論 文

삼중대각행렬 선형방정식의 解를 구하기 위한 내용-주소법 씨스톨릭 어레이

正會員 李 秉 洪* 正會員 金 正 善** 正會員 蔡 洙 煥***

Content-Addressable Systolic Array for Solving Tridiagonal Linear Equation Systems

Byeong Hong LEE*, Jeong Sun KIM**, Soo Hoan CHAE*** *Regular Members*

要 約 A가 nxn 삼중대각행렬인 선형방정식 Ax=b를 WZ분해 알고리즘을 이용하여 해석하고 이 알고리즘을 CAM (Content Addressable Memory) Systolic Array로 구현했다. 그리고 이 어레이를 평가하기 위하여 LU분해 알고리즘을 제시하고 이를 W,D,Z분해 알고리즘과 비교 고찰한 결과 LU분해 알고리즘 보다 WZ분해 알고리즘이 1/4정도 가까운 시간으로 실행시간이 단축될 수 있었다. CAM Systolic Array에서 실행되는 각 단계를 1 time step으로 가정하면 2n+1 times이 필요하고 CAM의 데이타워드는 메트릭스 원소의 값과 행 번호, 연산의 형태 및 상태에 관한 정보를 포함하고 pipeline 식으로 각 프로세서를 systolic processing하므로서 중앙제어가 필요없고, data broadcasting도 피할 수 있다.

ABSTRACT Using the WDZ decompositon algorithm, a parallel algorithm is presented for solving the linear system Ax=b which has an nxn nonsingular tridiagonal matrix. For implementing this algorithm a CAM (content-addressable memory) systolic array is proposed, and each processing element of this array has its own CAM to store the nonzero elements of the tridiagonal matrix. In order to evaluate this array the algorithm presented is compared to the LU decompositon algorithm. It is found that the execution time of the algorithm pressnted is reduced to about 1/4 than that of the LU decomposition algorithm. If each computation process step can be done in one time unit, the system of equations is solved in a systolic fashion without central control and the solution is obtained in 2n+1 time steps.

I. 序 論

삼중대각행렬 선형방정식을 해석하는 것은 자연과학의 계산을 하는 많은 프로그램에서 이용되고 있으며, 병렬 및 백타컴퓨터의 최근의 다양한 개발 및 可用性과 함께 이들 기계에 적합한 삼중대각행렬 선형방정식을 해석하는 새로운 알고리즘이 많이 보고 되어 있다[1,2,3]. 삼중대각행렬은 띠의 대역폭이 3인 특수한 띠행렬이며, 띠행렬에 적합한 cellular array가 제안 되었다 H.S.Stone 은 recursive doubling기법으로 recursive한 알고리즘을 만들어서 삼중대각행렬의선형방정식을 해석했으며[3], H.H.Wang은 分割方法으로 matrix A를 kxk의 소행렬로 분할하여 pxp블럭으로 만들어 M개의 프로세서로 실행시켰다[2]. 그리고 D.J.Evans와 M.Hatzopoulos는 삼중대각행렬 A를 WZ로 분해하여 Ax=b의 선형방정식을

*島山專門大學 電子計算學科
Dept. of Computer Science, Osan Junior College
**韓國航空大學 電子工學科
Dept. of Electronics Eng., Han Kuk Aviation University
***韓國航空大學 電子計算學科
Dept. of Computer Science, Han Kuk Aviation University
論文番號 : 91-53(接受 1991. 4. 13)

해석하는 알고리즘을 제안했다[8]. 한편 R.M. Kieckhager et al.은 sparse matrix계산을 위해서 CAM(content addressable memory)를 사용한 구조를 제안했으며, 이들이 제안한 구조는 PE (processing element)들의 선형 array로 구성되고 각각은 그 자신의 CAM과 국부제어를 가지고 있지만 array는 systolic이나 data driven이 아니고 데이타 전파(broadcast)가 행해진다[9]. 그리고 O.Wing이 보고한 CAM systolic array는 문제에 본래부터 존재하는 sparsity와 병렬성을 이용하여 sparse 선형방정식의 解를 얻는 계산구조이며 이 구조는 圖形으로 연결된 다수의 PE들로 구성한 CAM systolic array로서 각 PE는 CAM에 연결되고 데이타가 ring주위로 pipeline될때 방정식은 중앙제어 없이 systolic 방식으로 해석된다[6].

Ko와 O.Wing은 수치적 안정성 제어를 하는 CAM systolic ring구조를 제안했으며[7], 이것은 消去段과 피봇단(pivot stage)으로 되어 消去段은 CAM systolic ring상에서 병렬로 행해질 수 있으나 피봇단은 順次的으로 처리된다.

