PARALLEL PROCESSING

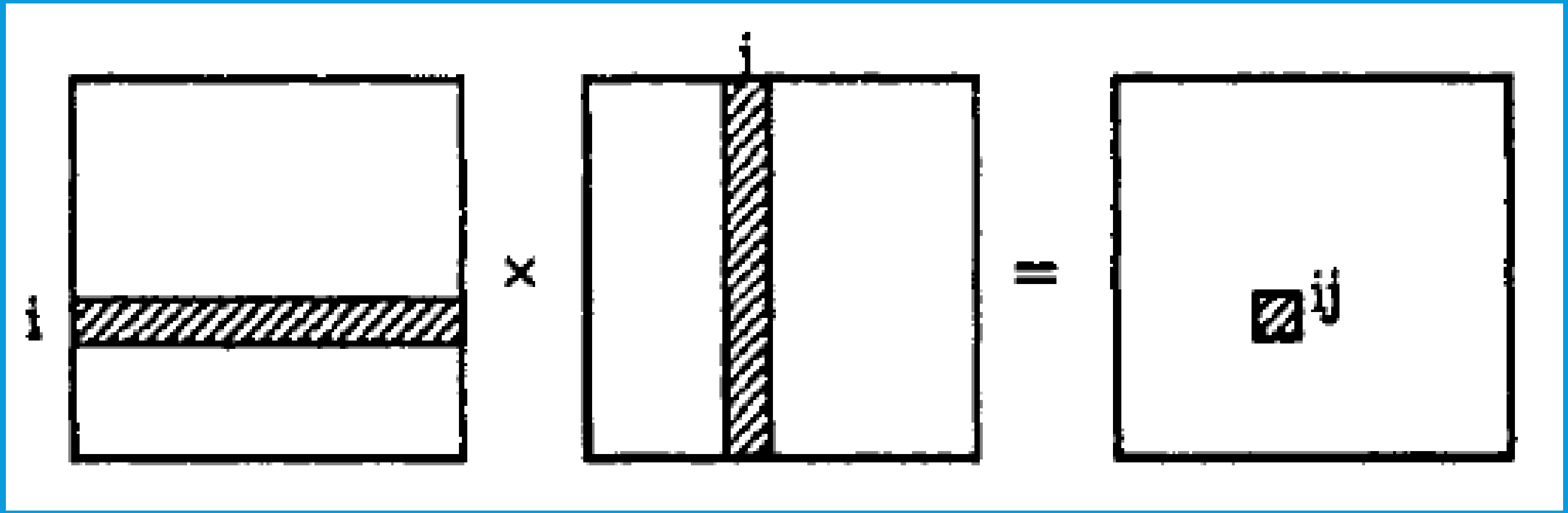
Mohammed Alabdulkareem

kareem@ksu.edu.sa

Office 2247

8

MATRIX MULTIPLICATION ON PRAM



MATRIX MULTIPLICATION ON PRAM

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}$$

MATRIX MULTIPLICATION ON PRAM

$$c_{ij} = \sum_{k=0}^{m-1} a_{ik} b_{kj}$$

Sequential matrix multiplication algorithm

```
for i = 0 to m – 1 do
  for j = 0 to m – 1 do
    t := 0
    for k = 0 to m – 1 do
      t := t + a_ik b_kj
    endfor
    c_ij := t
  endfor
endfor
```

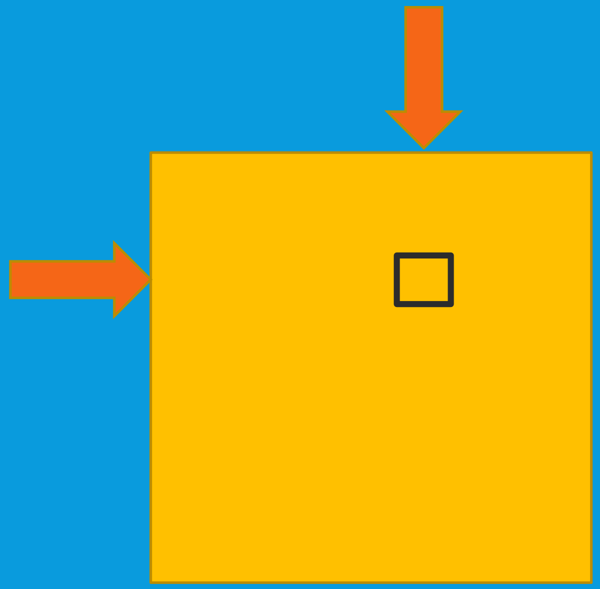
MATRIX MULTIPLICATION ON PRAM

PRAM matrix multiplication algorithm using m^2 processors

Processor (i, j), $0 \le i, j < m$, do
begin
    $t := 0$
    for $k = 0$ to $m - 1$ do
        $t := t + a_{ik} b_{kj}$
    endfor
    $c_{ij} := t$
end

