Systolic Architecture Design

Shao-Yi Chien

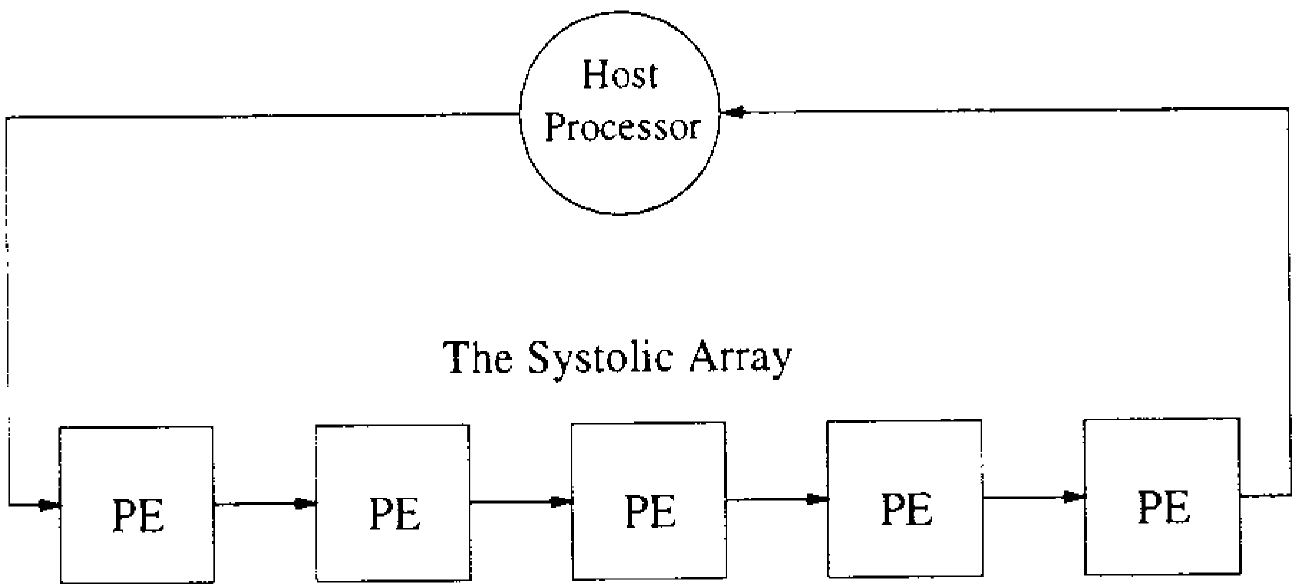
Some materials are referred to
S. Y. Kung, *VLSI Array Processors*, Prentice-Hall, 1988.

Introduction (1/3)

- Systolic architecture (systolic array)
  - A network of processing elements (PEs) that rhythmically compute and pass data through the system
  - Modularity and regularity
  - All the PEs in the systolic array are uniform and fully pipelined
  - Contains only local interconnection

Introduction (2/3)

- Typical systolic array

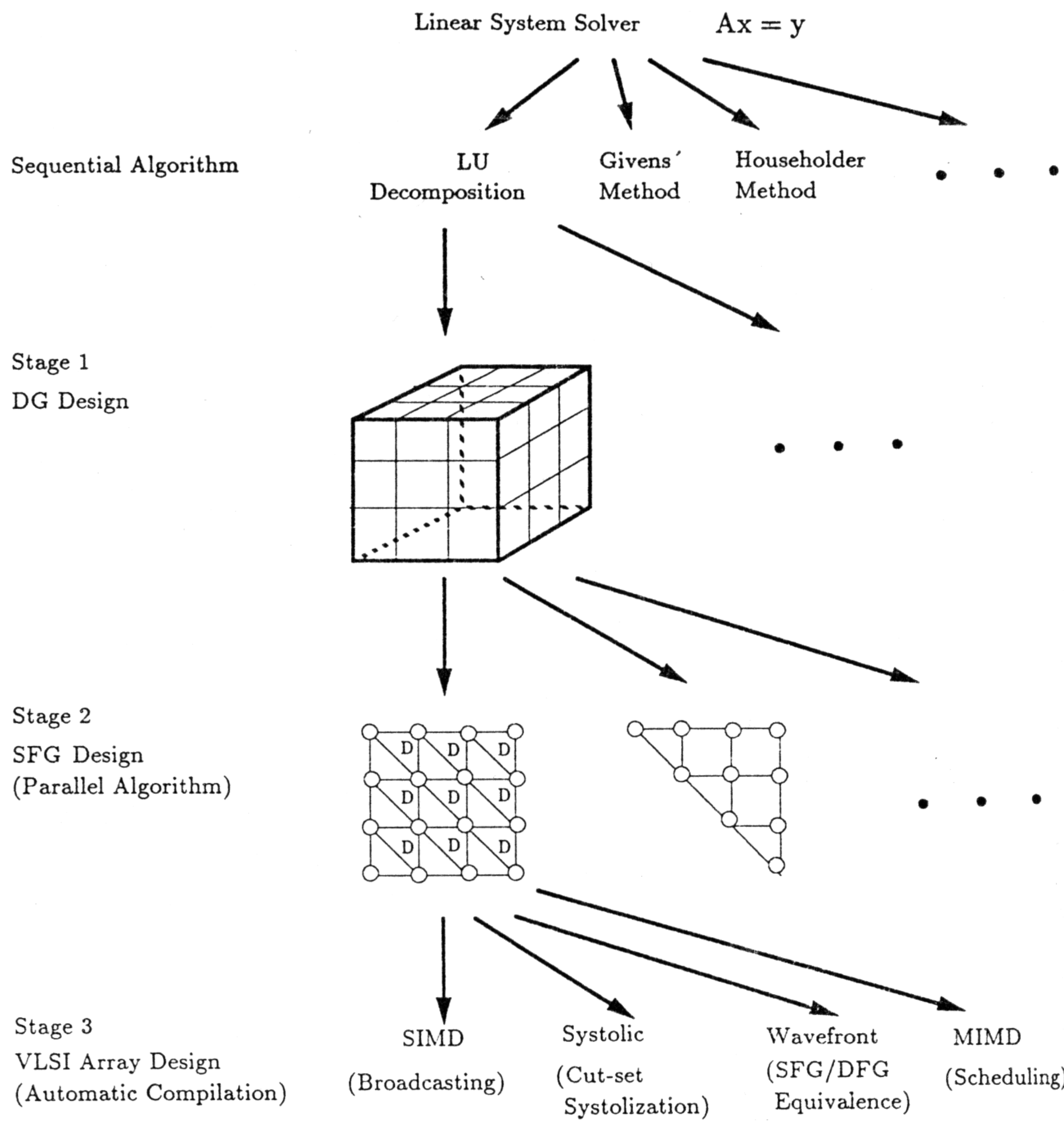


Introduction (3/3)

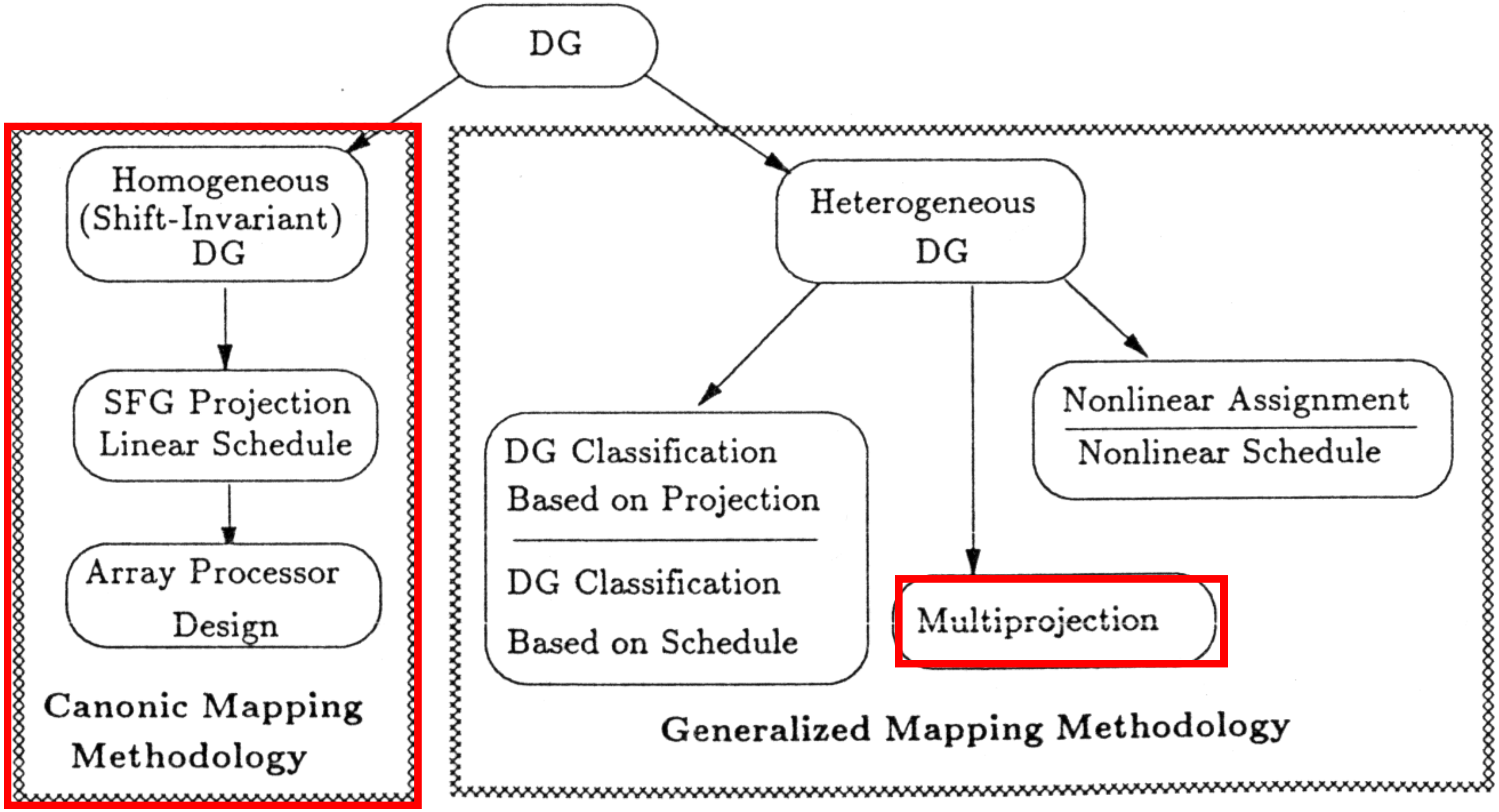
- Some relaxations
  - Not only local but also neighbor interconnections
  - Use of data broadcast operations
  - Use of different PEs in the system, especially at the boundaries
  - Also called as “semi-systolic array”

Systematic Design Methodology

- Step 1
  - DG design
- Step 2
  - Mapping to DFG
- Step 3
  - VLSI array design



Generalized Mapping



Canonic Mapping Methodology

- Applying **linear mapping** on regular **dependence graph (DG)**
- The DG corresponds to a space representation where no time instance is assigned to any computation
- Maps N-dimensional DG to a lower level dimensional systolic architecture
- We first introduce N-dimension to (N-1)-dimension mapping

Step 1: DG Design

- Purpose: use graph to represent an algorithm with parallel expression
  - Avoid unnecessary ordering information introduced by sequential code
  - The important first step of systolic array design

DG Design

- Write recursive form of an algorithm
- Transform it as **Single Assignment Code**
- Draw DG
- Localized DG

Example: Matrix-Vector Multiplication (1/7)

$$c = Ab$$

$$c_i = \sum_{j=1}^{m} A_{ij} b_j$$

- Write recursive form of an algorithm

```
for(i=1;i<=4;i++)
{
        c(i)=0;
        for(j=1;j<=4;j++)
        {
                c(i)=c(i)+A(i,j)*b(j);
        }
}
```

Example: Matrix-Vector Multiplication (2/7)

- Transform it as Single Assignment Code
  - A **Single Assignment Code** is a form where every variable is assigned one value only during the execution of the algorithm

```
for(i=1;i<=4;i++)
{
        c(i,1)=0;
        for(j=1;j<=4;j++)
        {
                c(i,j+1)=c(i,j)+A(i,j)*b(j);
        }
}
```

Example: Matrix-Vector Multiplication (3/7)

- A recursive algorithm is inherently given in a single assignment code

$$c = Ab$$

$$c_i^{(j+1)} = c_i^{(j)} + a_i^{(j)} b_i^{(j)}$$

$$c_i^{(1)} = 0$$

$$a_i^{(j)} = A(i, j)$$

$$b_i^{(j)} = b(j)$$

Broadcast Signal

Example: Matrix-Vector Multiplication (4/7)

- Draw DG
  - A DG can be considered as the graphical representation of a single assignment algorithm
  - DG specifies all the dependencies between all variables in the index space
  - An algorithm is **computable** if and only if its complete DG contains no loops or cycles

