Arrays, Records and Pointers

Instructors:
Md Nazrul Islam Mondal
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology
Rajshahi-6204

Outline

- Multidimensional Array
- Pointers: Pointer Array
- Records: Record Structure
- Representation of Records in Memory: Parallel Arrays
- Matrices
- Sparse Matrices

Multidimensional Array

- Two dimensional Array:

$$A_{J,K} \text{ or } A[J,K]$$

$$
\text{Rows} \quad
\begin{array}{c c}
 & \text{Columns} \\
 & \begin{array}{cccc} \;\;1\;\; & \quad\;\;2\;\;\quad & \quad\;\;3\;\;\quad & \;\;4\;\; \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \end{array} &
\begin{bmatrix}
A[1,1] & A[1,2] & A[1,3] & A[1,4] \\
A[2,1] & A[2,2] & A[2,3] & A[2,4] \\
A[3,1] & A[3,2] & A[3,3] & A[3,4]
\end{bmatrix}
\end{array}
$$

Fig. 4.8 *Two-Dimensional 3 × 4 Array A*

Multidimensional Array

- Representation of Two dimensional Array in memory:

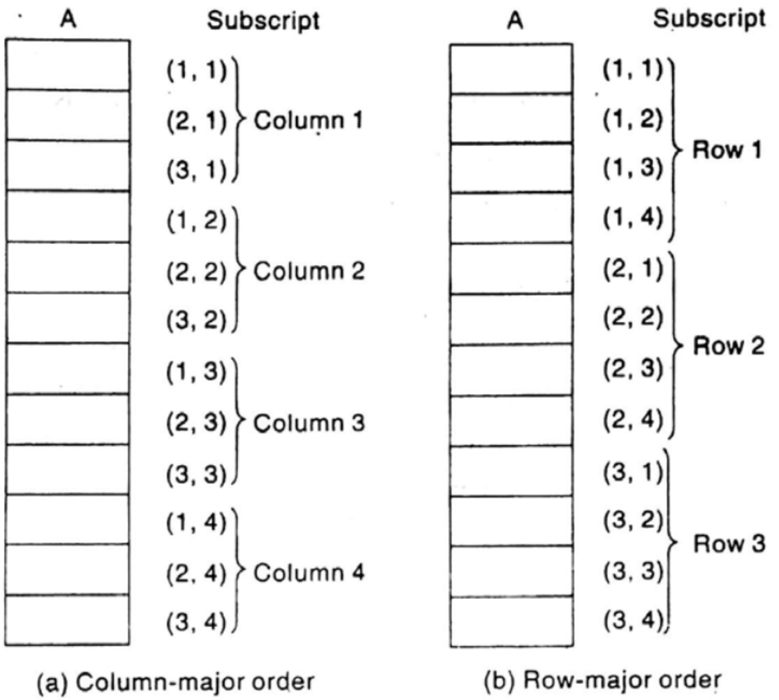


Fig. 4.10

Multidimensional Array

- General Multidimensional Arrays:

  $B_{K1\,,k2,k3,\ldots,kn}$ or $B[K_1, K_2, \ldots, K_N$

Multidimensional Array

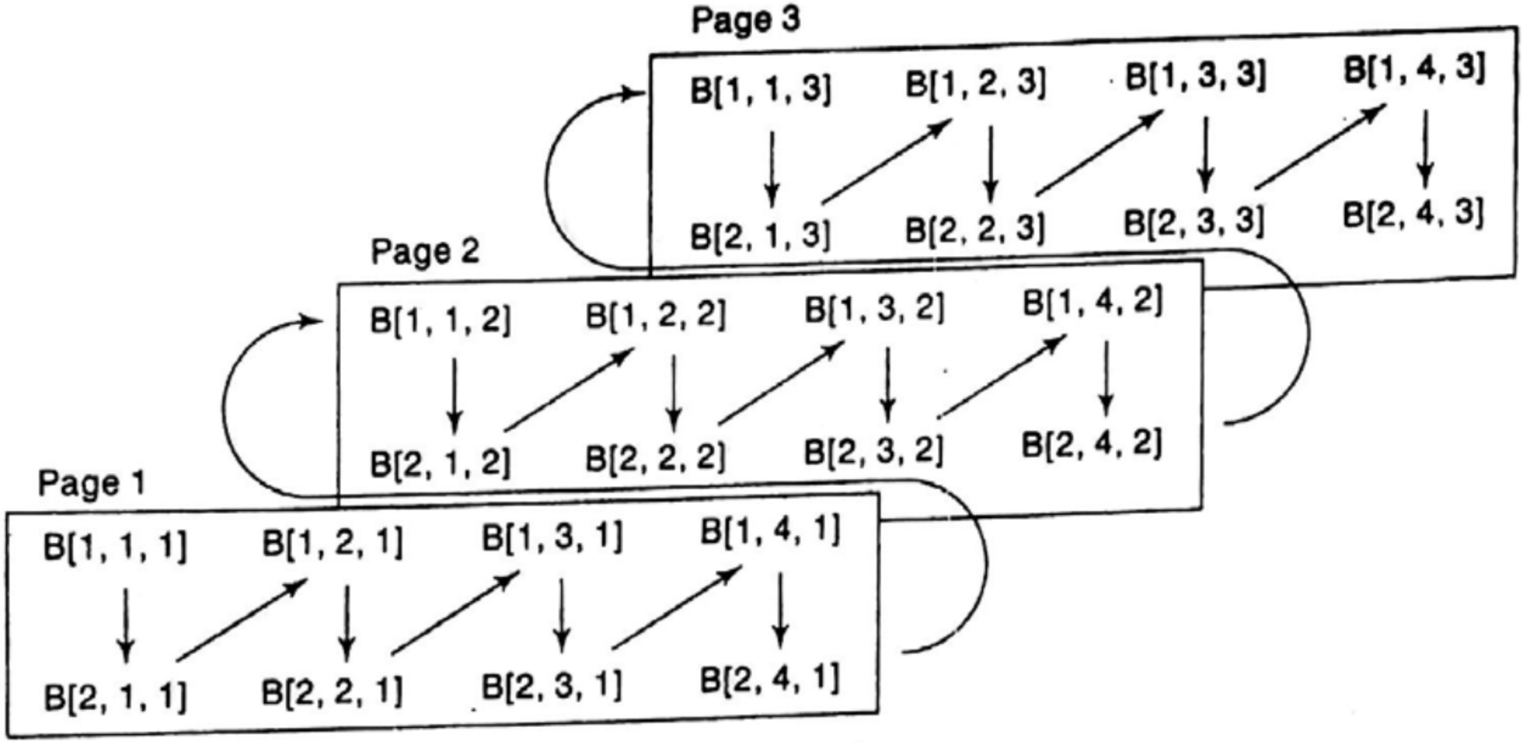
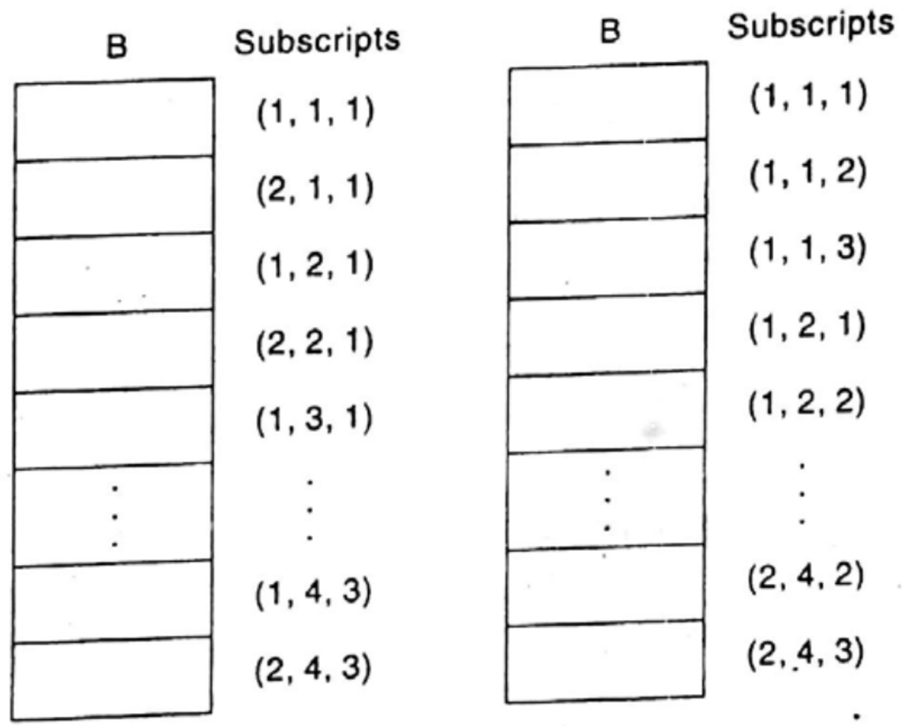