본 논문에서는 W,D,Z분해 알고리즘에 의해서 삼중대각행렬을 분해하여 미지벡타를 구하는 선형방정식의 병렬 解 알고리즘을 만들고 이를 LU분해 알고리즘의 경우와 비교 고찰하였다. 그리고 W,D,Z분해 알고리즘에 의한 선형방정식의 병렬 解 알고리즘을 CAM systolic array로 具現하였다. 논문 작성 순서는 제2절에서 삼중대각행렬의 병렬 解 알고리즘을 구하고, 제3절에서 이 알고리즘과 LU분해 알고리즘을 비교 고찰한다. 그리고 제4절에서 CAM systolic array를 具現하고, 제5절에서 프로세서간 상호통신에 관하여 기술한다음 제6절에서 결과를 고찰하고 결론을 맺는다.

Ⅱ. 삼중대각행렬의 竝列 解

선형방정식 Ax=b에서 A는 nxn 정칙 삼중대각행렬(nonsingular tridiagonal matrix)로서 A=[Aij]nxn이고 b는 旣知數, x는 未知數로서 둘다 $b=[b_1, b_2, \cdots\cdots, b_n]^t$, $x=[x_1, x_2, \cdots\cdots, x_n]^t$인 n차원의 벡타이다. 지금 A=WDZ의 관계를 만족하는 W,D,Z의 행렬이 존재한다고 가정하면 행렬 D는 n차원의 대각선 행렬이고 W와 Z는 $W=[W_{ij}]_{nxn}$, $Z=[Z_{ij}]_{nxn}$으로 다음 조건을 만족하는 것으로 한다.

$$W=\begin{cases} W_{ij}: & j=i-1,\ (i=2,3,\cdots\cdots,n/2) \\ & j=i+1,\ (i=n/2,\cdots\cdots,n-1) \\ 1 \ : & i=j \\ 0 \ : & \text{otherwise} \end{cases} \qquad \text{----------(2-1)}$$

$$Z=\begin{cases} Z_{ij}: & j=i+1,\ (i=1,2,\cdots\cdots,n/2 \\ & j=i-1,\ (i=n/2,\cdots\cdots,n) \\ 1= : & i=j \\ 0= : & \text{otherwise} \end{cases} \qquad \text{---------(2-2)}$$

$$\begin{bmatrix} A_{11} & A_{12} & 0 & 0 & \cdots & 0 \\ A_{21} & A_{22} & A_{23} & 0 & \cdots & 0 \\ 0 & A_{32} & A_{33} & A_{34} & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & A_{n,n-1} & A_{n,n} & \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ W_{21} & 1 & 0 & \cdots & 0 \\ 0 & W_{32} & 1 & 0 & \cdots \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & W_{n-1,n} & 1 \end{bmatrix} \begin{bmatrix} D_{11} & 0 & 0 & \cdots & 0 \\ 0 & D_{22} & 0 & \cdots & 0 \\ 0 & 0 & D_{33} & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & D_{n,n} \end{bmatrix} \begin{bmatrix} 1 & Z_{12} & 0 & \cdots & 0 \\ 0 & 1 & Z_{23} & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & Z_{n-1,n} & 1 \end{bmatrix}$$

---------(2-3)

위 식(2-1), (2-2), (2-3)으로 부터 W,D,Z 분해 알고리즘은 다음과 같다.

프로시져 WZ-1 : (WZ 분해)
Begin
D(1)=A(1,1)
For i=2 to n/2 do
D(i)=A(i,i)-A(i,i-1)* A(i-1,i)/D(i-1)

W(i,i-1)=A(i,i-1) / D(i-1)
Z(i-1,i)=A(i-1,i) / D(i-1)
End for
End

프로시저 WZ-2 ; (WZ 분해)
Begin
D(n)=A(n,n)
For i=n-1 downto (n / 2)+1 step -1 do
D(i)=A(i,i)-A(i+1,i)* A(i,i+1) / D(i+1)
W(i,i+1)=A(i,i+1) / D(i+1)
Z(i+1,i)=A(i+1,i) / D(i+1)
End for
End

따라서 선형방정식 Ax=b에 대한 알고리즘을 구하기 위해서 보조벡타 y와 ϕ를 사용하여
Ax=WDZx=b로 부터
DZx=y
Zx=ϕ
로 놓으면
Wy=b
Dϕ=y
Zx=ϕ

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ w_{21} & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & w_{32} & 1 & 0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & & \vdots \\ & & & 1 & w_{n-2,n-1} & 0 & \\ & & & & 1 & w_{n-1,n} & \\ 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-2} \\ y_{n-1} \\ y_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{bmatrix}$$