Example: Matrix-Vector Multiplication (5/7)

```
for(i=1;i<=4;i++)
{
        c(i,1)=0;
        for(j=1;j<=4;j++)
        {
                c(i,j+1)=c(i,j)+A(i,j)*b(j);
        }
}
```

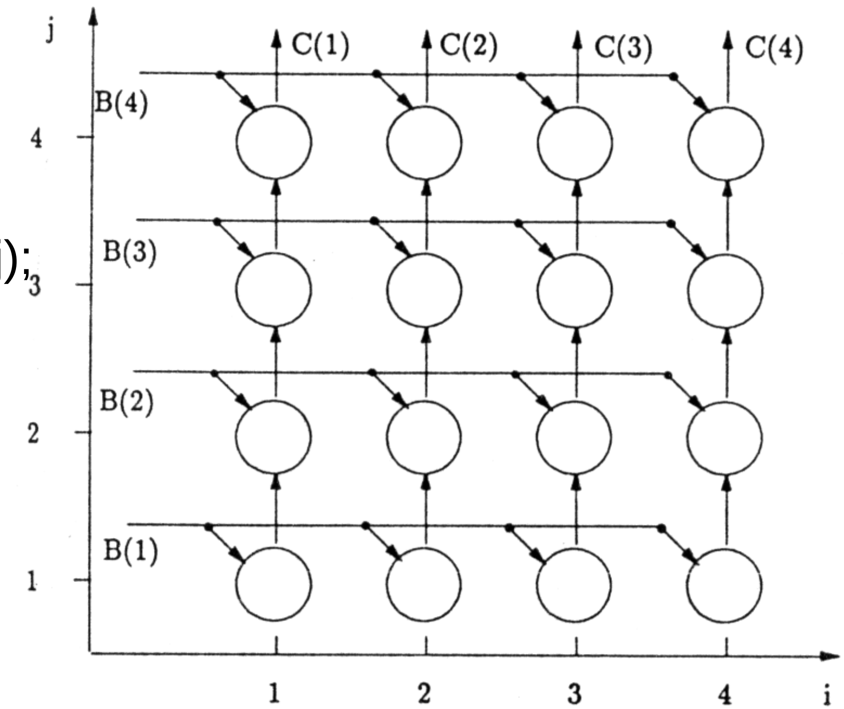
$$c = Ab$$

$$c_i^{(j+1)} = c_i^{(j)} + a_i^{(j)} b_j^{(j)}$$

$$c_i^{(1)} = 0$$

$$a_i^{(j)} = A(i, j)$$

$$b_i^{(j)} = b(j)$$



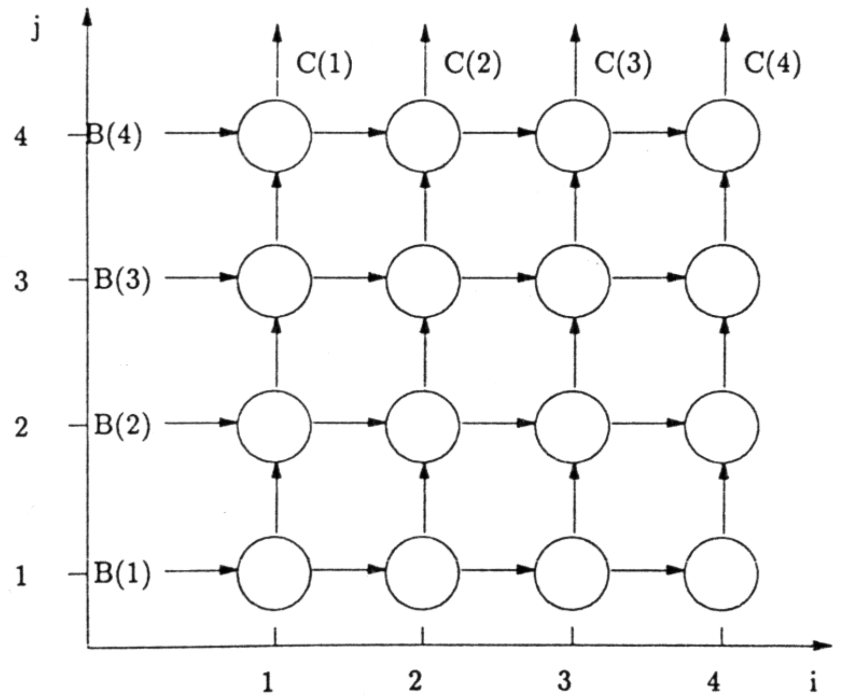
Example: Matrix-Vector Multiplication (6/7)

- Localized DG
  - Use transmittent data to replace broadcast data
  - A locally recursive algorithm is an algorithm whose corresponding DG has only local dependencies
    - The length of each dependency arc is independent of the problem size

Example: Matrix-Vector Multiplication (7/7)

```
b(1,1)=B(1);
b(1,2)=B(2);
b(1,3)=B(3);
b(1,4)=B(4);

for(i=1;i<=4;i++)
{
        c(i,1)=0;
        for(j=1;j<=4;j++)
        {
                b(i+1, j)=b(i,j);
                c(i,j+1)=c(i,j)+A(i,j)*b(i, j);
        }
}
```

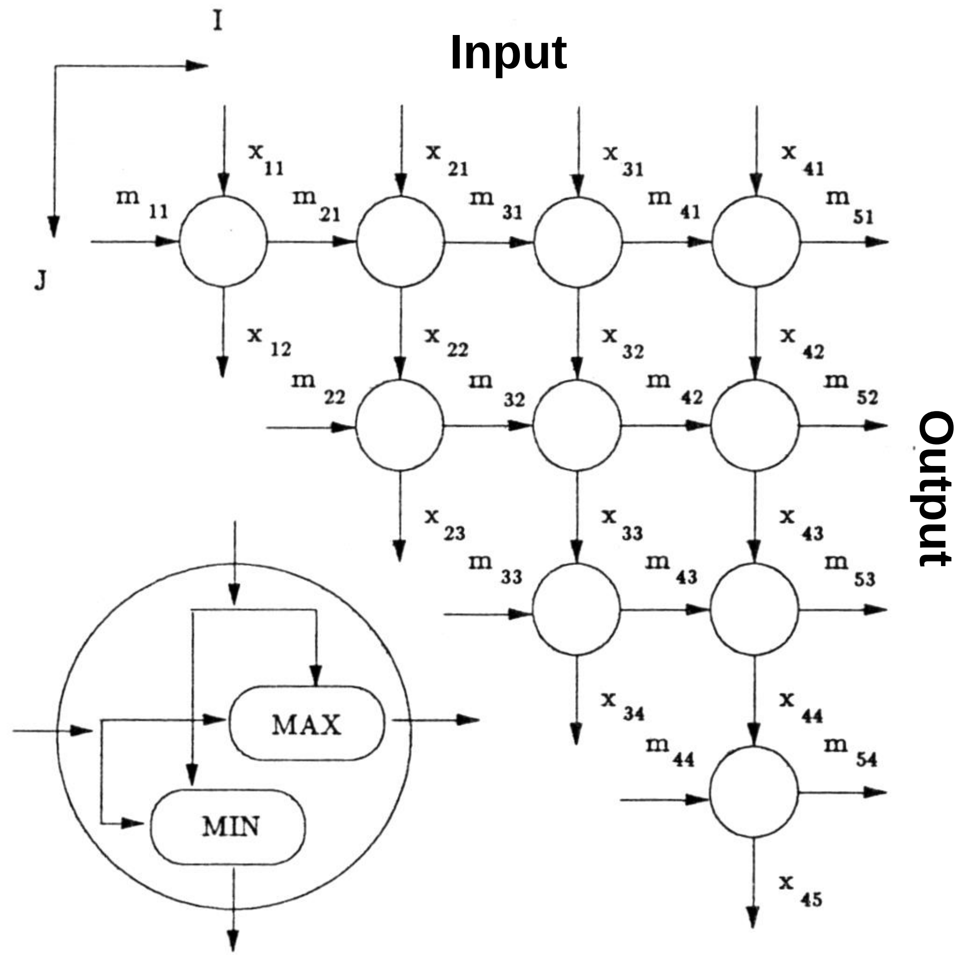


Example: Sorting

for(i=1;i<=N;i++)
for(j=1;j<=i;j++)
{
    m(i+1, j)=max(x(i,j), m(i,j));
    x(i, j+1)=min(x(i,j), m(i,j));
}

Initialization

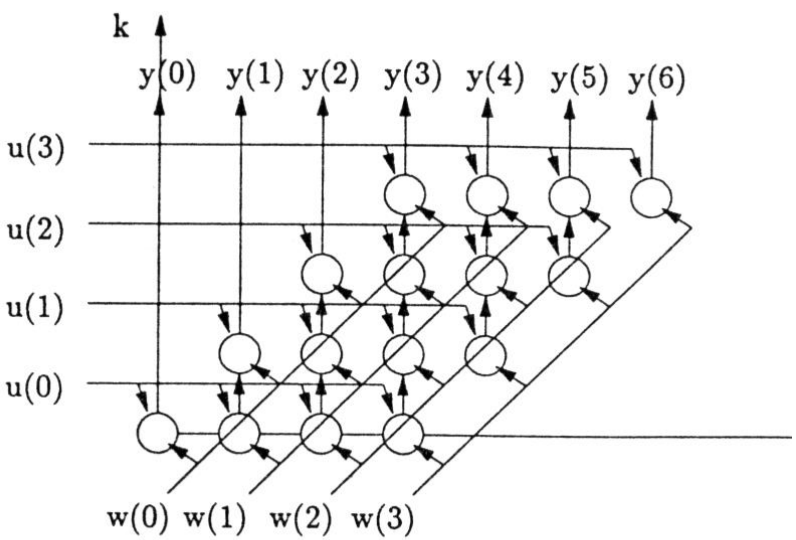
x(i,1)=x(i), the original sequence
m(i,i)= $-\infty$
Output of the sorter m(j)=m(N,j)

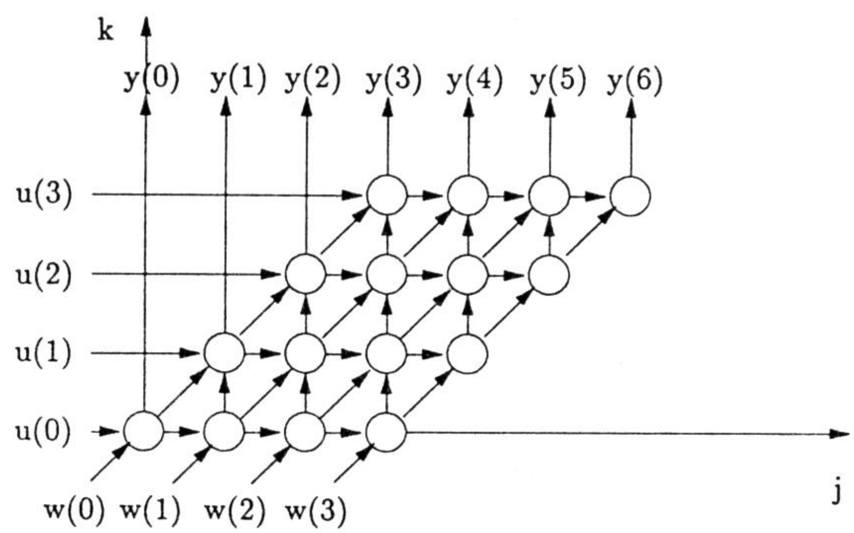


Example: Convolution

$$y_j = \sum_{k=0}^{3} u_k w_{j-k}$$

$$y_j^k = y_j^{k-1} + u_k \cdot w_{j-k}$$



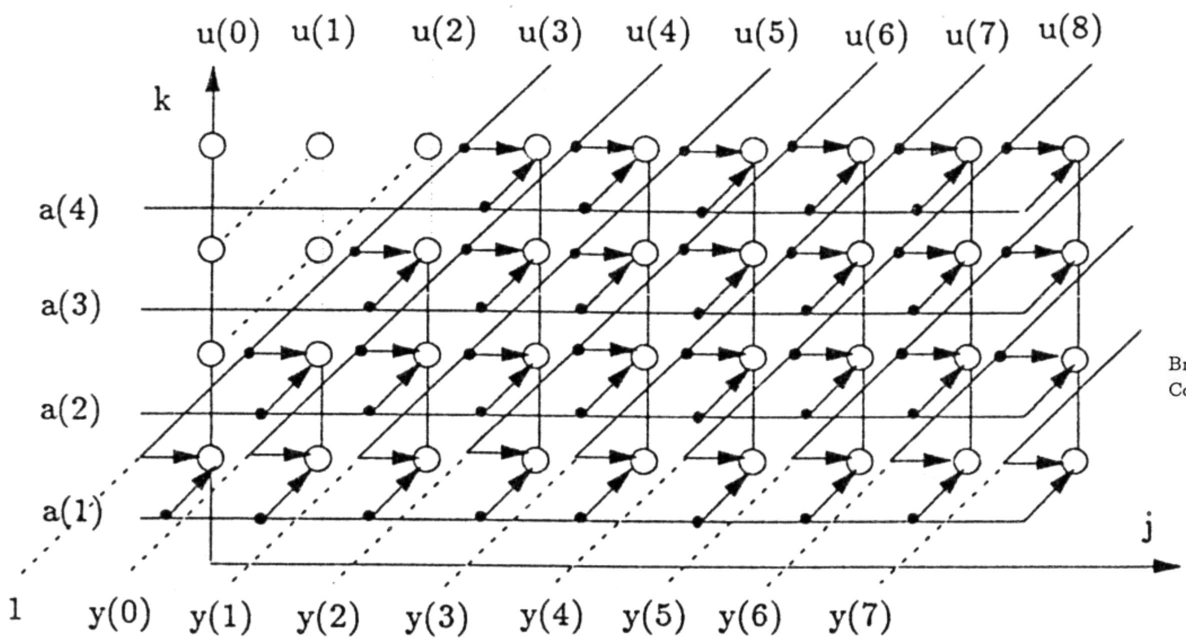


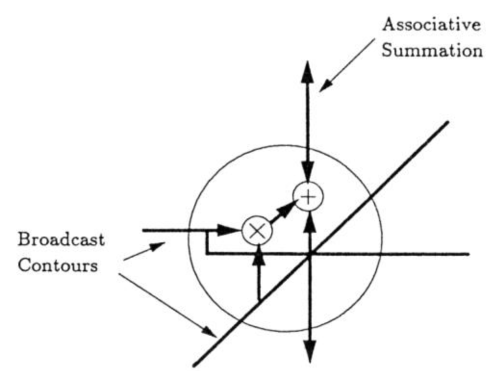
Localized DG

Example: Autoregressive Filter (AR Filter)

