM: Memory module　　S, S': Skip permutation
CP: Control processor

図 1 並列プロセッサの構成
Fig. 1 Parallel processor configuration.

ら得られる (base, distance) のペアから (b, d) を計算する．R, R' は，それぞれ回転置換 R_b, R_b^{-1} を行う回転部，S, S' は，スキップ置換 S_d, S_d^{-1} を行うスキップ部を示す．各プロセッサエレメントは自身のエレメント番号 i にもとづき，

$\lfloor$(base+distance$\cdot i$)$/M\rfloor$

を計算し，メモリモジュール内アドレスを得る．ここで $\lfloor x\rfloor$ は x より小さいか等しい最大の整数を表わす．一方，CP は，R', S' に，b, d を送り，プロセッサからメモリへのパスを確立する．そして，各プロセッサは，計算したアドレス，コマンド，ストアであればデータをメモリに送る．コマンドがフェッチならば，メモリサイクルの終りを待って，CP は，R, S に，b, d を送り，メモリからプロセッサへのパスを確立する．

このように，メモリに対するアクセスは，各プロセッサエレメントが完全に同期して行われ，SIMD 動作となる．

3. Rotation and Skip Network (RSN)

RSN は，回転置換を行う回転部と，スキップ置換を行うスキップ部に分けられる．ここでは，回転置換を行うシフタが得られたものとして議論する．

距離 d のスキップ置換 S_d は，Swanson が述べているように，k-飛び相互結合をルーティングすることで達成される．この結合はスキップ置換 S_k であり，j 回のルーティングで，置換 $(S_k)^j$ が行われる．以降，$(S_k)^j$ を S_k^j と書く．

[定理 2] スキップ置換 S_k^j は置換 $S_{k \bmod M}^j$ に等しい．

証明：置換 S_k は，$i=0, 1, 2, \cdots, M-1$ について，$(k\cdot i \bmod M)$ 番目の入力を，出力 i に移す置換であり，置換 S_k^2 は，$(k^2\cdot i \bmod M)$ を $(k\cdot i \bmod M)$ に移し，さらに，i に移す．したがって，置換 S_k^j は，$(k^j\cdot i \bmod M)$ を i に移す置換であり，置換 $S_{k \bmod M}^j$ である□

このように，距離 d のスキップ置換を行うには，$d=k^j \bmod M$ となる j を決め，j 回ルーティングを行えばよい．しかし，$1\le d\le M-1$ であり，ワーストケースで $(M-2)$ 回のルーティングが必要となる．また，すべての d について j が与えられるには，k が巡回群 G_M の generator であることを Swanson[2] が示した．ここでは，後の説明のため，べき剰余系[7] で説明する．

[定義 5] k が法 M に関して指数 $\phi(M)$ に属するとすると，k は M 法に関して原始根という．ここで，ϕ はオイラー関数であり，M が素数の場合，$\phi(M)=M-1$ である．

[定理 3] M を素数とし，k を M 法に関して原始根とすれば，集合 $\{k^i\}$ は，$1\le i\le M-1$ について，異なった i に対し，合同ではない．

証明：整数論で既知ゆえ，省略する□

[系] k が法 M に関して原始根ならば，集合 $\{k^i\}$ は，法 M に関して完全剰余系であり，与えられた d に対して，$d=k^i \bmod M$ となる i がかならず存在する．

[定理 4] スキップ置換 S_d は，要素 $1, 2, \cdots, M-1$ について，

$$S_d=T_k R_i T_k^{-1}$$

と表わせる．ここで，置換 T_k は，入力要素 $k^i \bmod M$ を要素 j, $0\le j\le M-1$, に移す置換であり，mod M の記述を省略して，

$$T_k=\begin{pmatrix} k^0 & k^1 \cdots k^{M-2} \\ 1 & 2 \cdots M-1 \end{pmatrix}.$$