------(2-4)

이를 해석하는 알고리즘은 다음과 같이 되며, 식 (2-4)로 부터

프로시저 W-1 ; (Wy=b)
Begin
y(1)=b(1)
For i=2 to n / 2 do
y(i)=b(i)-y(i-1)* W(i,i-1)
End for
End

프로시저 W-2 ; (Wy=b)
Begin
y(n)=b(n)
For i=n-1 downto (n / 2)+1 step -1 do
y(i)=b(i)-y(i+1)* W(i,i+1)
End for
End

로 되고, 식(2-5)로부터

$$\begin{bmatrix} D_{11} & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & D_{22} & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & D_{33} & 0 & 0 & \cdots & 0 \\ \vdots & & & \ddots & & & \vdots \\ & & & & D_{n-1,n-1} & 0 & \\ 0 & 0 & 0 & 0 & 0 & D_{n,n} & \end{bmatrix} \begin{bmatrix} \phi 1 \\ \phi 2 \\ \phi 3 \\ \vdots \\ \phi_{n-1} \\ \phi_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}$$

-------(2-5)

ϕ(1)=y(1) / D(1,1)
ϕ(2)=y(2) / D(2,2)
.
.
ϕ(n)=y(n) / D(n,n)
이다.

식 (2-6)으로 부터 미지벡터 x를 구하는 알고리즘은 다음과 같다.

프로시저 Z-1 ; (Zx=ϕ)
Begin
x(n / 2+1)=ϕ(n / 2+1)
For i=n / 2 downto 1 step -1 do
x(i)=ϕ(i)-Z(i,i+1)* x(i+1)
End for
End

프로시저 Z-2 ; (Zx=ϕ)

```
Begin
  For i=(n/2)+2 to n do
  x(i)=φ(i)-Z(i,i-1)* x(i-1)
  End for
End
```

$$\begin{bmatrix} 1 & z_{12} & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & z_{23} & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & z_{34} & 0 & \cdots & 0 \\ \cdots & & & & & & \\ \cdots & & & z_{n-1,n-2} & 1 & 0 & \\ 0 & 0 & 0 & 0 & 0 & z_{n,n-1} & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_{n-1} \\ \phi_n \end{bmatrix}$$

---------(2-6)

Ⅲ. LU 분해 알고리즘과 WZ 분해 알고리즘의 비교

행렬의 LU 형태 분해가 본래 직렬형 알고리즘인데 반해 WZ 형태 분해는 병렬형에 가까운 알고리즘이다. WZ 분해 알고리즘에 의한 삼중대각행렬의 평가를 하기위해서 먼저 선형방정식의 解를 구하는 데 표준이 되는 LU분해 알고리즘을 이용하여 解를 유도하고 이 두개의 알고리즘을 비교 고찰한다.

알고리즘에서 산술연산은 matrix의 전달연산, 곱셈(또는 나눗셈), 덧셈(또는 뺄셈)이며 실행시간은 전달연산은 무시하고 나머지 산술연산에 대해서 모두 하나의 단위 시간으로 계산한다.

다음은 삼중대각행렬 A의 선형방정식 Ax=b 에서 A를 L,D,U로 각각 분해하여 방정식의 解를 구하는 알고리즘이다.

```
프로시져 LU ; (LU 분해)
Begin
  D(1)=A(1,1)
  For k=2 to n do
  각각의 k에 대해서
    L(k,k-1)=A(k,k-1)/D(k-1)
  와
    U(k-1,k)=A(k-1,k)/D(k-1)
  은 병행 연산
    D(k,k)=A(k,k)-L(k,k-1)* D(k-1)
          *U(k-1,k)
  End for k
End

프로시져 L ; (Ly=b)
Begin
  y(1)=b(1)
  For k=2 to n do
  y(k)=b(k)-y(k-1)* L(k,k-1)
  End for k
End

프로시져 D ; (Dφ=y)
Begin
  For all 2<=k<=n do
  φ(k)=y(k)/D(k)
  End for k
End

프로시져 U ; (Ux=φ)
Begin
  x(n)=φ(n)
  For k=2 downto 1 step -1 do
  x(k)=φ(k)-φ(k+1)* U(k,k+1)
  End for k
End
```

위 알고리즘에서 산술연산의 수는 프로시져 LU에서 5, 프로시져 L에서 2, 프로시져 D에서 1, 프로시져 U에서 2등이며, 각 프로시져에서 실행되는 시간을 고려하면 삼중대각행렬 선형방정식의 解를 구하는 데 걸리는 시간은 8n-7 이다.