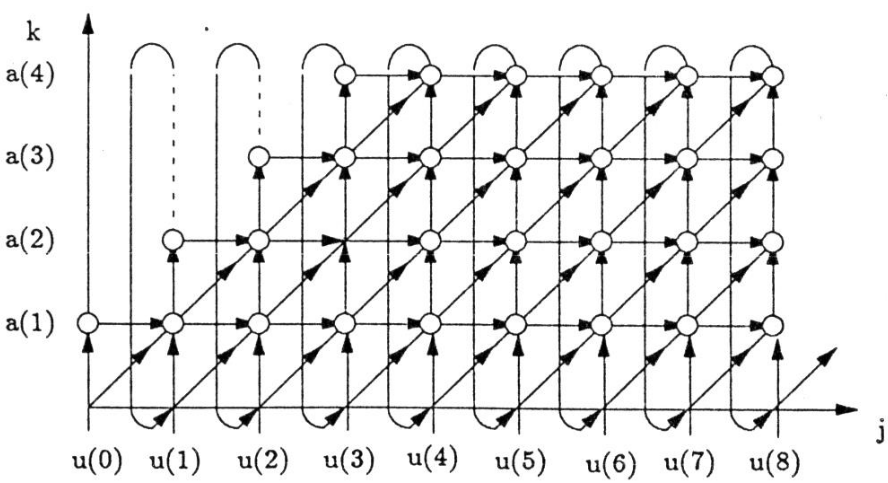
$$y(j) = \left[\sum_{k=1}^{N} a_k y(j-k) \right] + u(j)$$

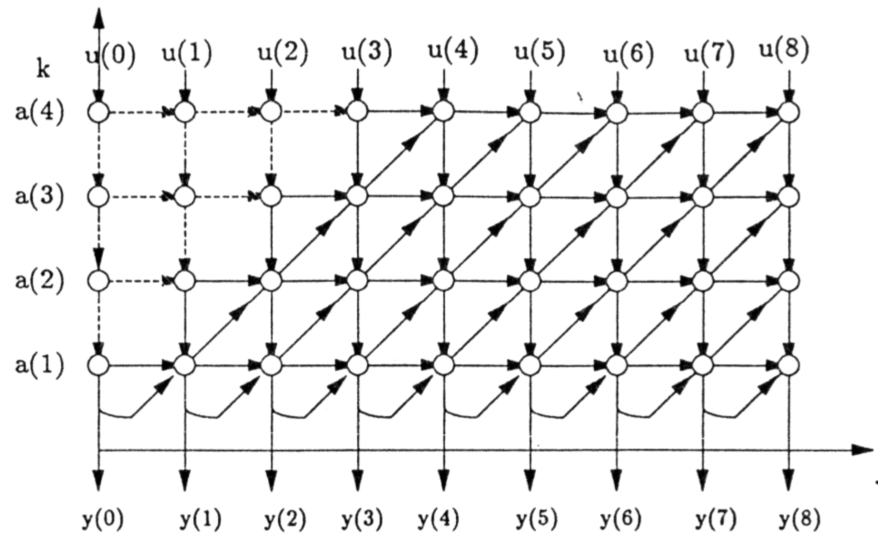
$$y(4) = u(4) + a_4 y(0) + a_3 y(1) + a_2 y(2) + a_1 y(3)$$





Example: Autoregressive Filter (AR Filter)

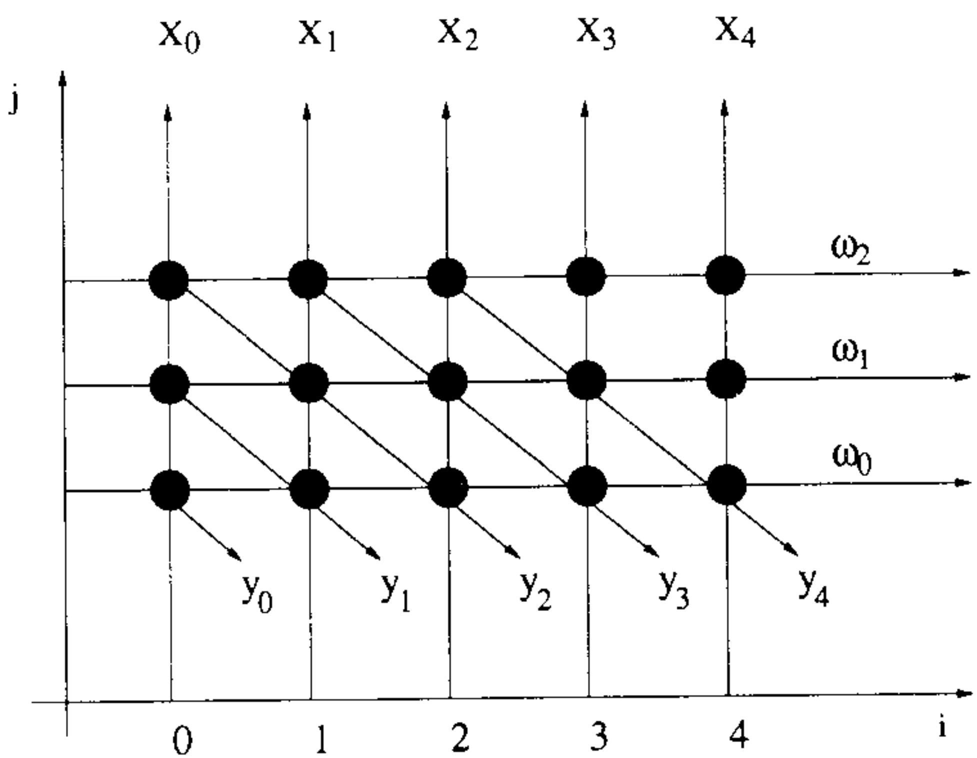




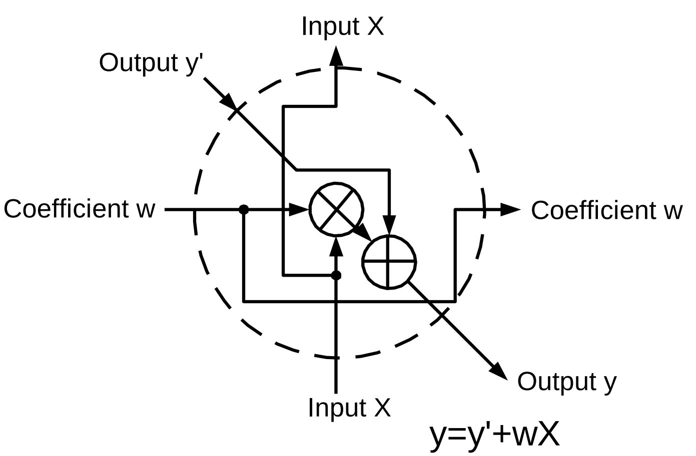
Localized DG

Example: FIR Filter

- $y(n)=w_0x(n)+w_1x(n-1)+w_2x(n-2)$



Input: edges in $[i\ j]^T=[0\ 1]^T$
Coefficient: edges in $[1\ 0]^T$
Output: edges in $[1\ -1]^T$



Step 2: Mapping to DFG (1/4)

- Projection vector (iteration vector) $\mathbf{d} = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$
  - Two nodes that are displaced by **d** or multiples of **d** are executed by the same processor
- Processor space vector $\mathbf{p}^T = \begin{pmatrix} p_1 & p_2 \end{pmatrix}$
  - Any node with index $\mathbf{I}^T$=(i,j) would be executed by processor

$$\mathbf{p}^T I = \begin{pmatrix} p_1 & p_2 \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix}$$

Spatial domain mapping vector

Step 2: Mapping to DFG (2/4)

Temporal domain mapping vector

- Scheduling vector $\mathbf{s}^T = \begin{pmatrix} s_1 & s_2 \end{pmatrix}$
  - Any node with index **I** would be executed at time $\mathbf{s}^T\mathbf{I}$
- Hardware utilization efficiency $HUE = 1/|\mathbf{s}^T\mathbf{d}|$
  - Two tasks executed by the same processor are spaced $|\mathbf{s}^T\mathbf{d}|$ time units apart
- Edge mapping
  - For an edge **e** in the DG, an edge $\mathbf{p}^T\mathbf{e}$ is introduced in the systolic array with $\mathbf{s}^T\mathbf{e}$ delays

Step 2: Mapping to DFG (3/4)

- Constraints
  - Processor space vector p and the project vector d must be orthogonal to each other

$$\mathbf{p}^T(I_A - I_B) = 0 \Rightarrow \mathbf{p}^T\mathbf{d} = 0$$

  - If A and B are mapped to the same processor, then they cannot be executed at the same time

$$\mathbf{s}^T I_A \neq \mathbf{s}^T I_B, \ i.e., \ \mathbf{s}^T\mathbf{d} \neq 0.$$

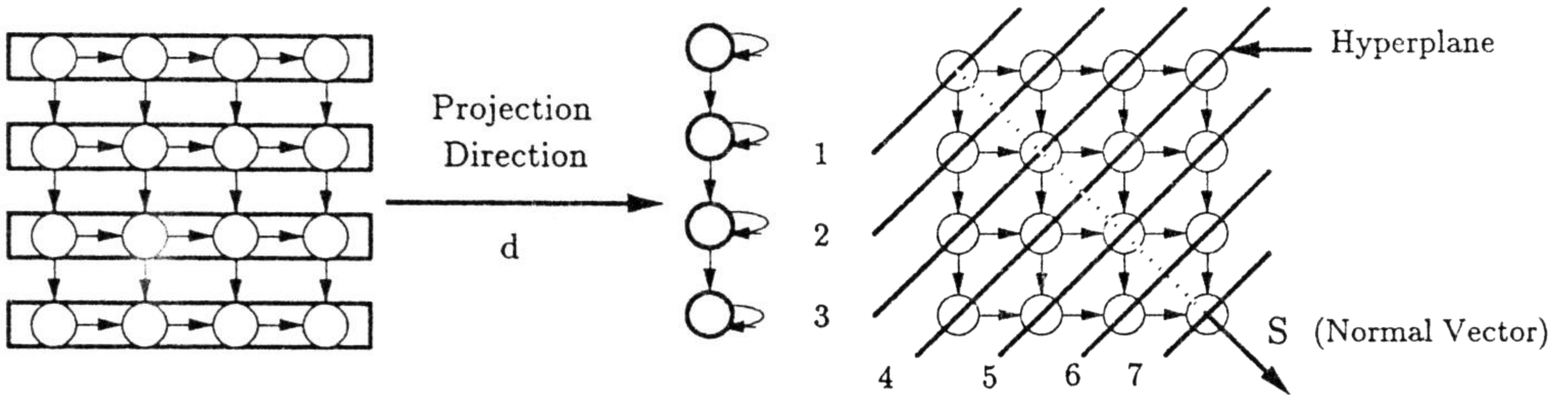
$$\mathbf{s}^T\mathbf{I}_B > \mathbf{s}^T\mathbf{I}_A \rightarrow \mathbf{s}^T\mathbf{d} > 0$$

Step 2: Mapping to DFG (4/4)