置換 T_k^{-1} は，置換 T_k の逆置換である．また，i は，$d=k^i \bmod M$ から与えられる．

証明：置換 S_k は要素 $k\cdot i \bmod M$ を i に移す置換であり，

$$S_k=\begin{pmatrix} 0 & k \cdots k\cdot i \bmod M \cdots k\cdot(M-1) \bmod M \\ 0 & 1 \cdots i \quad\quad \cdots \quad M-1 \end{pmatrix}.$$

この置換では，要素 0 は，置換に関係しないので，要素 $1, 2, \cdots, M-1$ について考える．ここで，つぎのマ

トリックスUを導入する．このマトリックス要素 u_{ij} は，距離 i のスキップ置換を行ったとき，入力 u_{ij} が出力 j に移されることを示す．各要素に係る mod M を省略して，

$$U=\begin{bmatrix} 1 & 2 & \cdots & j & \cdots & M-1 \\ 2 & 4 & \cdots & 2\cdot j & \cdots & 2\cdot(M-1) \\ \vdots & \vdots & & \vdots & & \vdots \\ M-1 & 2\cdot(M-1) & \cdots & j\cdot(M-1) & \cdots & (M-1)(M-1) \end{bmatrix}.$$

このマトリックスの行と列を適当に入れ換えて，つぎのマトリックス U^*，ここで各要素に係る mod M の記述を省略，を得る．

$$U^*=\begin{bmatrix} 1 & k & k^2 \cdots & k^{M-2} \\ k & k^2 & k^3 \cdots & 1 \\ k^2 & k^3 & k^4 \cdots & k \\ \vdots & \vdots & \vdots & \vdots \\ k^{M-2} & 1 & k & \cdots & k^{M-3} \end{bmatrix}.$$

マトリックス U^* は，U における $u_{ij}=i\cdot j \bmod M$ という関係を保持している．すなわち，$u_{ij}=i\cdot j \bmod M$ に，$i=k^{r_1} \bmod M$，$j=k^{r_2} \bmod M$ を代入して，

$$u_{ij}=k^{r_1}\cdot k^{r_2} \bmod M=k^{r_1+r_2} \bmod M=u^*_{r_1 r_2}.$$

k が法 M に関して原始根だから，集合 $1, k, \cdots, k^{M-2}$ は法 M に関して完全剰余系である．マトリックス U^* の各列のベクトルの要素列について，ベクトル $(j-1)$ の要素列を右に1ポジション回転置換した結果がベクトル j の要素列になっている．したがって，まず，入力要素並びを置換 T_k により出力並びが $1, k, \cdots, k^{M-2} \bmod M$ となるように入れ換え，$d=k^i \bmod M$ を満足する i により回転置換 R_i を行い，最後に置換 T_k の逆置換 T_k^{-1} を施せば，置換 S_d が実現される□

［系］ 定理3と4から，(b, d) のペアで表わされる結合は，フェッチの場合，つぎの置換

$$R_b T_k R_i T_k^{-1},$$

ここで，$d=k^i \bmod M$．置換 T_k は，d や b に依存しない固定の置換であり，RSN は d と b による回転置換を行うシフタで構成される．なお，置換 R_i, T_k, T_k^{-1} は要素0には関係しない．

一方，ストアの場合，置換 $S_d^{-1}R_b^{-1}$ である．

［定理 5］ 要素数 M について，

$$S_d^{-1}=S_{d'},$$

ここで，$d\cdot d' \bmod M=1$ である．

証明：$S_{d'}$ は要素 $d\cdot i \bmod M$ を i に移す置換であり，$d\cdot d\cdot i \bmod M$ を $d\cdot i \bmod M$ に移す．$d\cdot d' \bmod M=1$ ゆえ，i を $d\cdot i \bmod M$ に移す置換であり．すなわち，S_d^{-1}である□

また，回転置換の性質から $R_b^{-1}=R_{M-b}$ であり，置換，$R_j T_k R_i T_k^{-1}$ に対して，$j=M-b$ とし，$d'=k^i \bmod M$ を満足する i を計算することで，ストアの場合の置換が行える．