제2절에서 설명한 WZ분해 알고리즘에 의하면 산술연산의 수는 프로시져 WZ에서 5, 프로시져 W에서 2, 프로시져 D에서 1, 프로시져 Z에서 2등이며, 각 프로시져에서 실행시간을 고려하면 삼중대각행렬 선형방정식을 실행하는 데 걸리는 시간은 2n-3이다.

동일한 삼중대각행렬의 선형방정식을 H.S.

Stone 은 recursive doubling기법으로 해석하는데 산술연산의 수가 $17\log_2 n+7$이었다[7].

D.J.Evans와 M.Hatzopoulos는 동일한 삼중대각행렬 A를 WZ로 분해하는 데 있어서 $2(n-2)$개의 프로세서로 $6\lfloor(n-2/2)\rfloor$ time steps가 걸렸으며, Ax=b의 선형방정식을 해석하는 데는 $6\lfloor(n-1/2)\rfloor$ time steps가 걸렸다. 한편 M.Hatzopulos는 역시 동일한 삼중대각행렬 A를 WDZ로 분해하는 데 2n개의 프로세서로서 2logn time steps이 걸렸고, Ax=b의 선형방정식을 해석하는 데 2n개의 프로세서로서 4logn+5 time steps이 걸렸다.

알고리즘	산술연산의 총수	실행 시간	최대 병렬성
LU	1n	8n-7	2
WZ	1n	2n-3	4

註, n : 방정식의 수

표 3-1 LU알고리즘과 WZ알고리즘의 비교
Comparison of LU algorithm and WZ algorithm

Ⅳ. CAM SYSTOLIC ARRAY

array는 래지스타(latch)와 함께 n개의 프로세서로서 그림 4-1(a)와 같이 구성된다. 각 프로세서는 메모리스의 열과 관계되며 CAM에 연결된다. 각 메모리 워드(memory word)는 WA(i,j)로 표시하고, 그림 4-1(b)와 같이 4개의 필드(field)로 구성된다. 이들 필드중에서 연산비트(operation bit)는 데이타가 프로세서에서 프로세서로 array를 따라서 이동될때 필요로 하는 연산의 종류를 표시한다. 그리고 플래그 비트는 연산전과 연산후에 상태를 표시한다. array에서 알고리즘 수행과정은 그림 4-2와 같으며, 다음에 설명하는 바와 같다.

(Step-1)

PE-1이 CAM-1에서 WA(1,1)과 WA(2,1)을 선택한 후 WA(1,1)의 연산비트가 나눗셈 연산을 표시하고, WA(2,1)의 연산비트가 更新演算을 표시하므로서 A(2,1) : =A(2,1) / A(1,1)의 연산이 PE-1에 의해서 실행된다. 이 연산결과는 CAM-1의 WA(2,1)에 저장됨과 동시에

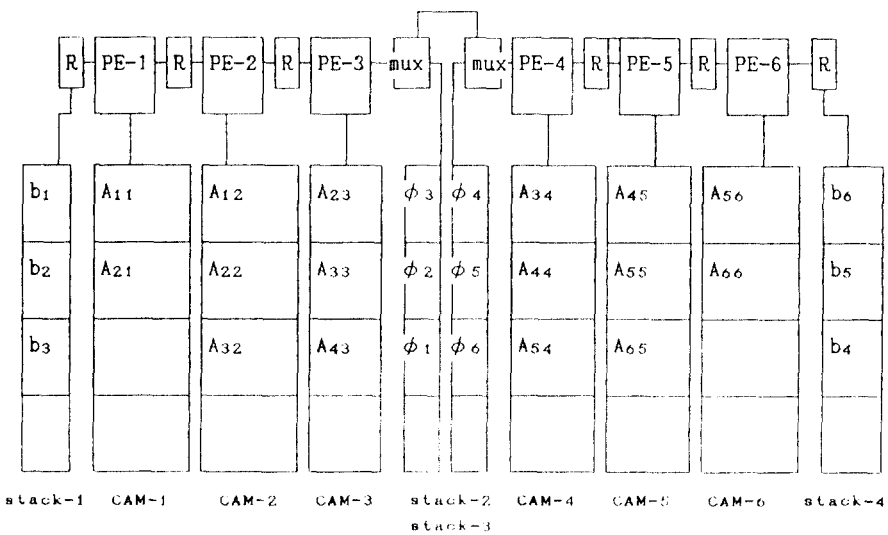

그림 4-1 (a) n=6의 경우 CAM systolic array
Content Addressable Memory systolic array in case of n=6