Fig. 4.11

Multidimensional Array



(a) Column-major order (b) Row-major order

Fig. 4.12

Pointers: Pointer Array

Pointers: Pointer Array

- Pointer Array

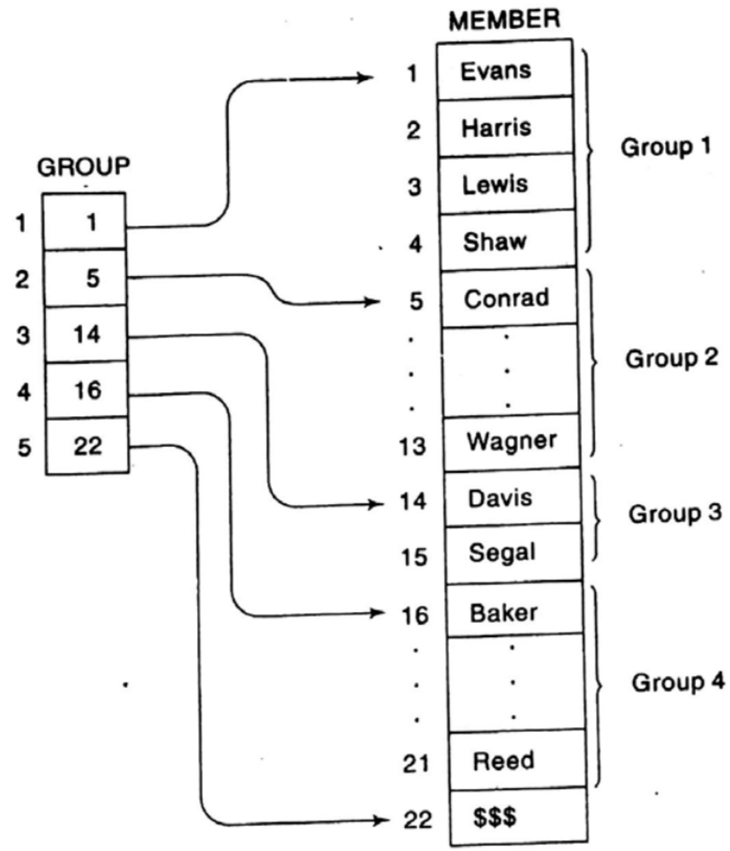


Fig. 4.16

Pointers: Pointer Array

- Pointer Array

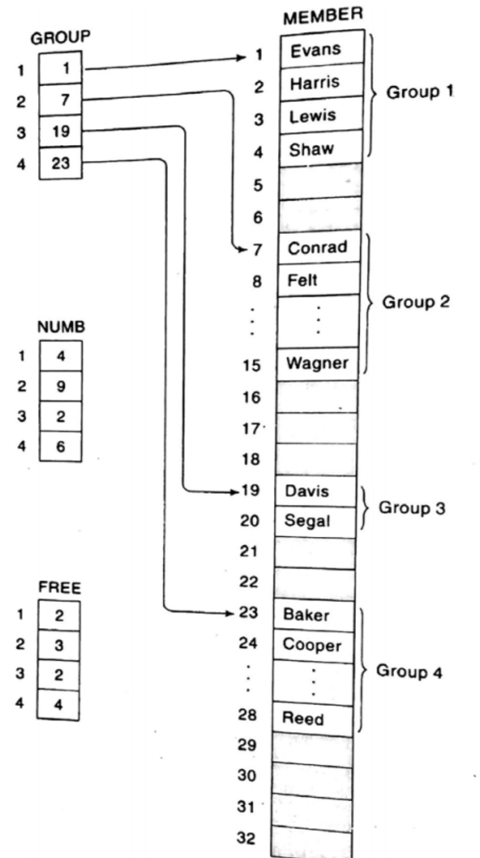


Fig. 4.17

Records: Record Structure

Records: Record Structure

- Differs from a linear array in the following ways –
  - A record may be a collection of nonhomogeneous data
  - The data items in a record are indexed by attribute name, so there may not be a natural ordering of its element.