4. 回転置換の構成

大規模な回転置換を行うシフタを構成する方法として，巡回置換スイッチと呼ぶ，制御信号 c に従い，ポジション左に回転シフトするスイッチによる実現を考える．制御信号 c が1ならば，入力要素並びを左に1ポジション回転シフトし，c が0ならば，入力要素並びをそのまま出力する．ここで，$R_1=R$ とおき，このスイッチによる置換を R^c と表わす．

［定理 6］ $(i, M)=1$ ならば，回転置換 R_i は，1つの巡回置換をなし，

$$R_i=S_i R S_i^{-1},$$

と表わせる．ここで，S_i は距離 i のスキップ置換である．

証明：回転置換 R_i は，置換ベクトル表示で表わすと，

$$R_i=\begin{pmatrix} i & i+1 & \cdots & i-1 \\ 0 & 1 & & \cdots M-1 \end{pmatrix}.$$

ベクトル要素を入れ換え，

$$R_i'=\begin{pmatrix} i & 2i & 3i \cdots & 0 \\ 0 & i & 2i \cdots & (M-1)i \end{pmatrix}.$$

R_i' の各要素は mod M がとられるが記述上省略する．集合 $0, i, 2i, \cdots, (M-1)i$ は定義2から完全剰余系だから，1つの巡回置換をなす．また，R' は要素並び $0, i, 2i, \cdots$ を左に1ポジション回転置換する R_1 であり，したがって，$S_i R S_i^{-1}$ と表わせる□

［定理 7］ $(i, M)=1$ の場合，$i=\sum_{j=0}^{t-1} c_j 2^j$ と表わすと，ここで t は $0\le i\le M-1$ ゆえ，$(M-1)$ を2進表現したときの桁数を表わす，回転置換 R_i は

$$R_i=\left(\prod_{j=0}^{t-2} R^{c_j} S_2\right) R^{c_{t-1}} S_{2^{t-1}}^{-1},$$

と表わせる．

証明：i を2進表現にし，R_i を展開すると，

$$R_i=R_1^{c_0} R_2^{c_1} R_{2^2}^{c_2} \cdots R_{2^{t-1}}^{c_{t-1}}.$$

定理6から

$$R_i=\prod_{j=0}^{t-1} (S_{2^j} R^{c_j} S_{2^j}^{-1}).$$

ここで，

$$S_{2j}^{-1}S_{2j+1}=S_{2j}^{-1}S_{2j}S_2=S_2,$$

ゆえ，

$$R_i=\prod_{j=0}^{t-2}(R^{e_j}S_2)R^{e_{t-1}}S_{2^{t-1}}^{-1}.$$

と表わせる□

したがって，M が2の累乗と互いに素であれば，巡回置換スイッチ R^e を直列に並べ，その間をスキップ置換 S_2 に従って結合し，最後に置換 $S_{2^{t-1}}^{-1}$ に対応する結合を行うことで，回転置換を実現できる．この置換に必要なスイッチの段数は，$0\leq i\leq M-1$ から $t=\lceil\log_2 M\rceil$，ここで「x」は x に等しいか大きい最小の整数を表わす．例として，$M=7$ 場合の巡回置換スイッチ R^e による回転置換の構成を**図2**に示す．

RSN の回転部は，M が素数として選ばれるため，定理7による構成をとるが，スモップ部は，定理4から，(素数-1)要素の回転置換を行う必要がある．つぎに，$N=M-1$ とし，$(N,2)\neq 1$ の場合について，N 要素の回転置換を行う方法について述べる．

[定理 8] $(i,N)\neq 1$ の場合，回転置換 R_i は，g 個の長さ h の巡回置換の積で表わされる．ここで，

$$h=[i,N]/i,\quad g=N/h,$$

[ ] は最小公倍数を示す．

証明：回転置換 R_i のベクトル要素を入れ換え，

$$R_i'=\begin{pmatrix}i & 2i & 3i\cdots & 0\\ 0 & i & 2i\cdots & (N-1)i\end{pmatrix}.$$