MATRIX MULTIPLICATION ON PRAM

- When PRAM has $p=m^2$ processors
  - Processor *i* , *j:*
    - Read the elements of Row *i* in *A*
    - Read the elements of column j in *B*
    - Write the results in C.
  - This requires CREW sub model.
  - p=m^2 processors the algorithm takes $\Theta(m)$ time



MATRIX MULTIPLICATION ON PRAM

King Saud University

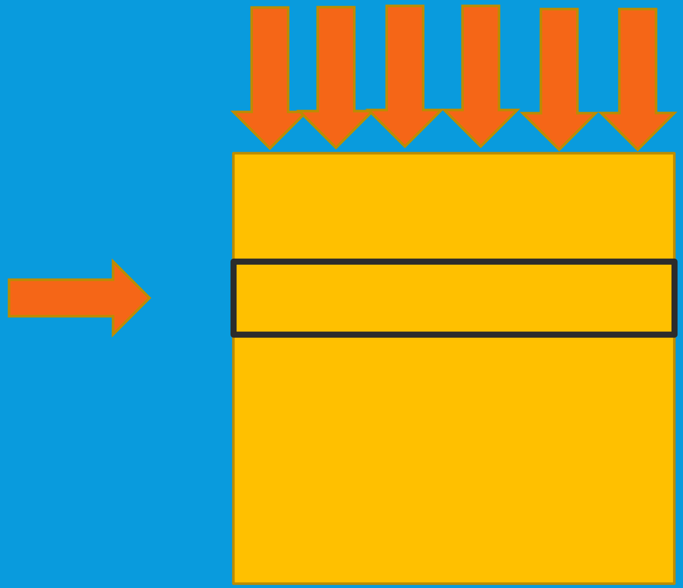
At processor i

PRAM matrix multiplication algorithm using m processors

for $j = 0$ to $m - 1$ Processor i, $0 \le i < m$, do
  $t := 0$
  for $k = 0$ to $m - 1$ do
    $t := t + a_{ik} b_{kj}$
  endfor
  $c_{ij} := t$
endfor