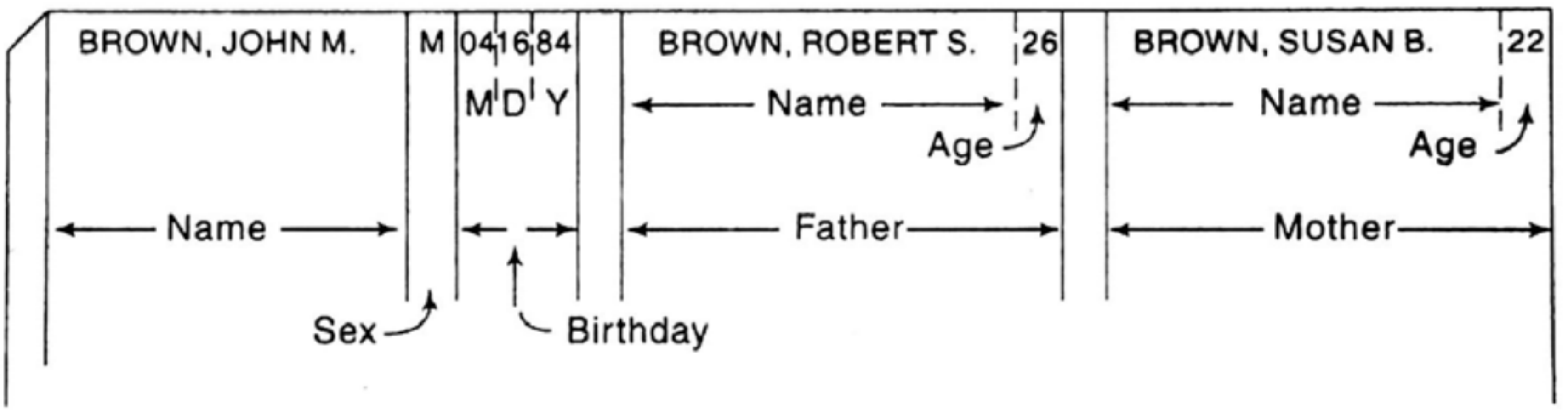


Fig. 4.18

Representation of Records in Memory: Parallel Arrays

Representation of Records in Memory: Parallel Arrays

	NAME	AGE	SEX	PHONE
1	John Brown	28	Male	234-5186
2	Paul Cohen	33	Male	456-7272
3	Mary Davis	24	Female	777-1212
4	Linda Evans	27	Female	876-4478
5	Mark Green	31	Male	255-7654
⋮	⋮	⋮	⋮	⋮

Fig. 4.19

Matrices

Matrices

- An n-elements **vector** V –

$$V = (V_1, V_2, \ldots, V_n)$$

- An mxn **matrix** A -

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{pmatrix}$$

Matrices

- Scalar Product of two n-elements **vectors** V and U –

$$U \cdot V = U_1V_1 + U_2V_2 + \ldots + U_nV_n = \sum_{k=1}^{n} U_kV_k$$

- Matrix multiplication

$$C_{ij} = A_{i1}B_{1j} + A_{i2}B_{2j} + \ldots + A_{ip}B_{pj} = \sum_{k=1}^{p} A_{ik}B_{kj}$$

A is an $m \times p$

B is a $p \times n$ matrix.

Matrices

Algorithm 4.7: (Matrix Multiplication) MATMUL(A, B, C, M, P, N)
Let A be an $M \times P$ matrix array, and let B be a $P \times N$ matrix array. This algorithm stores the product of A and B in an $M \times N$ matrix array C.

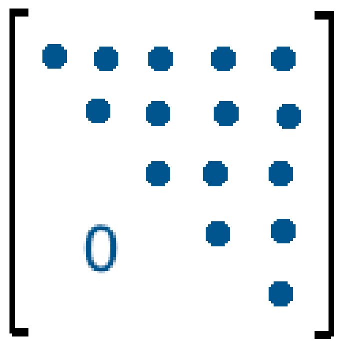
1. Repeat Steps 2 to 4 for I = 1 to M:
2. Repeat Steps 3 and 4 for J = 1 to N:
3. Set C(I, J] := 0. [Initializes C(I, J].]
4. Repeat for K = 1 to P:
   C(I, J] := C(I, J] + A[I, K] * B[K, J]
   [End of inner loop.]
   [End of Step 2 middle loop.]
   [End of Step 1 outer loop.]
5. Exit.