- Space-time transformation

$$\begin{pmatrix} i' \\ j' \\ t' \end{pmatrix} = T \begin{pmatrix} i \\ j \\ t \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ & \mathbf{p}^T & 0 \\ & \mathbf{s}^T & 0 \end{pmatrix} \begin{pmatrix} i \\ j \\ t \end{pmatrix}$$

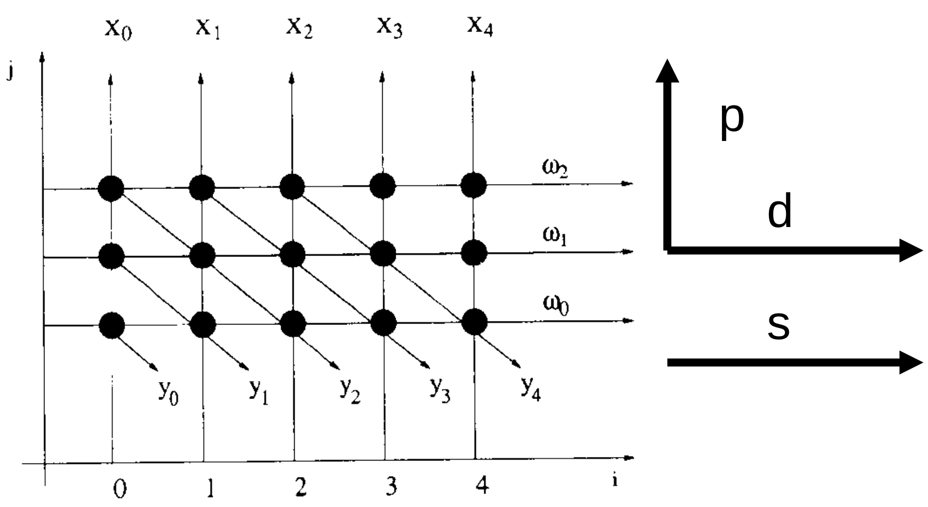
$i' = 0$, $t = 0$



FIR Systolic Arrays – B1 (1/5)

- Design B1 (broadcast input, move result, weight stay)

$$\mathbf{d} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{p}^T = \begin{pmatrix} 0 & 1 \end{pmatrix}, \quad \mathbf{s}^T = \begin{pmatrix} 1 & 0 \end{pmatrix}$$



FIR Systolic Arrays – B1 (2/5)

- Node $I^T(i,j)$ is mapped to processor

$$\mathbf{p}^T I = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix} = j$$

- Node $I^T(I,j)$ is executed at time

$$\mathbf{s}^T I = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix} = i$$

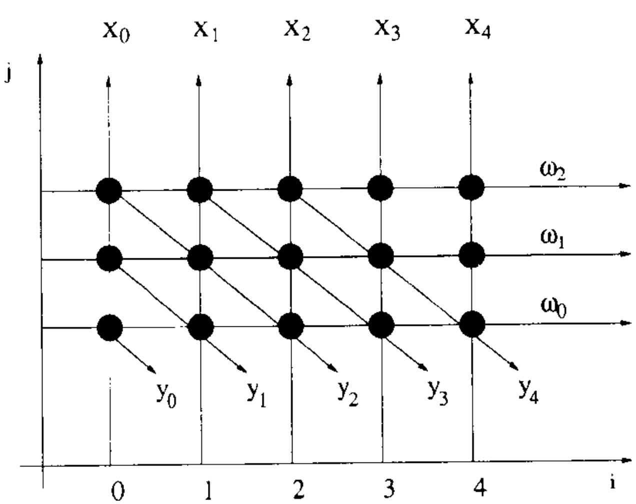
- HUE

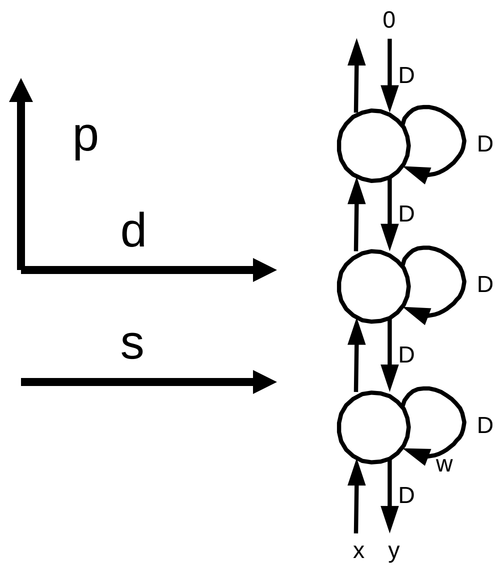
$$\mathbf{s}^T \mathbf{d} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1 \qquad HUE = \frac{1}{|\mathbf{s}^T \mathbf{d}|} = 1$$

FIR Systolic Arrays – B1 (3/5)

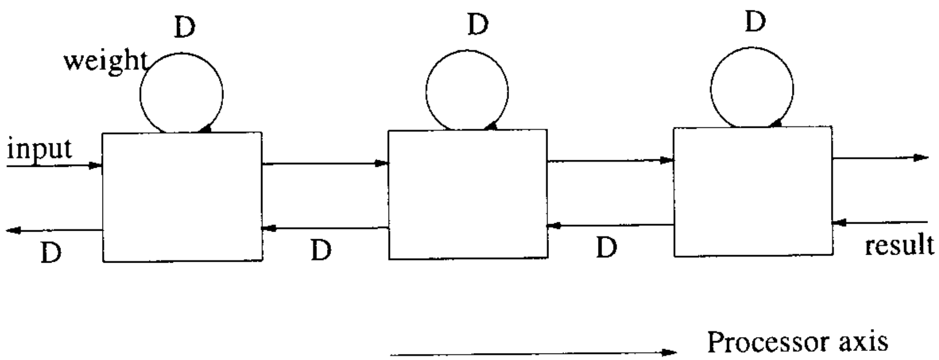
- Edge mapping

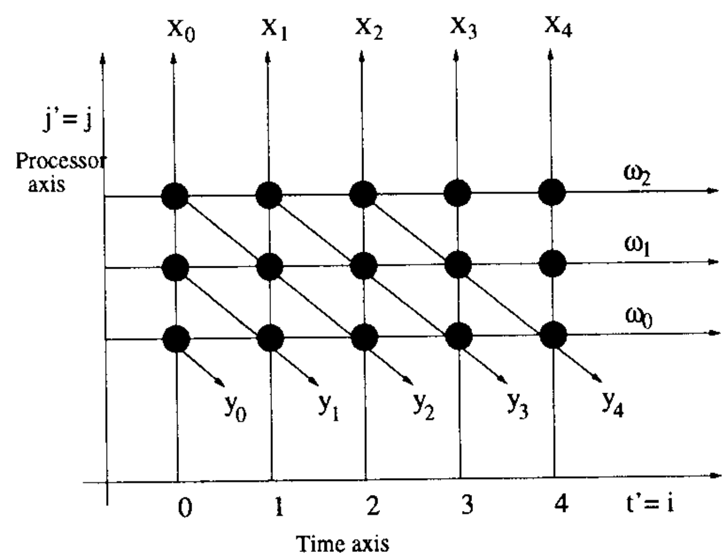
e	$p^T e$	$s^T e$
wt(1 0)	0	1
i/p(0 1)	1	0
result(1 −1)	−1	1

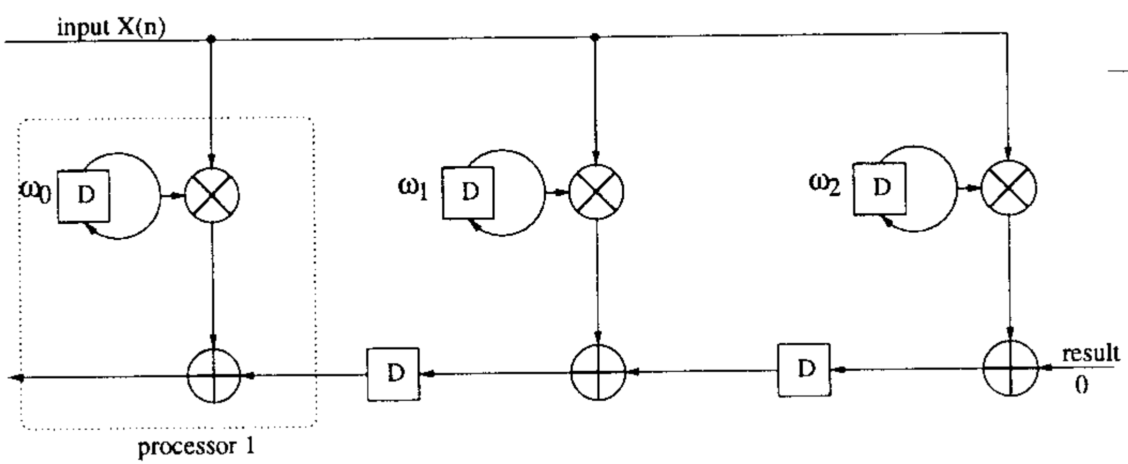




FIR Systolic Arrays – B1 (4/5)





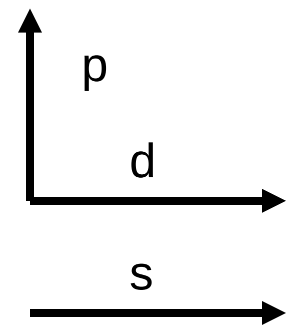


FIR Systolic Arrays – B1 (5/5)

- Constraints

$$p^T d = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 0$$

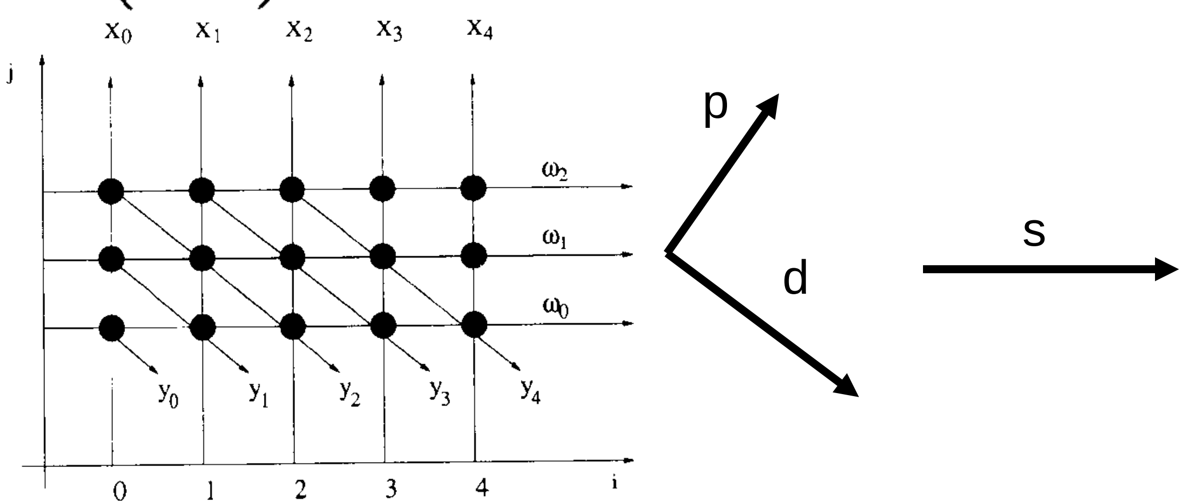
$$s^T d = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1 \neq 0$$



FIR Systolic Arrays – B2 (1/4)

- Design B2 (broadcast input, move weight, result stay)

$$\mathbf{d} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad \mathbf{p}^T = \begin{pmatrix} 1 & 1 \end{pmatrix}, \quad \mathbf{s}^T = \begin{pmatrix} 1 & 0 \end{pmatrix}$$



FIR Systolic Arrays – B2 (2/4)

- Space-time mapping

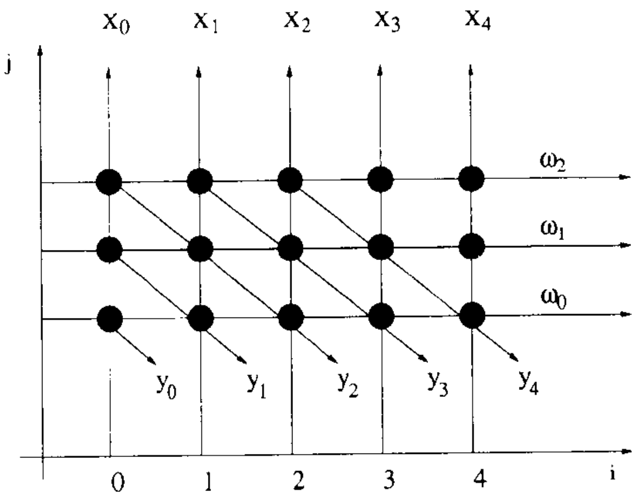
$$j' = \mathbf{p}^T \begin{pmatrix} i \\ j \end{pmatrix} = i + j$$