ここで各要素に係る mod N は略している．$(i,N)\neq 1$ ゆえ

$$h\cdot i\equiv 0\ (\mathrm{mod}\ N)$$

となるようなNより小さい最小の正の整数 h が存在する．ここで，$[i,N]\equiv 0\ (\mathrm{mod}\ N)$ ゆえ，

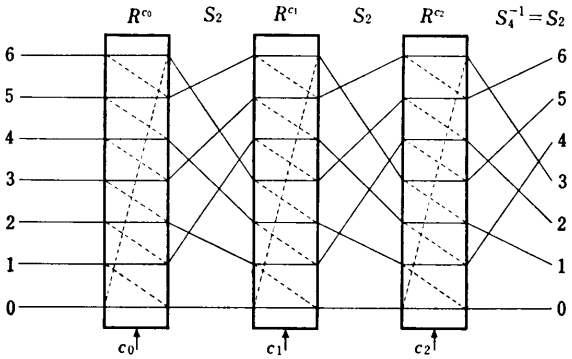


図 2 $M=7$ に対する巡回置換スイッチ R^{c_j} を用いた回転置換

Fig. 2 Rotation permutation using cyclic permutation switches for $M=7$

$$h=[i,N]/i.$$

また，任意の要素 j について，$h\cdot i\equiv 0\ (\mathrm{mod}\ N)$ ゆえ，

$$j+h\cdot i\equiv j\ (\mathrm{mod}\ N)$$

である．したがって，要素 j から始めると，巡回置換は $(j+i)\ \mathrm{mod}N\rightarrow j$, $(j+2\cdot i)\ \mathrm{mod}\ N\rightarrow(j+i)\ \mathrm{mod}\ N$, $\cdots$, $(j+(h-1)\cdot j)\ \mathrm{mod}\ N\rightarrow j$ と表わせ，長さ h の巡回置換となる．他の巡回置換に含まれる要素 j' は，要素 j を含む巡回置換に含まれることはない．もしあるとすれば，

$$j'\equiv j+k'i\ (\mathrm{mod}\ N).\quad 0\leq k'\leq h,$$

となり，矛盾となる．したがって，$g=N/h$ とすれば回転置換 R_i は，g 個の長さ h の要素を共有しない巡回置換の積で表わされる□

例として，$N=6$ の場合の R_4 について説明する．この場合，$h=[6,\ 4]/4=3$, $g=6/3=2$ である．すなわち，

$$R_4=\left(\begin{array}{ccc|ccc}0 & 4 & 2 & 1 & 5 & 3\\ 2 & 0 & 4 & 3 & 1 & 5\end{array}\right),$$

であり，R_4 は2つの巡回置換 (0 2 4)，(1 3 5) の積である．ここで，R_i の巡回置換表現は，巡回置換スイッチが左へ1ポジション回転置換を行うため，逆巡回置換表現の方が扱いやすい．そこで，R_4 についても，逆巡回置換 (0 4 2)，(1 5 3) と考えることにする．

[定義 6] g 個の長さ h の巡回置換で表わされる回転置換 R_i を，

$$R_i=\prod_{j=1}^{g}R_{i,j},\ \text{または},\ R_i=R_{i,1}\circ R_{i,2}\circ\cdots\circ R_{i,g}$$

と表わす．$R_{i,j}$ は第 j 番目の長さ h の巡回置換である．

Γ は置換の積を表わすが，要素を共有しない置換の積であり，一般の積 Π と区別することにする．

[定理 9] $(2,N)\neq 1$ の場合，$i=\sum_{j=0}^{t-1}e_j2^j$ と書くと，

$$R_i=R^{e_0}\prod_{j=0}^{t-1}\Gamma_{j=1}^{g}(S_{2^j,k}R_{1,k}^{e_j}S_{2^j,k}^{-1}).$$