Complexity: $O(n^3)$

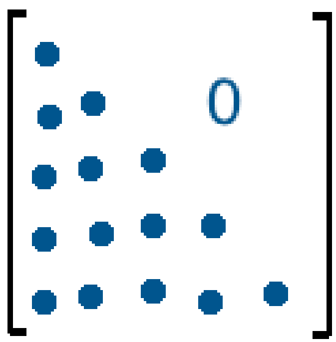
Sparse Matrices

Sparse Matrices

- High proportion of zero entries.



Upper Triangular Matrix



Lower Triangular Matrix

Sparse Matrices

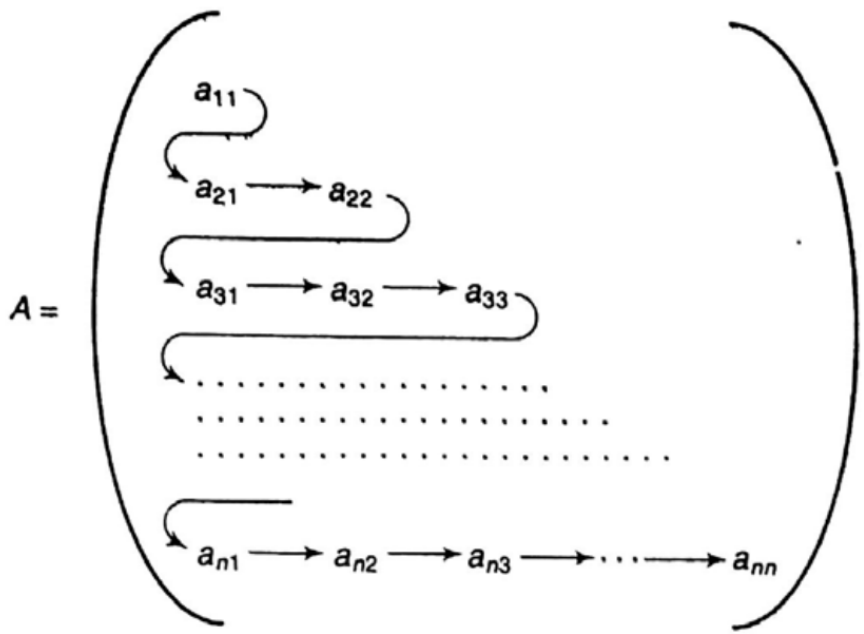


Fig. 4.22

$$B[1] = a_{11},\quad B[2] = a_{21},\quad B[3] = a_{22},\quad B[3] = a_{31},\quad \ldots$$