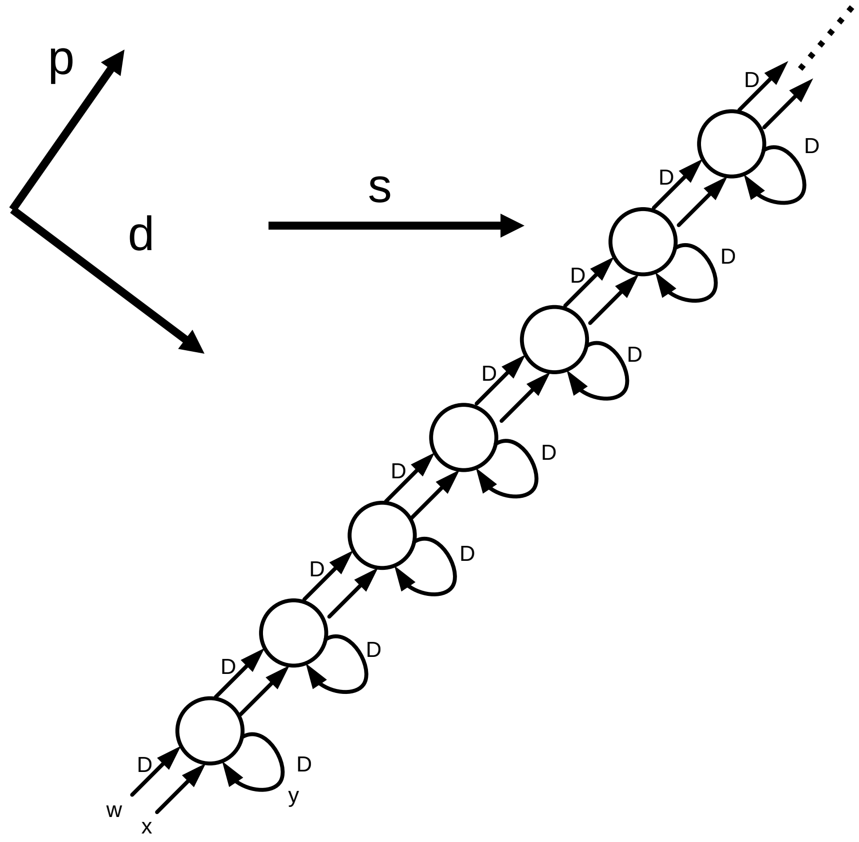
$$t' = \mathbf{s}^T \begin{pmatrix} i \\ j \end{pmatrix} = i.$$

$$\mathbf{s}^T \mathbf{d} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = 1$$

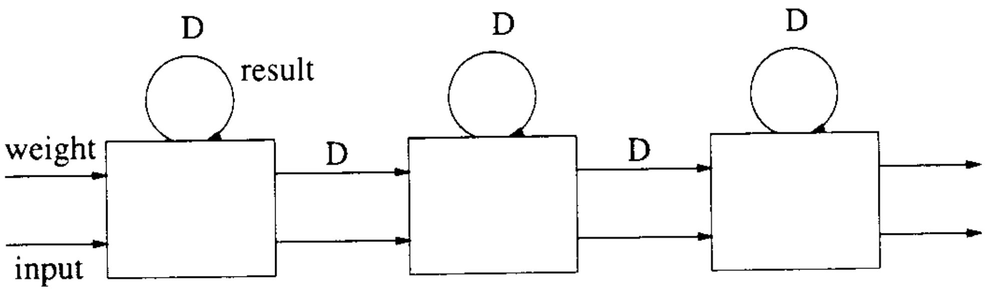
$\mathbf{e}$	$\mathbf{p}^T\mathbf{e}$	$\mathbf{s}^T\mathbf{e}$
wt(1, 0)	1	1
i/p(0, 1)	1	0
result(1, −1)	0	1

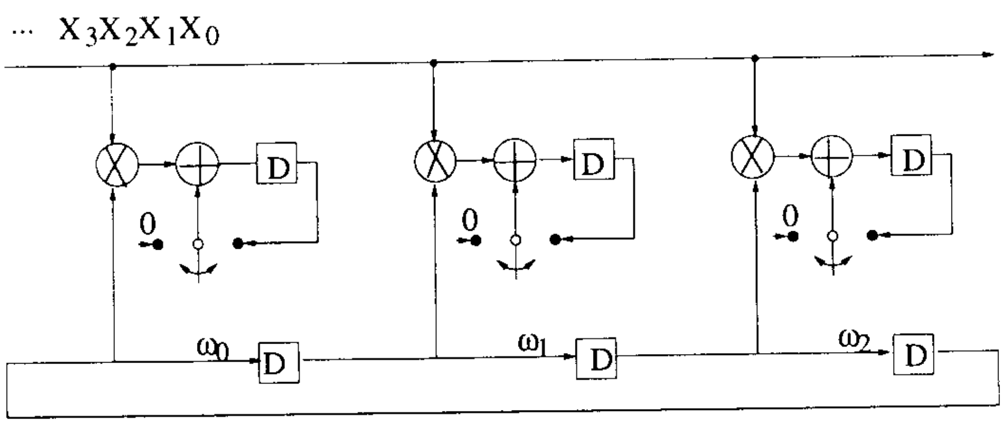
FIR Systolic Arrays – B2 (3/4)

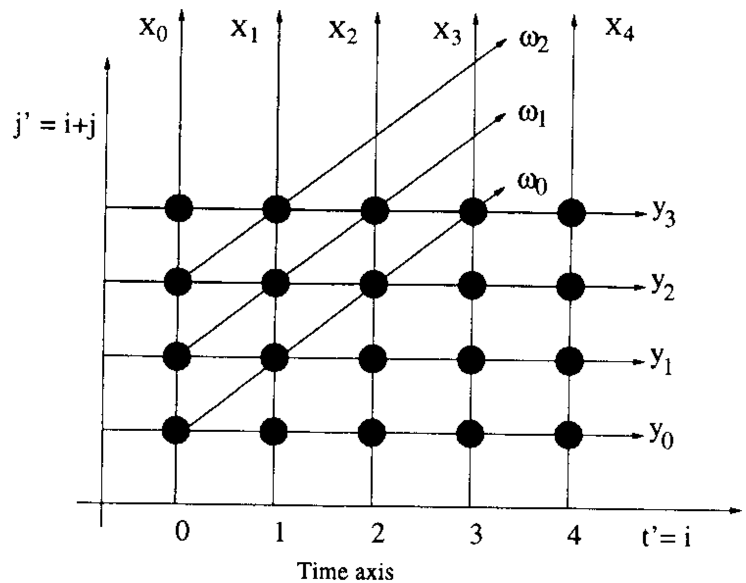




FIR Systolic Arrays – B2 (4/4)



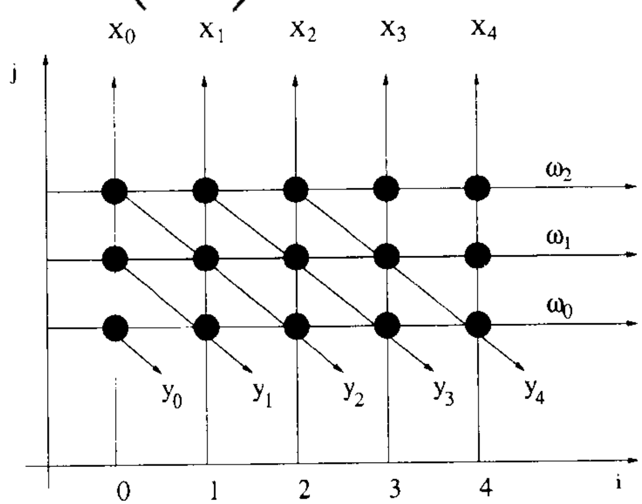


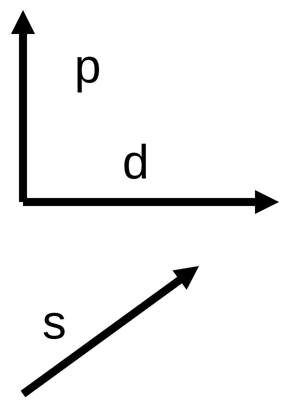


FIR Systolic Arrays – F (1/3)

- Design F (fan-in results, move inputs, weight stay)

$$\mathbf{d} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{p}^T = \begin{pmatrix} 0 & 1 \end{pmatrix}, \quad \mathbf{s}^T = \begin{pmatrix} 1 & 1 \end{pmatrix}$$





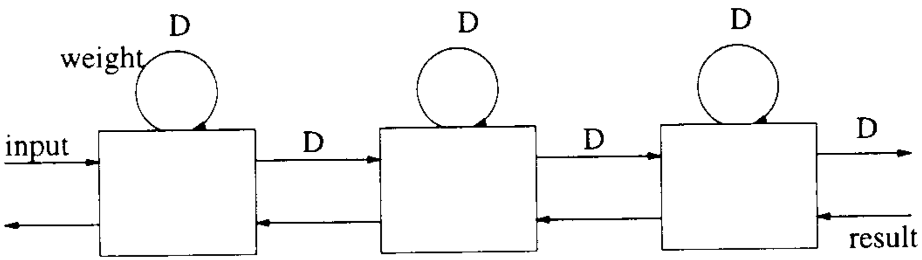
FIR Systolic Arrays – F (2/3)

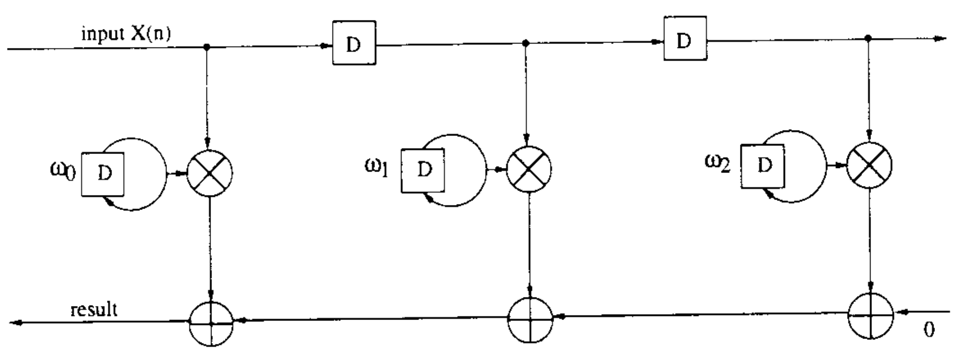
- Space-time mapping

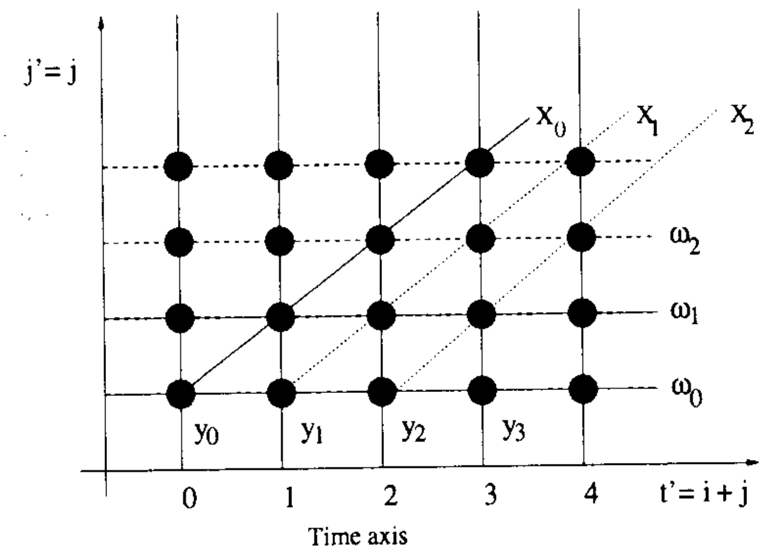
$$j' = \mathbf{p}^T \begin{pmatrix} i \\ j \end{pmatrix} = j, \quad t' = \mathbf{s}^T \begin{pmatrix} i \\ j \end{pmatrix} = i + j$$

$\mathbf{e}$	$\mathbf{p}^T\mathbf{e}$	$\mathbf{s}^T\mathbf{e}$
wt(1, 0)	0	1
i/p(0, 1)	1	1
result(1, −1)	-1	0

FIR Systolic Arrays – F (3/3)