메트릭스 원소의 값(value of element)	행 번호 (number of row)	연산 비트 (operation bit)	플랙 비트 (flag bit)

그림 4-1 (b) CAM의 워드구조
Word structure of CAM

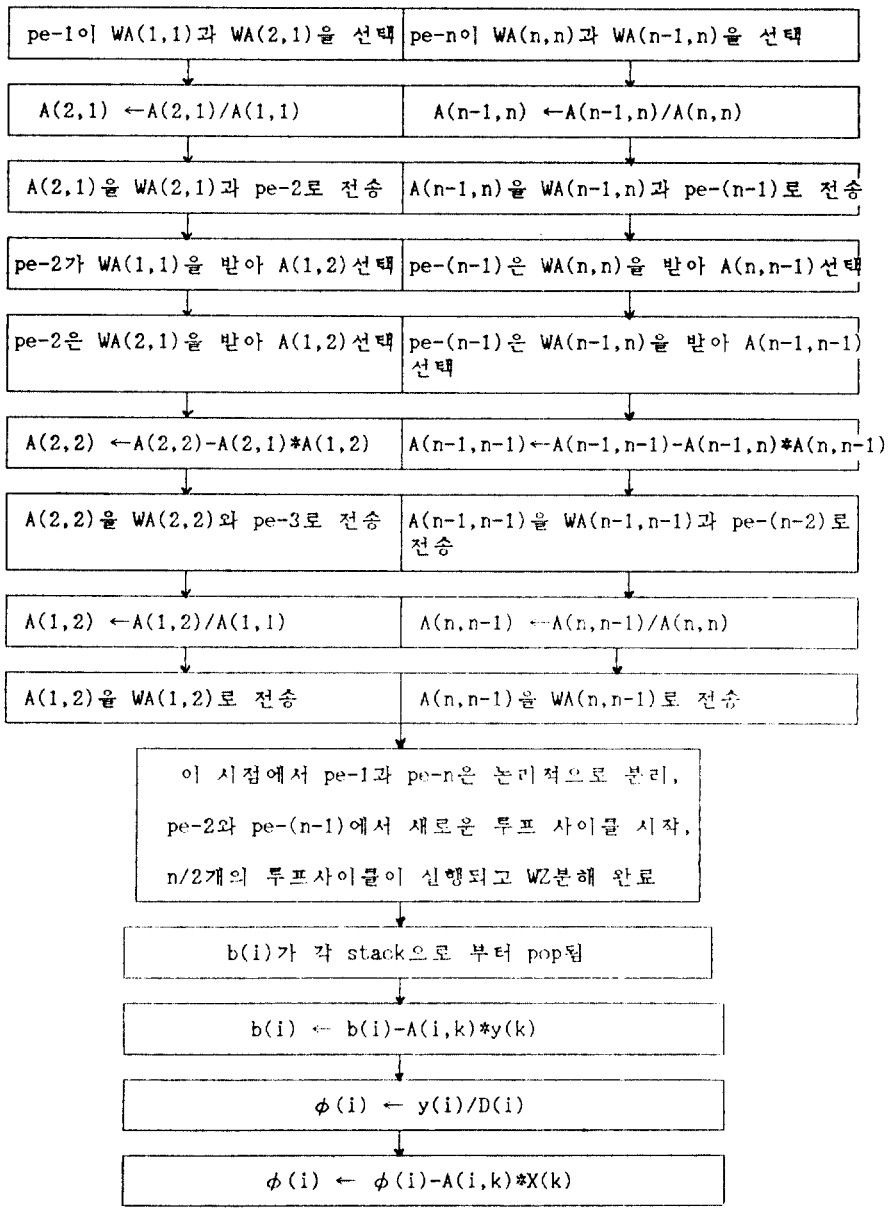

그림 4-2 array상에서 알고리즘 수행과정
Execution procedure of an algorithm in the CAM systolic array

PE-2로 WA(1,1)과 함께 보내진다. WA(2,1)에 저장된 A(2,1)은 다음에 W(2,1)의 값으로 된다.

PE-n은 CAM-n에서 WA(n,n)과 WA(n-1, n)를 선택한후 WA(n,n)의 연산비트가 나눗셈 연산을 표시하고 WA(n-1,n)의 연산비트가 更新演算을 표시하므로서 A(n-1,n) : =A(n-1,n) / A(n,n)의 연산이 PE-n에 의해서 실행된다. 이 연산의 결과는 CAM-n의 WA(n-1,n)에 저장됨과 동시에 PE-(n-1)로 WA(n,n)과 함께 보내진다. WA(n-1,n)에 저장된 A(n-1,n)은 다음에 W(n-1,n)의 값으로 된다.