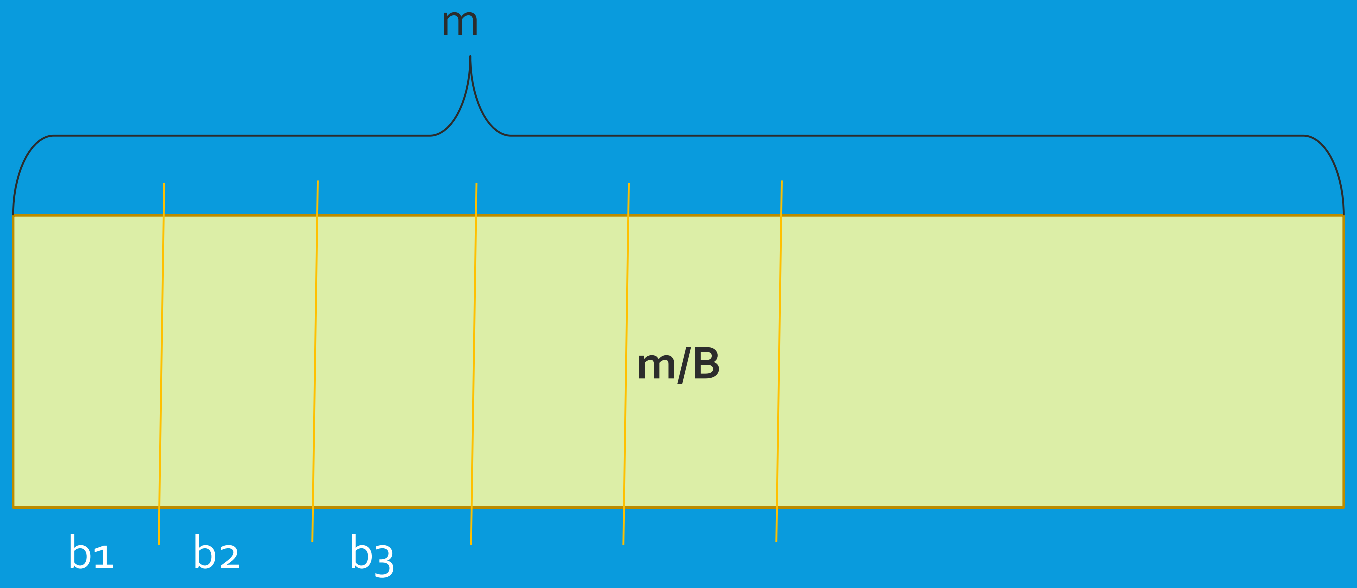
MATRIX MULTIPLICATION ON PRAM

- When PRAM has p=m processors
  - Processor *i* :
    - Read the elements of Row *i* in *A*
    - Read the elements of all columns in *B*
    - Write the results in row *i* in *C*
  - This requires CREW sub model.
  - p=m processors the algorithm takes $\Theta(m^2)$ time



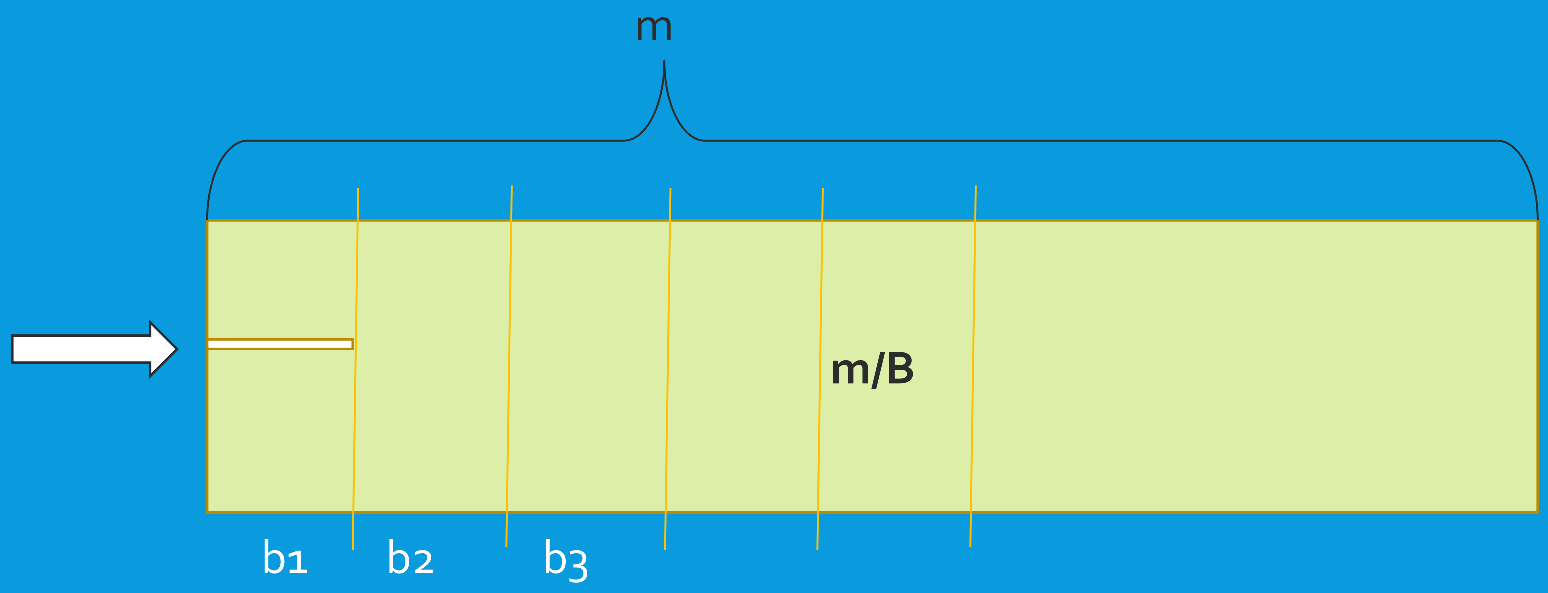
MATRIX MULTIPLICATION ON PRAM

- In any physical implementation of shared memory, the *m* memory locations will be in *B* memory banks (modules), each bank holding *m*/*B* addresses.