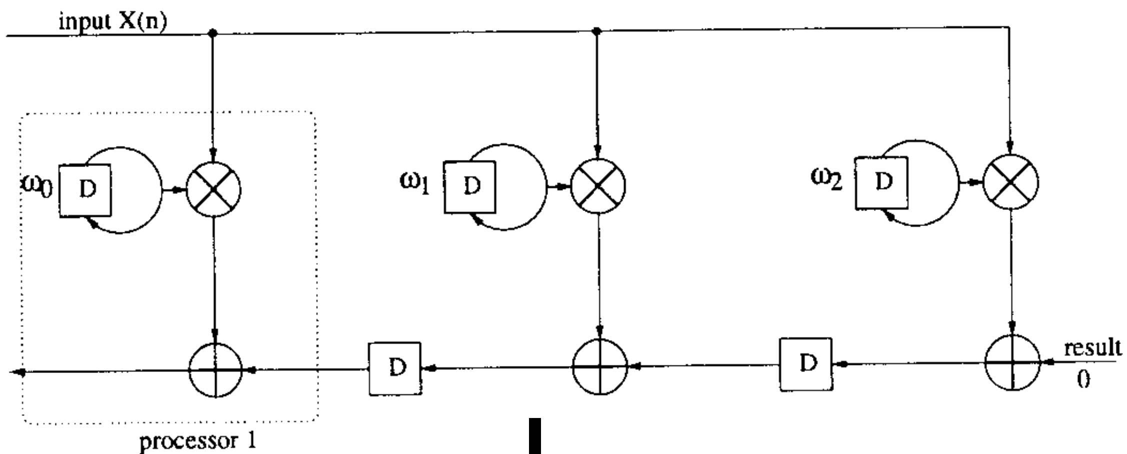




Relationship to Other Transformations (1/2)

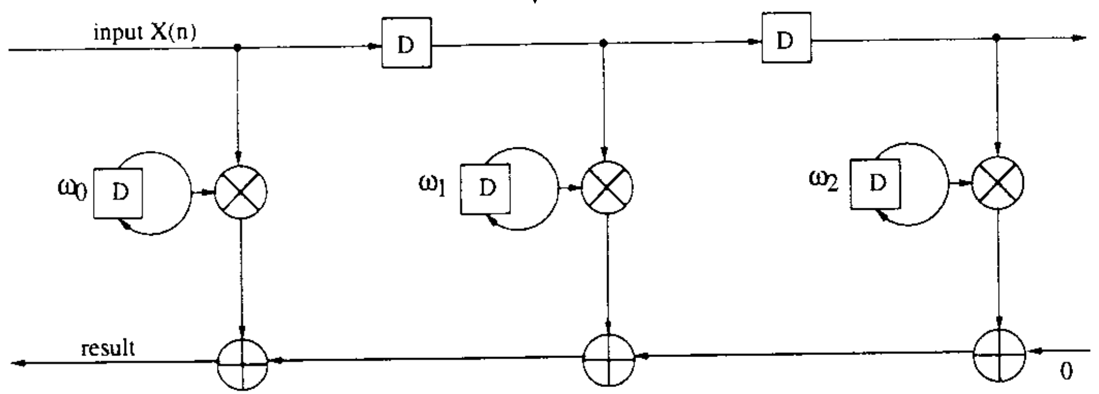
- Systolic array architectures with same project vector and processor space vector, but different scheduling vectors can be derived from other transformations
  - Edge reversal, associativity, slow-down, retiming, and pipelining

Relationship to Other Transformations (2/2)



B1

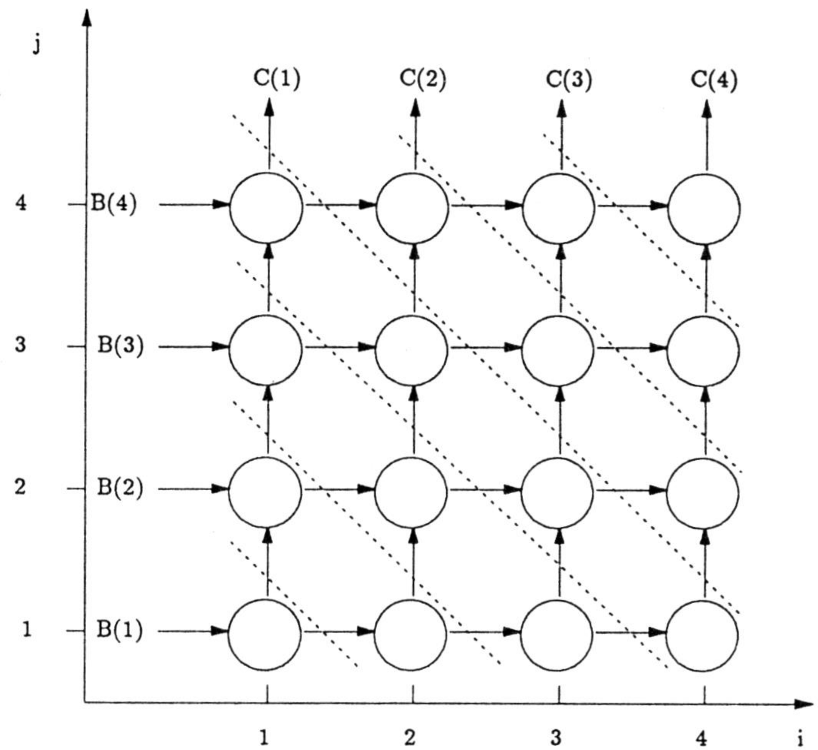
Cutset retiming

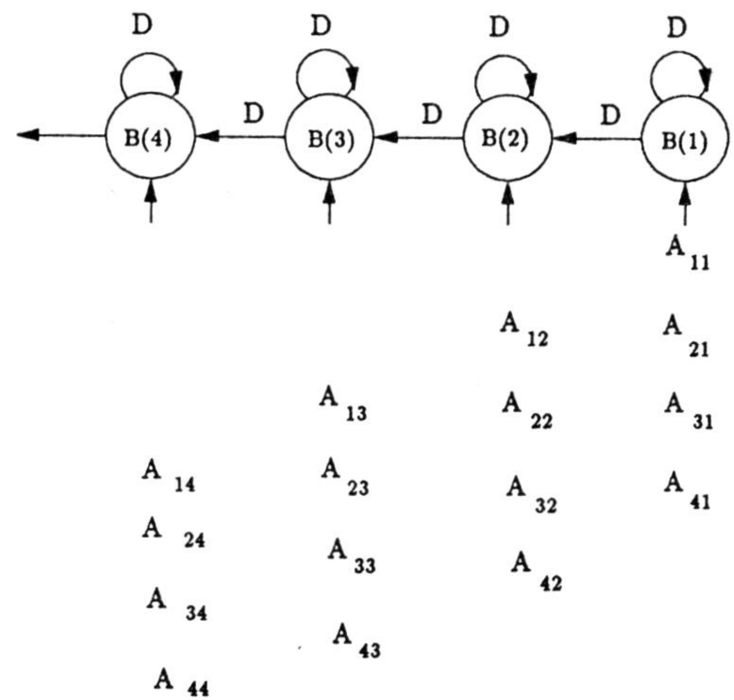


F

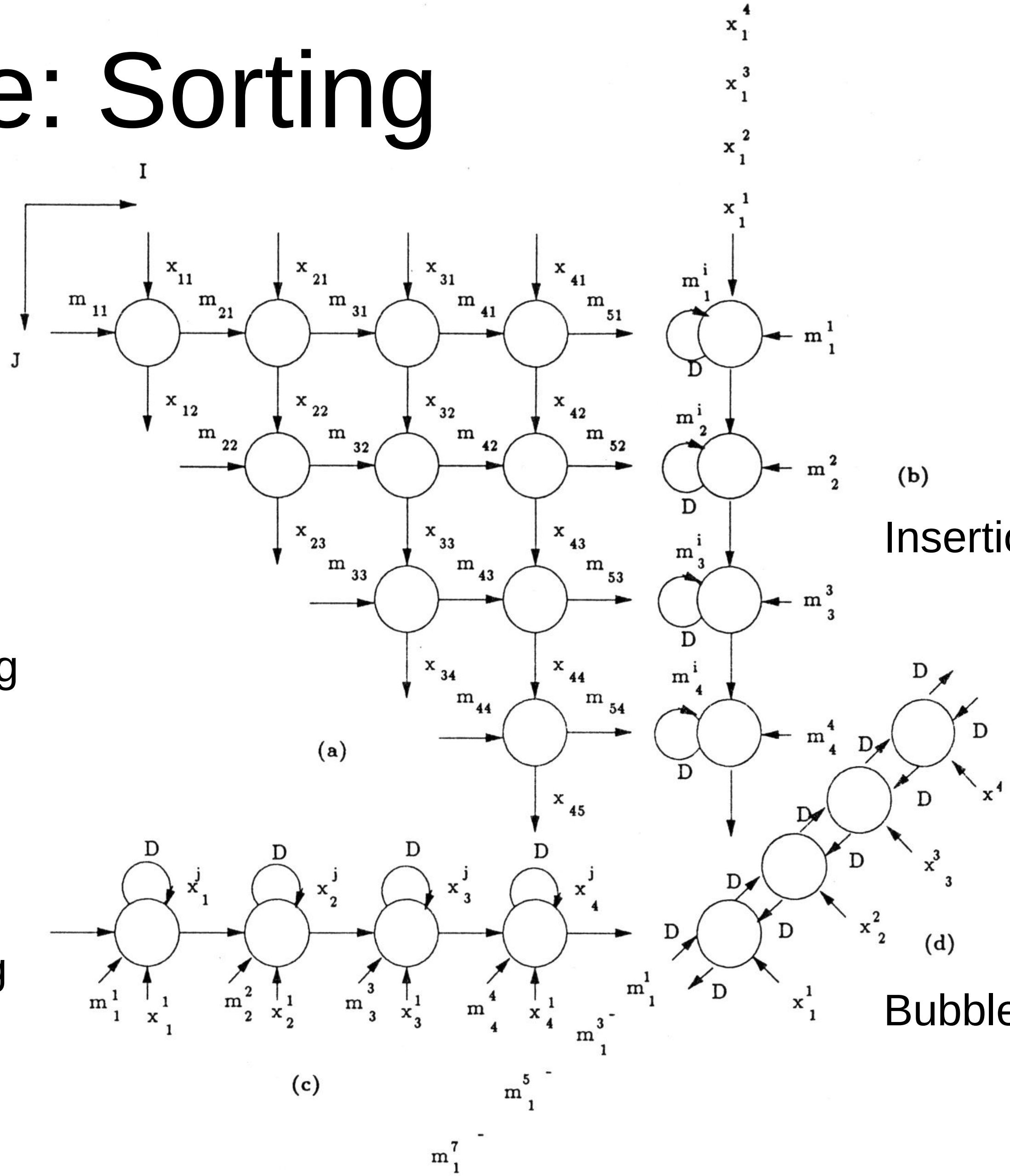
Example: Matrix-Vector Multiplication

- $d^T=[1\ 0]^T$, $p^T=[0\ 1]^T$, $s^T=[1\ 1]^T$





Example: Sorting



Insertion Sorting

Default scheduling
d||s

Selection Sorting

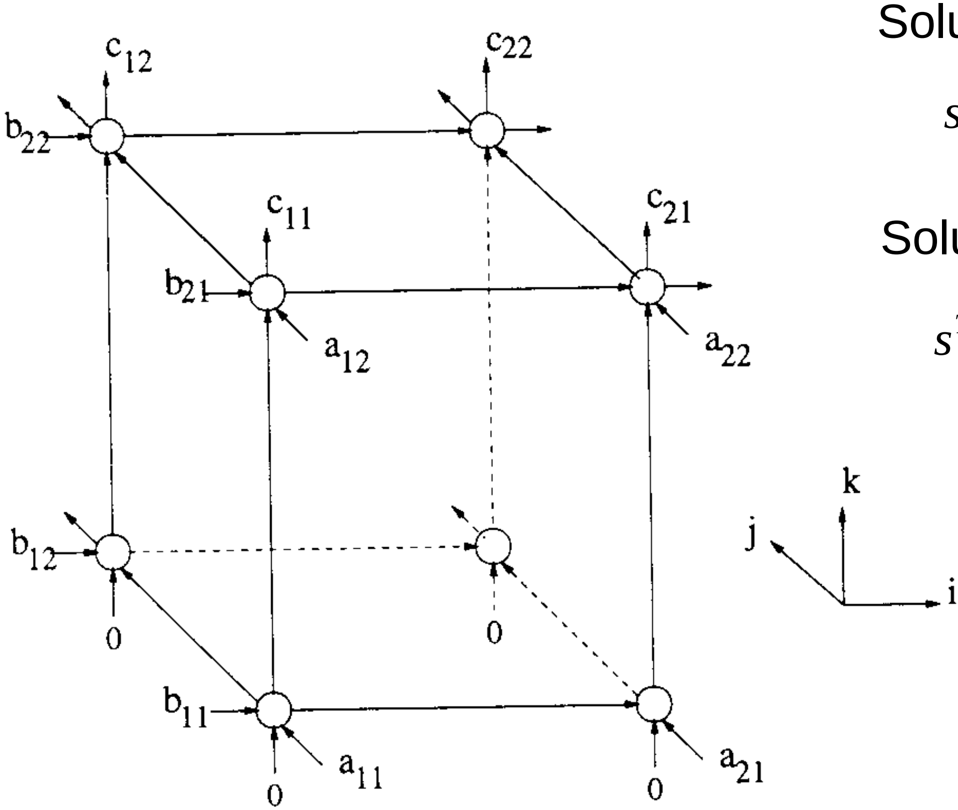
Bubble Sorting

Matrix-Matrix Multiplication (1/3)