PE-2는 WA(1,1)를 받아서 WA(1,1)의 행번호를 키로하여 같은 행의요소 A(1,2)를 선택하고, PE-(n-1)는 WA(n,n)를 받아서 WA(n,n)의 행번호를 키로 하여 같은 행의 요소 A(n,n-1)를 선택한다.

(Step-2)

PE-2는 WA(2,1)를 받아서 WA(2,1)의 행번호를 키로 하여 같은 행의 요소 A(2,2)를 선택하고, WA(2,1)의 연산비트가 更新演算을 표시하므로서 A(2,2) : =A(2,2)-A(2,1)*A(1,2)의 연산이 PE-2에 의해 실행된다. 更新된 결과는 CAM-2의 WA(2,2)에 저장됨과 동시에 PE-3로 보내진다.

PE-(n-1)는 WA(n-1, n)를 받아서 WA(n-1,n)의 행번호를 키로 하여 A(n-1,n-1)를 선택하고 연산비트가 更新演算을 표시하므로서 A(n-1,n-1) : =A(n-1,n-1)-A(n-1,n)*A(n,n-1)의 연산이 PE-(n-1)에 의해서 실행된다. 更新된 결과는 CAM-(n-1)의 WA(n-1,n-1)에 저장됨과 동시에 PE-(n-2)로 보내진다.

(Step-3)

PE-2는 WA(1,1) 연산비트가 나눗셈 연산을 표시하므로서 A(1,2) : =A(1,2) / A(1,1)의 연산을 실행하고, 연산결과는 CAM-2의 WA(1,2)에 저장된다. 이것은 다음에 Z(1,2)의 값으로 된다.

PE-(n-1)은 WA(n,n)의 연산비트가 나눗셈 연산을 표시하므로서 A(n,n-1) : =A(n,n-1) / A(n,n)의 연산을 실행하고 연산결과는 CAM-(n-1)의 WA(n,n-1)에 저장된다. 이것은 다음에 Z(n,n-1)의 값으로 된다.

이 시점에서 PE-1과 PE-n은 논리적으로 array로 부터 분리되며 하나의 루프사이클이 끝나고 새로운 루프사이클이 PE-2와 PE-(n-1)에서 동시에 시작되면서 PE-2는 WA(3,2)를 선택하고 PE-(n-1)은 WA(n-2,n-1)을 선택한다. 이때 연산비트는 나눗셈 연산을 표시하며, 이렇게 해서 n/2개의 루프사이클이 모두 수행되면 실행은 끝나고 마지막 사이클에서는 Z(n/2,n/2+1)만 계산한다. 그리고 루프사이클내에 모든 연산을 1 time step으로 간주하면 CAM의 원래의 위치에 저장된 모든 W,Z원소들을 구하는데 걸리는 시간은 n개의 프로세서로서 n+1 time steps이 된다.

Wy=b를 해석하기위하여 b를 PE-1의 좌측과 PE-n의 우측에 있는 stack에다 그림 3-1(a)과 같이 놓고 각 루프사이클이 실행될때 b의 원소 b(i)는 각 stack에서 차례로 모든 프로세서로 팝(pop)된다. k번째 프로세서에서 b(i)는 (k=1,2,……,n/2 인때 i=k+1과 k=n, n-1,……,(n/2)+2인때 i=k-1) b(i)=b(i)-A(i,k)*y(k)로 更新되고 모든 y(i)를 계산하는데 걸리는 시간은 (n/2)-1 steps이 된다. 계산과정은 그림 4-3에 표시되어 있으며 각각의 y벡터는 다음의 ϕ벡터를 계산하기 위하여 프로세서내의 레지스터에 보관되었다가 ϕ벡터로 계산된다음 stack에 보관된다.

Dϕ=y를 해석하는 것은 D가 대각선행렬이므로 n개의 프로세서로서 1 time step으로 계산될 수 있다.

Zx=ϕ를 해석하기위해서 ϕ의 원소 ϕ(i)는 각 stack에서 차례로 팝되며 한번에 하나씩 프로세서에 보내진다. 처음에는 ϕ(n/2+1)이 팝되고 다음부터는 2개의 stack에서 동시에 하나씩 팝된다. k(k=2,3,…,n-1)번째 프로세서에서 i=1,2,…,n/2 및 i=n/2+2,……,n에 대하여 ϕ(i)는 ϕ(i)=ϕ(i)-A(i,k)*x(k)로 更新되며

n / 2 time steps 후에 모든 x(i)가 계산된다. 결과적으로 n개의 프로세서로서 Ax=b를 해석하기 위해서는 2n+1 time steps이 필요하다.