ここで，g_i は第 j 段目の回転置換における巡回置換の個数であり，$S_{2^j,k}$ は第 j 段目における第 k 番目の巡回置換に対する距離 2^j のスキップ置換である．また，$R_{1,k}^{e_j}$ は，制御信号が e_j である長さ N/g_i の巡回置換スイッチである．

証明：回転置換 R_i を展開すると，

$$R_i=R_1^{e_0}R_2^{e_1}R_4^{e_2}\cdots R_{2^{t-1}}^{e_{t-1}}.$$

定義6から，R_{2^j} の巡回置換の数を g_i とすると，

$$R_2{}^j=\prod_{j=1}^{g} R_2{}^j{}_{,k}$$

であり，$R_2{}^j{}_{,k}$ は巡回置換ゆえ

$$R_2{}^j{}_{,k}=S_2{}^j{}_{,k}R_{1,k}S^{-1}_{2^j,k},$$

したがって，

$$R^{e_j}_{2^j}=\prod_{j=j}^{g} S_2{}^j{}_{,k}R^{e_j}_{1,k}S^{-1}_{2^j,k}$$

を，R_i に代入すれば，定理が得られる□

[定理 10] $N=2^kL$，ここで $(L,2)=1$，とすれば，

$g_j=2^{j-1}$，if $0\leq j\leq k$,

$g_j=2^k$，if $j>k$.

証明：定理 8 より直接得られる□

例として，$N=126$ の場合を考える．$N=2\cdot 63$ ゆえ，第 1 段目は長さ 126 の巡回置換，2 段目以降は長さ 63 の 2 つと巡回置換により表わされる．ところが，$N=256$ の場合，$N=2^8$ であり，第 $(i+1)$ 段目は $2^8/2^i$ 個の巡回置換に分けられ，最終段は長さ 2 の 128 個の巡回置換に分けられる．この長さ 2 の巡回置換は，permutation cell[4] に対応する．巡回置換スイッチの LSI 化を考えると，入出力ピンの制限があるため，長さが異なった巡回置換を少なくし，LSI の種類が少なくなるよう，定理 10 における Lをできるだけ大きくするようなNを選ぶ方が有利であろう．

定理 9 における R_i の表現を簡単にするため，つぎの表現を使用する．

$$\prod_{j=1}^{g} S_{i,k}R_{1,k}S^{-1}_{i,k}=\prod_{j=1}^{g} S_{i,k}\prod_{j=1}^{g} R_{1,k}\prod_{j=1}^{g} S^{-1}_{i,k}.$$

例として，$N=6$ の場合について説明する．定理 9 から，上述の表現で表わすと，$g_0=1$，$g_1=2$，$g_2=2$ ゆえ，

$$R_i=R^{e_0}\{(S_{2,1}\circ S_{2,2})(R^{e_0}_{1,1}\circ R^{e_1}_{1,2})(S^{-1}_{2,1}\circ S^{-1}_{2,2})\}$$
$$\{(S_{4,1}\circ S_{4,2})(R^{e_1}_{1,1}\circ R^{e_2}_{1,2})(S^{-1}_{4,1}\circ S^{-1}_{4,2})\}.$$

ここで，第 2 段目，e_1 に対する置換において，2 つの巡回置換は逆巡回置換表現で (0 2 4)，(1 3 5) であり，スキップ置換 $S_{2,k}$ は，それぞれ

$$S_{2,1}=\begin{pmatrix}0&2&4\\0&1&2\end{pmatrix},\quad S_{2,2}=\begin{pmatrix}1&3&5\\3&4&5\end{pmatrix}.$$

同様にして，第 3 段目，e_2に対する置換において，2 つの巡回置換は，逆巡回置換表現すると，(0 4 2)，(1 5 3) であり，

$$S_{4,1}=\begin{pmatrix}0&4&2\\0&1&2\end{pmatrix},\quad S_{4,2}=\begin{pmatrix}1&5&3\\3&4&5\end{pmatrix}.$$