- How about more-than-two dimensional DG?
- See this example: matrix-matrix multiplication
- **C**=**AB**, **C**, **A**, **B** are nxn matrices
- For n=2 $\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$

$$\begin{aligned} c_{11} &= a_{11}b_{11} + a_{12}b_{21} \\ c_{12} &= a_{11}b_{12} + a_{12}b_{22} \\ c_{21} &= a_{21}b_{11} + a_{22}b_{21} \\ c_{22} &= a_{21}b_{12} + a_{22}b_{22}. \end{aligned}$$

Matrix-Matrix Multiplication (2/3)



Solution 1:

$$s^T = (1,1,1),\ d = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},\ p^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

Solution 2:

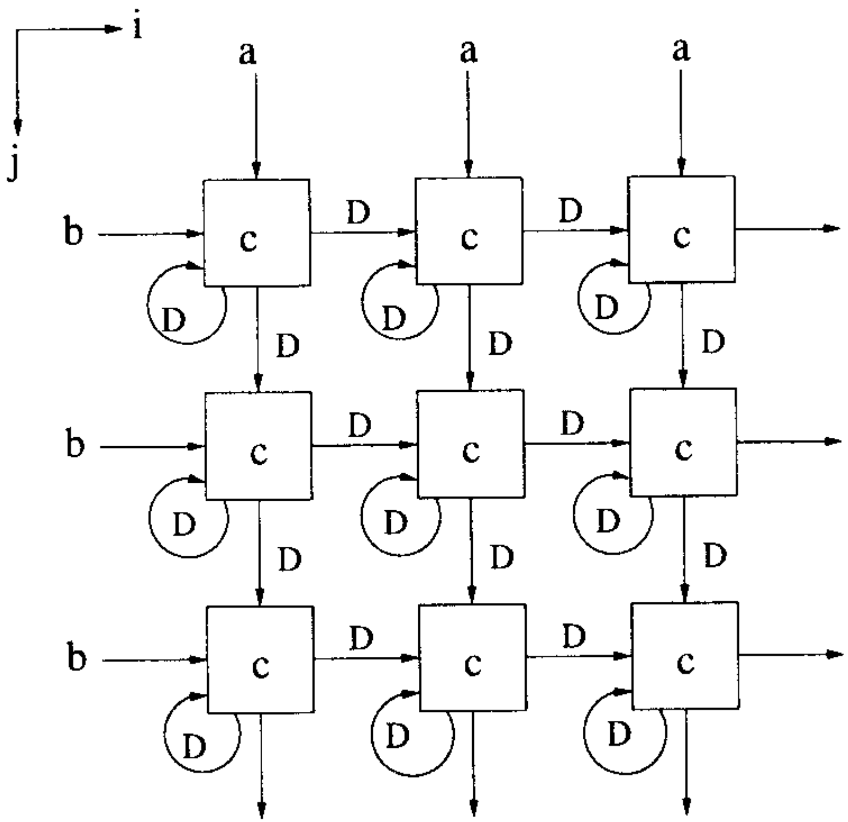
$$s^T = (1,1,1),\ d = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix},\ p^T = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

Edge mapping:

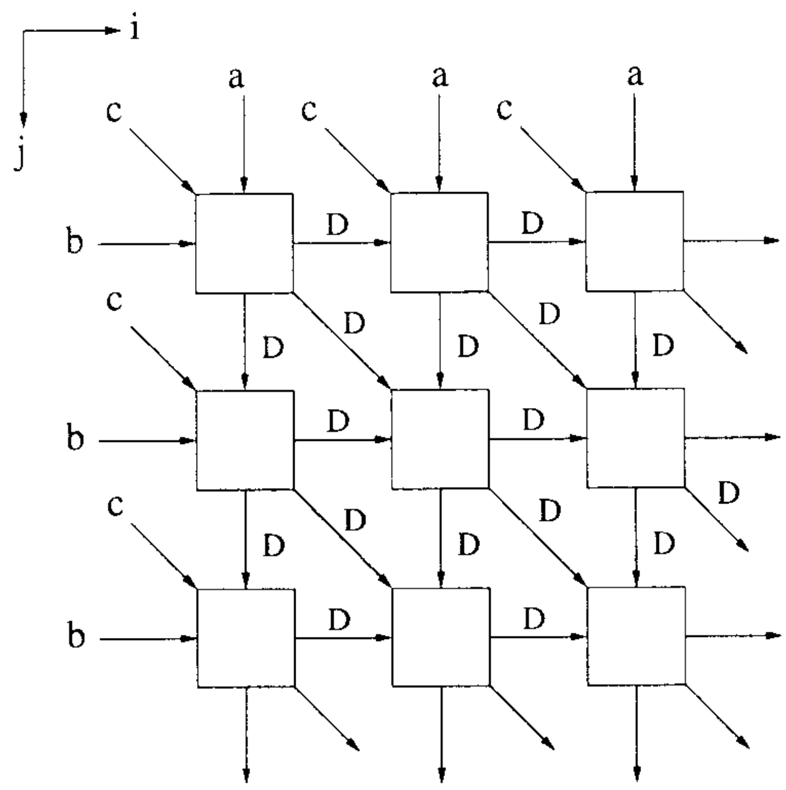
	Sol. 1		Sol. 2	
e	$\mathbf{P}^T\mathbf{e}$	$\mathbf{s}^T\mathbf{e}$	$\mathbf{P}^T\mathbf{e}$	$\mathbf{s}^T\mathbf{e}$
a(0, 1, 0)	(0, 1)	1	(0, 1)	1
b(1, 0, 0)	(1, 0)	1	(1, 0)	1
C(0, 0, 1)	(0, 0)	1	(1, 1)	1

Matrix-Matrix Multiplication (3/3)

Solution 1



Solution 2



Selection of Schedule Vector

- Choose the schedule vector **s** first, then **d** and **p** can be selected according to

$$p^T d = 0, s^T d \neq 0$$

- Procedure to get $\mathbf{s}^T$
  - Capture all the fundamental edges in reduced dependence graph (RDG) constructed by regular iteration algorithm (RIA) description
  - Construct the scheduling inequalities
  - Solve feasible $\mathbf{s}^T$ ($\mathbf{s}^T\mathbf{d}$=1 or $\mathbf{s}^T\mathbf{d}$=iteration bound is optimal)

Scheduling Inequalities (1/2)

- For the edge X→Y

$$X:\ I_x = \begin{pmatrix} i_x \\ j_x \end{pmatrix} \longrightarrow Y:\ I_y = \begin{pmatrix} i_y \\ j_y \end{pmatrix}$$

- The scheduling inequality is

$$S_y \geq S_x + T_x$$

  - S_x: scheduling time for note X
  - S_y: scheduling time for node Y
  - T_x: computation time of node X

Scheduling Inequalities (2/2)

- Linear scheduling

$$S_x = \mathbf{s}^T I_x = \begin{pmatrix} s_1 & s_2 \end{pmatrix} \begin{pmatrix} i_x \\ j_x \end{pmatrix}$$

$$S_y = \mathbf{s}^T I_y = \begin{pmatrix} s_1 & s_2 \end{pmatrix} \begin{pmatrix} i_y \\ j_y \end{pmatrix}$$

- Affine scheduling

$$S_x = \mathbf{s}^T I_x + \gamma_x = \begin{pmatrix} s_1 & s_2 \end{pmatrix} \begin{pmatrix} i_x \\ j_x \end{pmatrix} + \gamma_x$$

$$S_y = \mathbf{s}^T I_y + \gamma_y = \begin{pmatrix} s_1 & s_2 \end{pmatrix} \begin{pmatrix} i_y \\ j_y \end{pmatrix} + \gamma_y$$

- So the scheduling inequalities:

$$\mathbf{s}^T I_y + \gamma_y \geq \mathbf{s}^T I_x + \gamma_x + T_x \longrightarrow \mathbf{s}^T \mathbf{e}_{x-y} + \gamma_y - \gamma_x \geq T_x$$

RDG from RIA (1/2)

- For the FIR filter

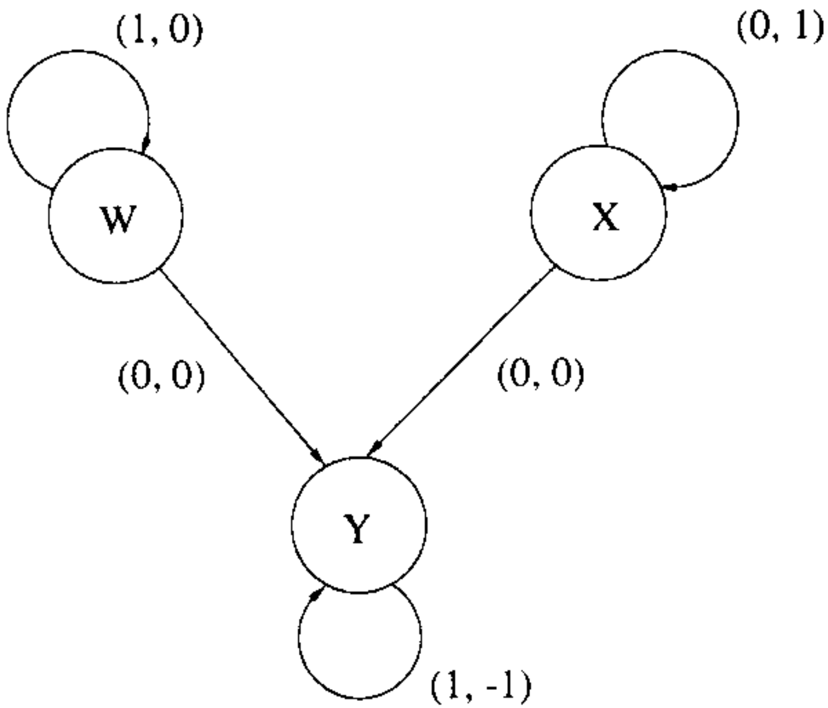
$$
\begin{aligned}
W(i+1,j) &= W(i,j) \\
X(i,j+1) &= X(i,j) \\
Y(i+1,j-1) &= Y(i,j) + W(i+1,j-1)X(i+1,j-1).
\end{aligned}
$$

- Standard output RIA form

$$
\begin{aligned}
W(i,j) &= W(i-1,j) \\
X(i,j) &= X(i,j-1) \\
Y(i,j) &= Y(i-1,j+1) + W(i,j)X(i,j)
\end{aligned}
$$

RDG from RIA (2/2)

- RDG



Scheduling Vector Selection

- Assume $T_{mult} = 5, \quad T_{add} = 2, \quad T_{com} = 1.$ $\mathbf{s} = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$

$$W \to Y: \quad \mathbf{e} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \gamma_y - \gamma_w \geq 0$$

$$X \to X: \quad \mathbf{e} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad s_2 + \gamma_x - \gamma_x \geq 1$$

$$W \to W: \quad \mathbf{e} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad s_1 + \gamma_w - \gamma_w \geq 1$$

$$X \to Y: \quad \mathbf{e} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \gamma_y - \gamma_x \geq 0$$

$$Y \to Y: \quad \mathbf{e} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad s_1 - s_2 + \gamma_y - \gamma_y \geq 5 + 2 + 1.$$

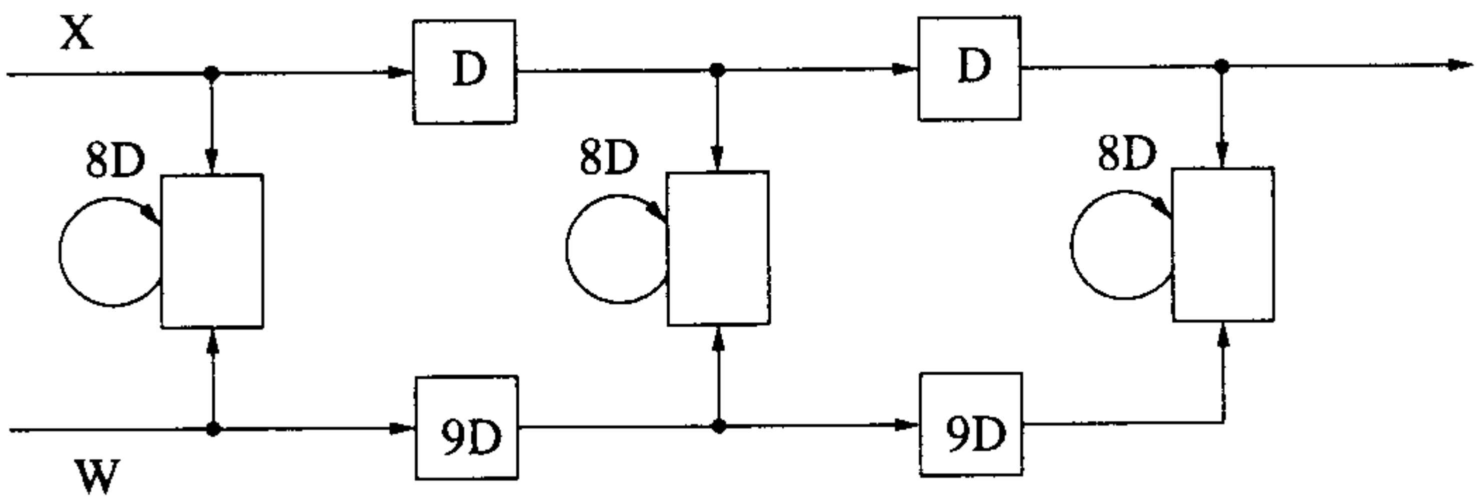
$\longrightarrow s_1 \geq 1, \quad s_2 \geq 1, \quad s_1 - s_2 \geq 8.$