step	PE-1	PE-2	PE-3
1	A(2,1)← A(2,1) / A(1,1)	A(1,2)	
2		A(2,2)← A(2,2)-A(2,1)* A(1,2)	
3		A(1,2)← A(1,2) / A(1,1)	
4		A(3,2)← A(3,2) / A(2,2)	A(2,3)
5			A(3,3)← A(3,3)-A(3,2)* A(2,3)
6			A(2,3)← A(2,3) * A(2,2)

step	PE-4	PE-5	PE-6
1		A(6,5)	A(5,6)← A(5,6) / A(6,6)
2		A(5,5)← A(5,5)-A(5,6)* A(6,5)	
3		A(6,5)← A(6,5) / A(6,6)	
4	A(5,4)	A(4,5)← A(4,5) / A(5,5)	
5	A(4,4)← A(4,4)-A(4,5)* A(5,4)		
6	A(5,4)← A(5,4) A(5,5)		
7	A(3,4)← A(3,4) / A(3,3)		

그림 4-3(a) n=6의 경우 systolic arrary상에서 pipeline된 W,D,Z분해과정
Pipelined WDZ decomposition on systolic array in case of n=6

step	PE-1	PE-2	PE-3
1	y(1)←b(1) y(2)←b(2)-y(1)*W(2,1)	y(2)	
2		y(3)← b(3)-y(2)*W(3,2)	y(3)

step	PE-4	PE-5	PE-6
1		y(5)	y(6)←b(6) y(5)←b(5)-y(6) * W(5,6)
2	y(4)	y(4)← b(4)-y(5)*W(4,5)	

그림 4-3(b) n=6의 경우 Wy=b의 解
Systolic solution of Wy=b in case of n=6

step	PE-1	PE-2	PE-3
1			x(3)
2		x(2)	x(2) ϕ(2)-Z(2,3)*x(3)
3	x(1)	x(1)← ϕ(1)-Z(1,2)*x(2)	

step	PE-4	PE-5	PE-6
1	x(4)← ϕ(4) x(3)← ϕ(3)-Z(3,4)*x(4)		
2	x(5)← ϕ(5)-Z(5,4)*x(4)	x(5)	
3		x(6)← ϕ(6)-Z(6,5)*x(5)	x(6)

그림 4-3(c) n=6의 경우 Zx=ϕ의 解
Systolic solution of Zx=ϕ in case of n=6

V. 프로세서간 상호통신

각 PE는 독립된 clock을 가지고 있으며 2개의 이웃하는 PE들 사이에 데이타 전송은 비동기적으로 행해진다. PE-i가 계산을 마치면 PE-(i+1)이 데이타를 수신할 준비가 되었는지를 점검한다. 만약에 준비가 되었다면 PE-(i+1)에다 "송신준비 완료" 신호를 보낸다. 데이타를 전송후 PE-i는 PE-(i-1)이 데이타를 전송할 준비가 되었는지를 점검한다. 만약에 전송할 준비가 되었다면 PE-i는 데이타를 수신하고 그렇지 않으면 PE-i는 PE-(i-1)에다 "수신준비 완료" 신호를 보낸다. CAM이 행렬의 列과 관계가 있고 CAM속의 각 워드는 행렬 원소의 값 뿐만 아니라 행렬의 원소가 속하는 행의 번호와 다른 정보를 가지고 있으므로 데이타가 입력 format이 되면 각 data item은 systolic array로 하나씩 들어가 각 data item의 列 번호에 해당하는 CAM에 저장된다. 외부장치로부터 프로세서에 READ/WRITE 신호는 없고 신호는 data item의 列 번호와 프로세서 번호가 대응해서 국부적으로 프로세서에서 생성된다.

VI. 결 론

삼중대각행렬 A의 선형방정식 Ax=b를 해석하기 위하여 A의 원소들을 WZ알고리즘에 의해서 분할하므로서 LU알고리즘에 의해 분할한 것보다 1/4정도의 가까운 시간으로 실행할 수 있다.

이 알고리즘을 CAM systolic array로서 具現하여 각 step에서 수행되는 시간을 1time step으로 하면 2n+1 time steps이 걸리고 CAM에 기억되는 data item들이 메트릭스 원소들의 값 뿐만아니라 행번호와 연산의 형태 및 상태에 관한 정보를 가지고 sytolic 프로세싱 하므로서 중앙제어를 하지않아도 된다. 각 列의 원소들은 CAM으로 부터 나눗셈이나 更新演算을 위하여 선택되고 병렬연산을 중첩하므로서 systolic 프로세싱이 가능하다. 따라서 data broadcasting을 제할 수 있다. 그리고 벡타 b나 ϕ를 스택에다 놓으므로서 동일한 array를 최종적인 解를 얻을 때까지 계속해서 사용할 수 있다.