ここで，便宜上，$S_{2,1}\circ S_{2,2}=S^*_2$，$S_{4,1}\circ S_{4,2}=S^*_4$ と

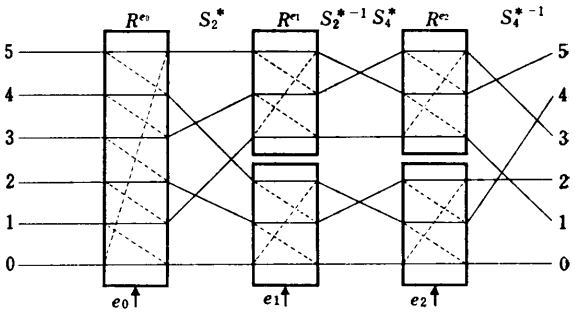


図 3 $N=6$ に対する巡回置換スイッチ R^{e_i} を用いた回転置換

Fig. 3 Rotation permutation using cyclic permutation switches for $N=6$.

書く．

$$S^*_2=\begin{pmatrix}0&2&4&1&3&5\\0&1&2&3&4&5\end{pmatrix},$$

$$S^*_4=\begin{pmatrix}0&4&2&1&5&3\\0&1&2&3&4&5\end{pmatrix}.$$

したがって，

$$S^{*-1}_2S_4=\begin{pmatrix}0&1&2&3&4&5\\0&2&1&3&5&4\end{pmatrix}.$$

図 3 に，$N=6$ の場合の巡回置換スイッチを用いて構成した回転置換，シフタを示す．図に示すように，g_i に対応して，1 段目は 1 つの巡回置換スイッチ，2 段目，3 段目は 2 つの長さ 3 の巡回置換スイッチを並べ，上述の置換 S^*_2，$S^{*-1}_2S^*_4$，S^{*-1}_4 で結合し，実現される．

このように，任意の回転置換は，巡回置換スイッチを用いて実現できる．M 要素の回転置換は，$\lceil\log_2M\rceil$ 段で構成され，2 入力セレクタを単位素子とすると，$M\cdot\lceil\log_2M\rceil$ 素子で実現できる．したがって，RSN は，$(M\cdot\lceil\log_2M\rceil+(M-1)\cdot\lceil\log_2(M-1)\rceil)$ 素子で構成される．大きな M に対しては，$2M\log_2M$ 素子のオーダである．

図 4 に，巡回置換スイッチを用いた RSN の構成を，$M=7$ の場合について示す．この結合はメモリモジュールからプロセッサエレメントへの RSN を示しており，置換 $R_bT_kR_iT^{-1}_k$ を実行する．ここで，$k=3$，$b=\sum_{j=0}^{t-1}c_j2^i$，そして，$d=k^i \bmod M$，$i=\sum_{j=0}^{t'-1}e_j2^j$ であり，$t=\lfloor\log_2(M-1)\rfloor$，$t'=\lfloor\log_2(M-2)\rfloor$ である．d と i の対応は，あらかじめ，ROM に登録しておくと有用であろう．また，置換 T_k は，$k=3$，$M=7$ について，

$$T_k=\begin{pmatrix}k&k^2 \bmod M\cdots k^j \bmod M\cdots&1\\1&2\quad\cdots\quad j\quad\cdots&M-1\end{pmatrix},$$