Sparse Matrices

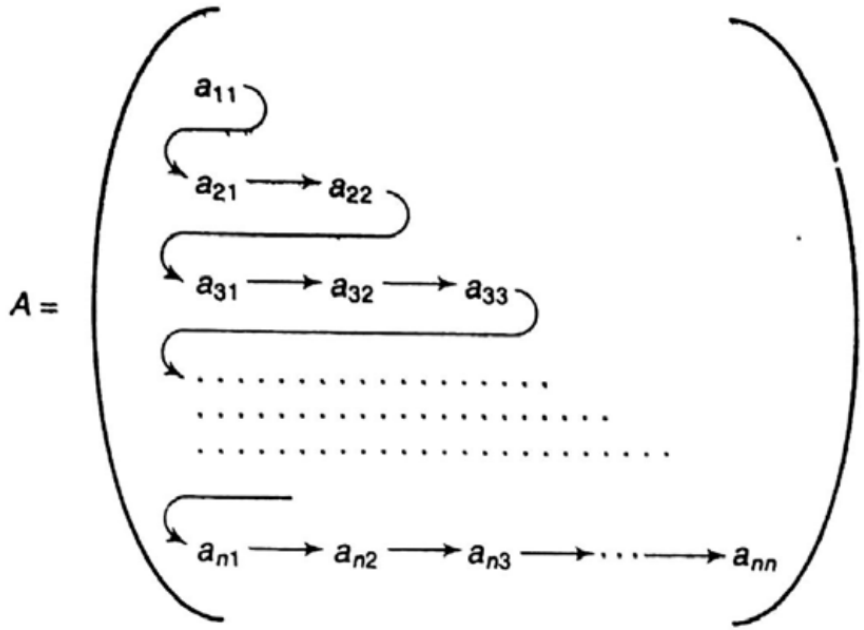


Fig. 4.22

$$1 + 2 + 3 + 4 + \ldots + n = \frac{1}{2}n(n + 1)$$

Sparse Matrices

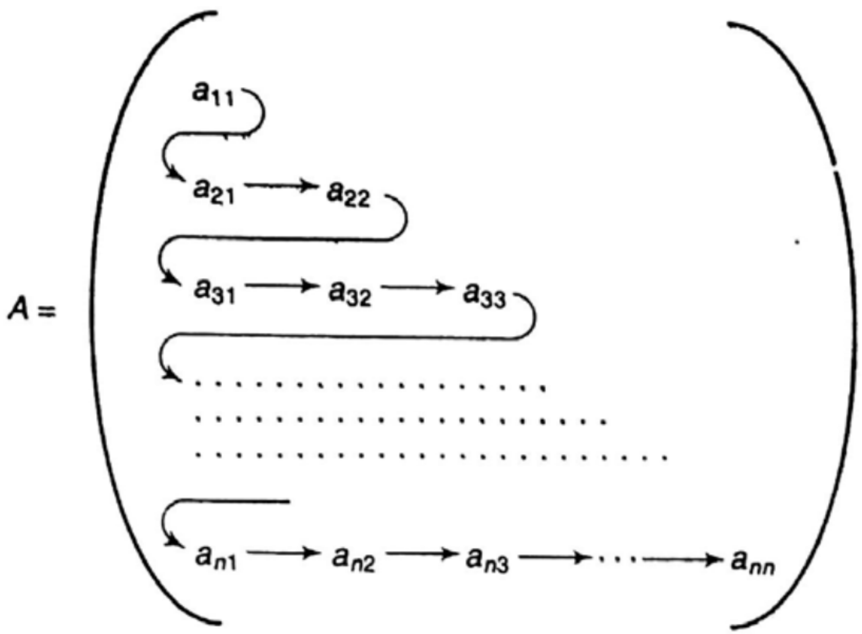


Fig. 4.22

$$B[L] = a_{JK}$$

1^{st} row - 1 element
2^{nd} row - 2 elements
….
Jth row - k elements

(J-1)th row -
1+2 + …. + (J-1)

So L = 1+2+⋯ +(J-1) + K

Sparse Matrices

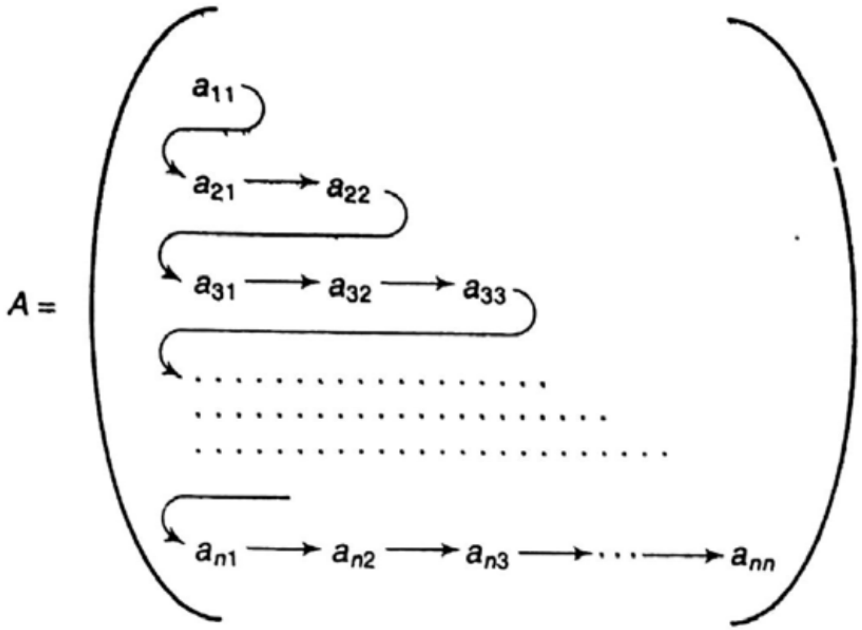


Fig. 4.22

$$B[L] = a_{JK}$$

$$L = \frac{J(J-1)}{2} + K$$

END