참 고 문 헌

1. Don Heller, "Some aspects of the cycle reduction algorithm for block tridiagonal linear systems", SIAM J.Numer.Anal., Vol.13, No.4, pp.484-496, Sept.1976.
2. H.H.Wang, "A parallel method for tridiagonal equations", ACM Trans. Math. Soft., Vol.7, No.2, pp.170-183, June 1981.
3. H.H.Stone, "An efficient parallel algorithm for the solution of a tridiagonal linear system of equations", J.of Association for Computing Machinery, Vol.20, No.1, pp.27-38, Jan.1973.
4. Fawcett, J.W. "Solution of large sets of linear equations using cellular arrays", Ph.D.dissertation, Syracuse University, Syracuse, N.Y., June.1981.
5. Kieckhager, R.M. and Pottle, C., "A processor array for factoization of unstructured space networks", Proc. IEEE Int. Conf.on Circuits and Computers, pp.380-383, 1982.
6. Omar Wing, "A content addressable systolic array for sparse matrix computation", J.of parallel and distributed computing 2,pp.170-181, 1985.
7. Chung K.Ko and Omar Wing, "A systolic sparse linear equations solver with numerical stability control", IEEE Trans. Comput. pp.290-293, 1986.
8. D.J.Evans and M.Hatzpoulos, "A parallel linear system solver", Intern. J.Computer Math., Vol.7, pp.227-238, 1979.
9. Chin Wen Ho and R.C.T Lee, "A parallel algorithm for solving sparse triangular systems", IEEE Trans.on Computers, Vol.39, No.6, pp.848-852, June 1990.
10. Kai Hwang and Yeng Heng Cheng, "Partitioned matrix algorithms for VLSI arithmetic systems", IEEE Trans. on computer, Vol.c-31, No.12, pp.1215-1224, Dec.1982.
11. D.J.Evans and A.Hdjidimos, "A modification of the quadrant interlocking factorisation parallel mathod", Intern.J.Computer Math., Vol.8, pp.149-166, 1980.
12. D.J.Evans, et al, "The parallel solution of banded linear equations by the new quadrant interlocking

factorisation mathod", Intern.J.Computer Math., Vol. 9, pp.151-161, 1981.
13. P.C.Mathias and L.M.Patnaik, "Systolic evaluation of polynominal expressions", IEEE Trans.on Computers, Vol.39, No.5, pp.653-665, May 1990.
14. G.R.Gao, "A pipelined solution mathod of tridiagonal linear equation system", Proce.of Intern. Conf. parallel processing, pp.84-91, Aug.1986.
15. Ferng-Ching Lin and Kung Chen, "On the design of a unidirectional systolic array for key enumeration", IEEE Trans.on Computers, Vol.39, No.2, pp. 266-269, Feb.1990.
16. Wei-ming Lin and V.K.Prasanna Kumar, "A note on the linear transformation method for systolic array design", IEEE Trans.on Computers, Vol.39, No.3, pp.393-399, March 1990.

李 秉 洪(Byeong Hong LEE) 正會員
1944年 12月 6日生
1970年 2月 : 韓國航空大學通信工學科卒業(學士)
1982年 2月 : 東國大學校大學院電子工學科卒業(碩士)
1988年 9月 ~ 現在 : 韓國航空大學大學院 電子工學科 博士課程
1974年 11月 ~ 1979年 2月 : 麗水水産專門大學 專任講師
1979年 3月 ~ 現在 : 鳥山專門大學 敎授

蔡 洙 煥(Soo Hoan CHAE) 正會員
1950年 10月 28日生
1973年 2月 : 韓國航空大學 電子工學科卒業(工學士)
1973年 7月 ~ 1977年 8月 : 空軍技術將校
1977年 8月 ~ 1983年 7月 : 金星通信硏究所
1983年 8月 ~ 1985年 5月 : 美國알라바마大學電算學科(理學碩士)
1985年 5月 ~ 1988年 12月 : 美國알라바마大學電氣工學科(工學博士)
(電子計算機構造專攻)
1989年 3月 ~ 現在 : 韓國航空大學 電子計算學科 助敎授

金 正 善(Jeong Sun KIM) 正會員
1941年 5月 5日生
1965年 ~ 1990年 現在 : 航空大學 敎授
1988年 ~ 1990年 : 電子工學科 學科長
1990年 2月 : 電子計算所長

1990.1月 ~ 현 한국통신학회 이사