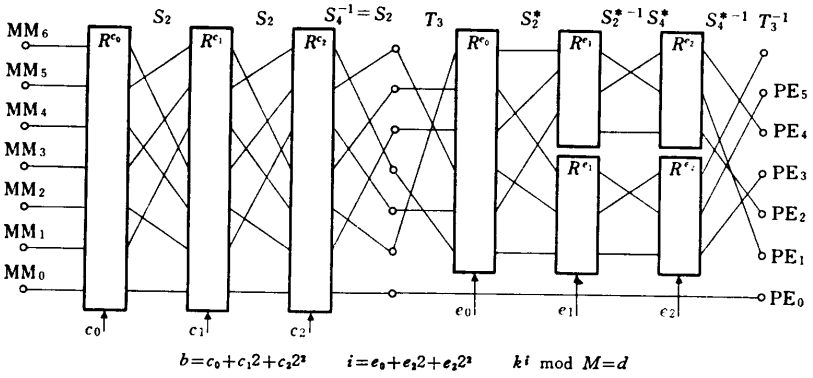


図 4 7メモリモジュールと6プロセッサエレメントに対する RSN の構成
Fig. 4 Rotation and skip network configuration.

$$T_3=\begin{pmatrix}3 & 2 & 6 & 4 & 5 & 1\\ 1 & 2 & 3 & 4 & 5 & 6\end{pmatrix},$$

$$S_4^{*-1}T_3^{-1}=\begin{pmatrix}5 & 4 & 1 & 6 & 2 & 3\\ 1 & 2 & 3 & 4 & 5 & 6\end{pmatrix}.$$

ここで，スキップ部の回転置換の対象となる要素は，0, 1, …, 5 ではなく 1, 2, …, 6 であり，図3の説明の要素番号に1を加えた番号になっている．

以上のように，RSN は巡回置換スイッチを用いて構成される．このハードウェア量は，現在，ハードウェア量が少ないといわれている OMEGA ネットワークと比較すると次のようになる．$M\times M$ の OMEGA ネットワークは，$(M\log_2 M)/2$ 個の permutation cell で構成され，各 permutation cell は2つの2入力セレクタで構成されるため，2入力セレクタを単位素子とすると，$M\log_2 M$ 素子で実現される．したがって，RSN は，OMEGA ネットワークの約2倍の素子を必要とする．しかし，このハードウェア量は，ある入出力ピンの制限をもつ LSI 化を考慮すると，RSN では，1つの巡回置換スイッチについて1本の制御信号のみだが，OMEGA ネットワークでは，それぞれの cell に1本ずつの制御信号が必要であり，また，各 cell のスイッチのためのロジックがそれぞれの cell に必要である．したがって，ネットワーク全体での LSI の量は，むしろ RSN の方が少なくなるか，同程度と考えてよい．

この2つのネットワークは，第2章で議論したように，機能的には大きく異なっている．したがって，ネットワークの選択は，対象とする応用領域におけるプログラムの特性をみきわめた上でなされるべきである．

5. 応用に対する制限

RSN は，その構造から (base, distance) のペアで表わされるアクセスパターン以外のアクセスをサポートできない．応用プログラムに対する制限は，このアクセスパターンの制限によって生じる．この制限は，多くの基本的な数値計算アルゴリズムでは，大きな障害にはならない．たとえば，行列の計算が，FFT といった規則的なデータ構造，計算モデルをもつ計算では，変数のインデックス関数は1次関数で表わされるのが一般的である．

2次元，3次元の偏微分方程式の陽／陰解法においても各プロセッサエレメントが，等しい距離の grid，あるいは等しい距離の軸を分担し，軸方向スキャンを行う場合，各 grid の変数，定数のアクセスは (base, distance) のペアで表わすことができ，RSN にとって，適切な応用と考えられる．ただし，境界条件の処理については，その問題領域におけるノウハウに属する特殊な工夫がなされていると予想され，この部分の処理については，RSN が適切であるかは明確ではない．

また，RSN ではサポートできない，あるいは非効率な応用がある．このような応用として，プラズマシミュレーションにおけるプラズマ粒子輸送コード[8]があげられる．このコードを概観すると，FFT により粒子の電荷のコンボリューションを行いポテンシャルを計算する部分と，そのポテンシャル場で各粒子ごとに運動方程式を解き，新しい粒子の位置を計算する粒子プッシュ部分に分けられる．前者の FFT は RSN にとって都合のよい処理だが，後者では，粒子の位置，すなわちインデックス，が計算により与えられるため，ランダムなアクセスパターンをもつことになる．このため，RSN では (base, distance) のペアを設定できず，サポートできない．