One solution: $\mathbf{s}^T = (9, 1)$

Set $\gamma_x = \gamma_y = \gamma_w = 0$

Systolic Architecture Mapping

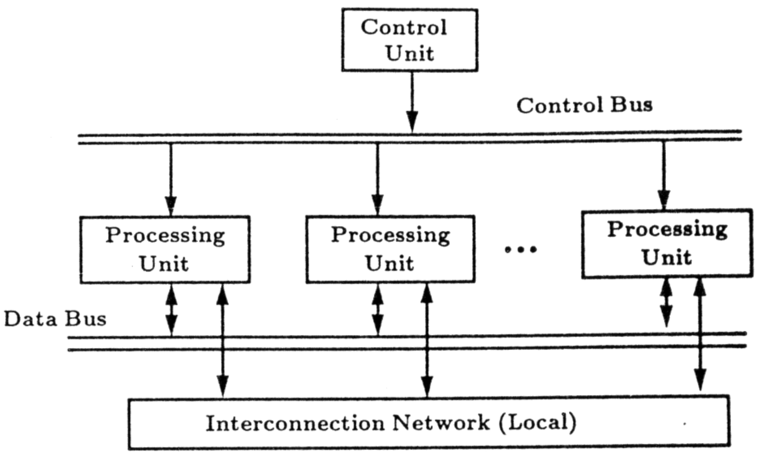
- Then also choose **d**=(1,-1), $\mathbf{p}^T$=(1, 1)



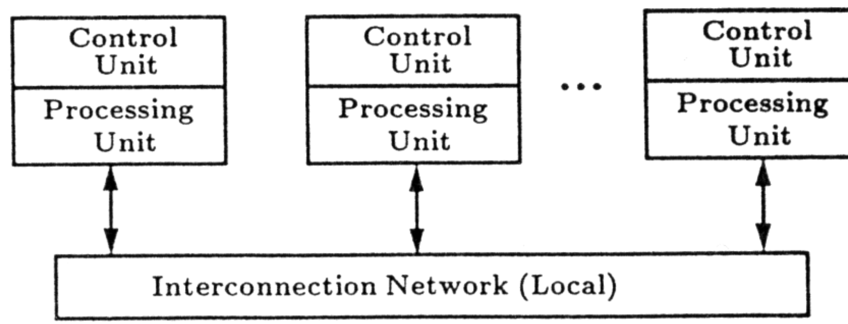
Stage 3: VLSI Array Design (1/2)

- Lots of choices
  - Single Instruction Multiple Data (SIMD) stream array
  - Systolic array
  - Wavefront array
  - SFG array

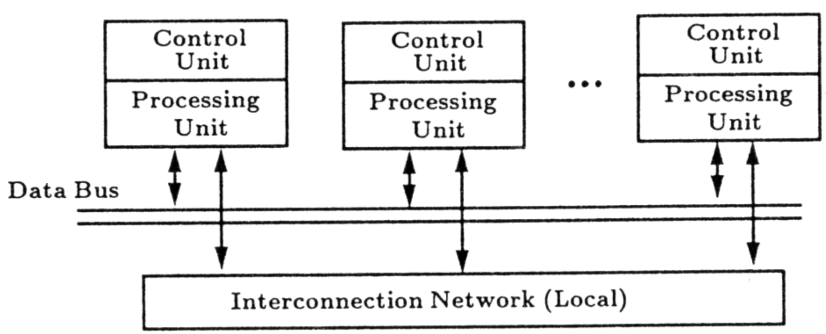
Stage 3: VLSI Array Design (2/2)



SIMD Array



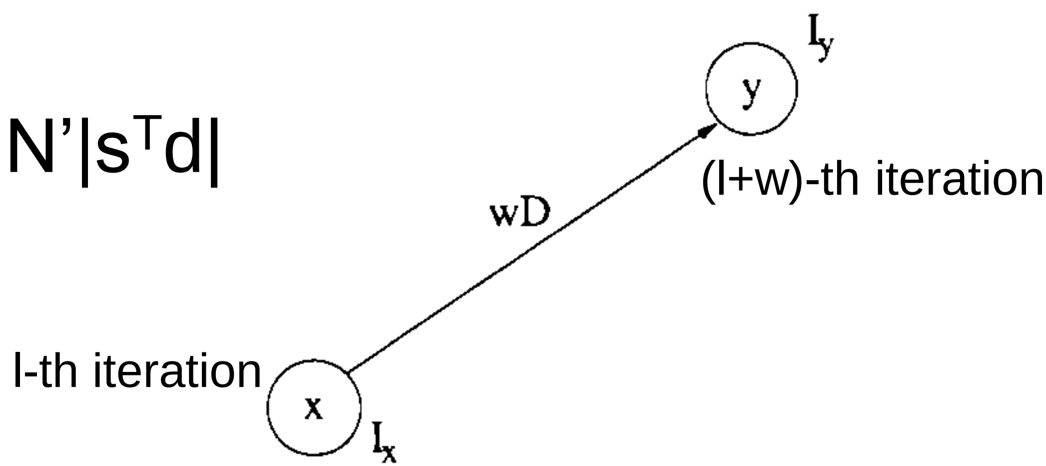
Systolic Array



DFG Array

Multiprojection

- Systolic Design for Space Representations with Delays
  - Can be used for multilevel systolic mapping
  - Define N': number of nodes mapped to a processor
  - Iteration period: $N'|s^T d|$



Scheduling Inequality and Systolic Transformation

- Scheduling inequality

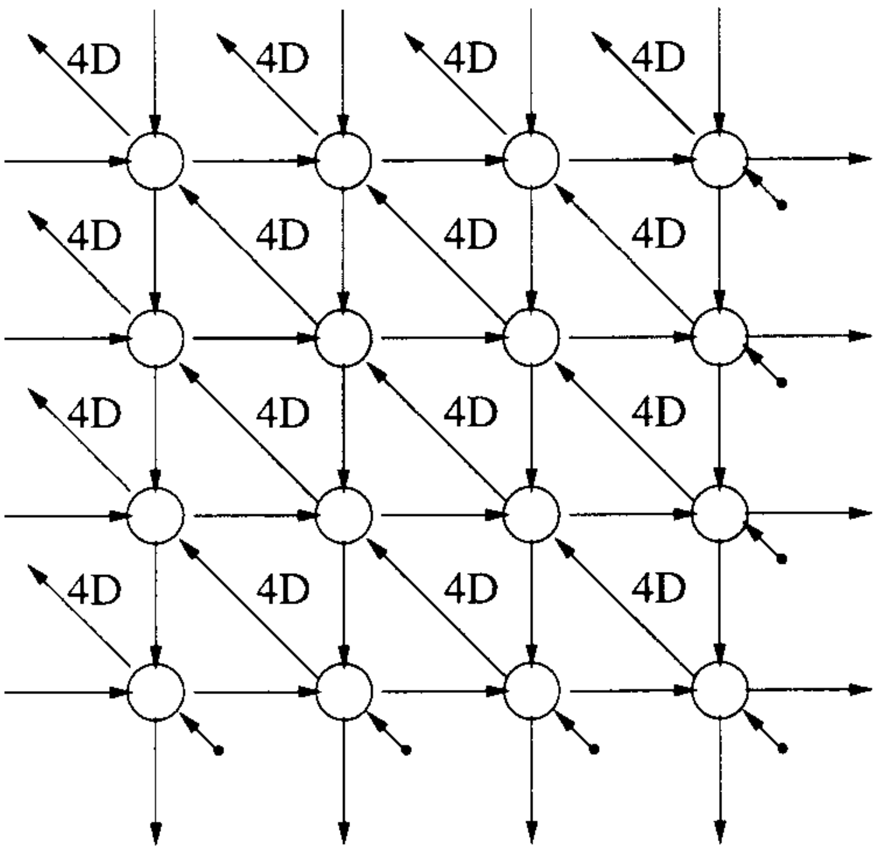
$$\mathbf{s}^T I_y + (l + w)N'|\mathbf{s}^T\mathbf{d}| \geq \mathbf{s}^T I_x + lN'|\mathbf{s}^T\mathbf{d}| + T_x$$

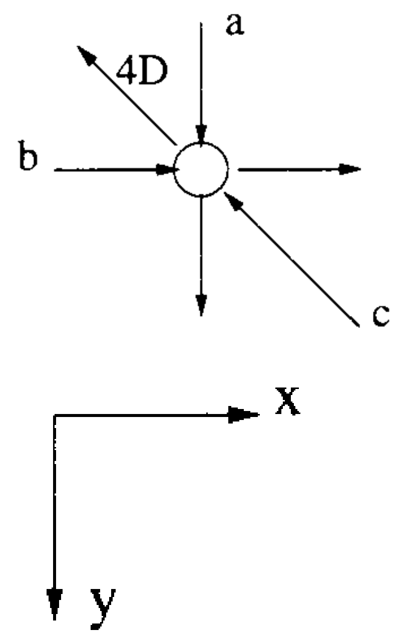
$$\mathbf{s}^T\mathbf{e} + wN'|\mathbf{s}^T\mathbf{d}| \geq T_x$$

- All the mapping equations are the same except for the edge delay mapping

$$s^T e \Rightarrow s^T e + wN'|s^T d|$$

Example of DG with Delays (1/3)





Example of DG with Delays (2/3)

$$\mathbf{e_a} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} s_1 & s_2 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} + 0 \cdot N' \left| S^T \mathbf{d} \right| \geq 1 \Rightarrow s_2 \geq 1$$

$$\mathbf{e_b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad s_1 \geq 1$$

$$\mathbf{e_c} = \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \quad -s_1 - s_2 + 4N'(s_1 d_1 + s_2 d_2) \geq 1.$$

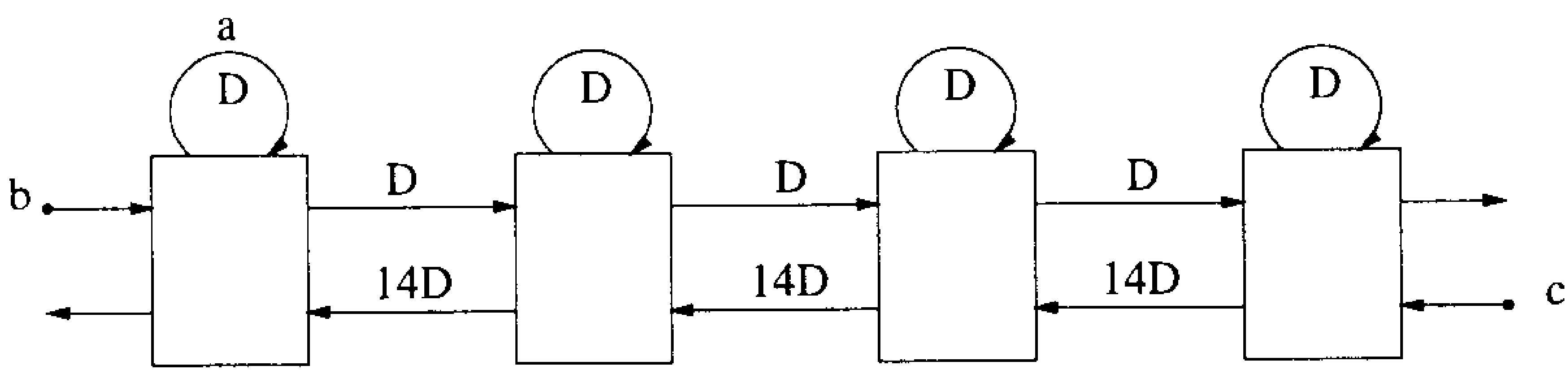
Assume $\mathbf{d} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ → N'=4

s^T=(1 1), p^T=(1 0)

$\mathbf{e}$	$\mathbf{p}^T\mathbf{e}$	$\mathbf{s}^T\mathbf{e} + wN'$ $\mathbf{s}^T\mathbf{d}$
a(0, 1)	0	1
b(1, 0)	1	1
c(−1, −1)	−1	14

Example of DG with Delays (3/3)

- 1-D systolic array



Remark

- The performance of the resulting array is affected by
  - The choice of a particular DG for an algorithm
  - The direction of the projection and the schedule vectors