プラズマ粒子輸送コード以外にも，スパース行列を扱うときにも同じような問題があり，RSN はある程度，応用に制限を加える．しかし，RSN は，制御が簡単であり，メモリ競合による性能のロスがなく，かつ，ハードウェア量も少なくすむため，大規模な並列プロセッサシステムを実現する方向を示している．

6. まとめ

複数のメモリモジュールと複数のプロセッサエレメントが相互結合される並列プロセッサシステムでは，メモリ競合と結合方式が重要な問題となる．これらの問題はすでに研究がなされており，メモリモジュールの数を素数とし，p-順序ベクトルにアクセスを限定した，k-飛び相互結合が提案されている．

RSN はこのような流れの上にあるものであり，つぎのような特徴をもっている．

（1） メモリに対するアクセスを (base, distance) で表わされるパターンに限定する：変数のインデックス関数が，並列表現インデックスの一次関数で表わされるか，並列表現インデックスに依存しない場合（ただし，この場合ブロードキャスティングが必要）に限定する．

（2） RSN は回転部とスキップ部に分けられ，それぞれ回転置換により実現できる：回転置換は，ここで導入した巡回置換スイッチにより実現することができる．

（3） 巡回置換スイッチにより RSN を構成することで，2入力セレクタを単位素子とすると，$2 \cdot M \cdot \log_2(M-1)$ のオーダの素子で実現され，LSI 化も容易である．

（4） RSN は並列に同期してデータ転送を行うため，データ転送に関しては MIMD にはできない．しかし，転送されたデータについて各プロセッサエレメントが異なった処理を行うことを禁止するものではない．RSN の最も大きな問題は，（1）に起因する応用の制限である．RSN の特徴である制御の簡単さ，ハードウェア量の少なさから，大規模な並列プロセッサシステムの実現を可能にするが，この種のシステムが対象とする大規模科学計算の分野で，先に述べた応用に対する制限がどのような問題領域で問題になるか明確にする必要がある．大規模科学計算を必要とする分野の数値計算の専門家の方々の意見を期待する．

最後に，応用プログラムの提供や検討を頂いた，航空宇宙技術研究所室長三好甫博士，名大プラズマ研阿部芳彦助教授，そして助言を頂いた当社，川北健次氏，岩屋暁宏氏，渡辺貞氏に謝意を表わします．

参考文献

1) Budnik, P. and Kuck, D. J.: The Organization and Use of Parallel Memories, IEEE Trans. Comput., Vol. C-20, No. 12, pp. 1566-1569 (1971).
2) Swanson, R. C.: Interconnections for Parallel Memories to Unscramble p-Ordered Vectors, IEEE Trans. Comput., Vol. C-23, No. 11, pp. 1105-1115 (1974).
3) Stone, H. S.: The Organization of High-Speed Memory for Parallel Block Transfer of Data, IEEE Trans. Comput., Vol. C-19, No. 1, pp. 47-53 (1970).
4) Kuck, D. J.: The Structure of Computers and Computations Vol. 1, John Wiley and Son (1978).
5) Stevens, K. G. Jr.: Numerical Aerodynamics Simulation Facility Project, INFOTECH State of the art report, Supercomputers, INFOTECH International Ltd. (1979).
6) Lowrie, D. H.: Access and Alignment of Data in an Array Processor, IEEE Trans. Comput., Vol. C-24, No. 12, pp. 1145-1155 (1975).
7) Niven, I. and Zuckerman, H. S: An Introduction to the Theory of Numbers, John Wiley and Sons (1960).
8) 阿部芳彦，上村鉄雄：プラズマ粒子コードのアレイプロセッサへの適用，核融合研究，Vol. 44 (1980) 別冊その3，

（昭和56年8月21日受付）
（昭和56年11月18日採録）