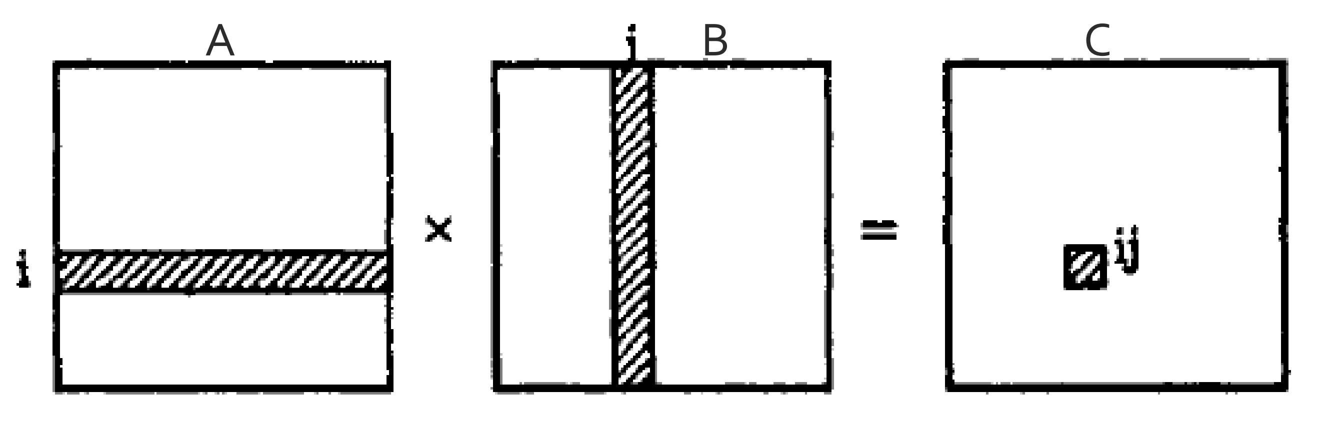
MATRIX MULTIPLICATION ON PRAM

- Typically, a memory bank can provide access to a single memory word in a given memory cycle.



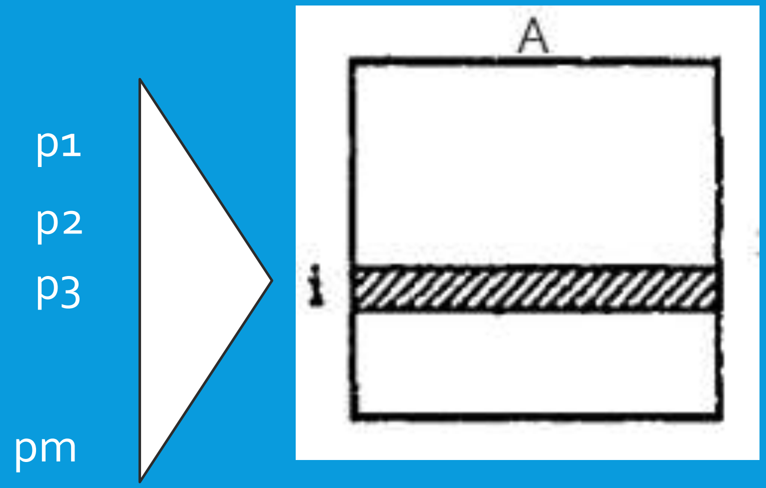
MATRIX MULTIPLICATION ON PRAM

- Let us take the $m \times m$ matrix multiplication algorithm in which $p = m^2$ processors are used.
- We identify each processor by an index pair (i, j). Then, Processor P_{ij} will be responsible for computing the element c_{ij} of the result matrix C.



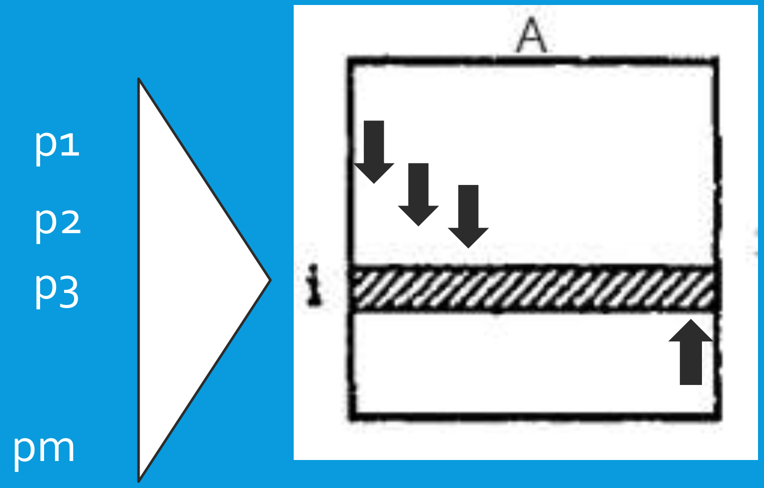
MATRIX MULTIPLICATION ON PRAM

- The m processors $P_{i\,y}$, $0 \le y < m$, would need to read (the same row) Row i of the matrix A for their computation.



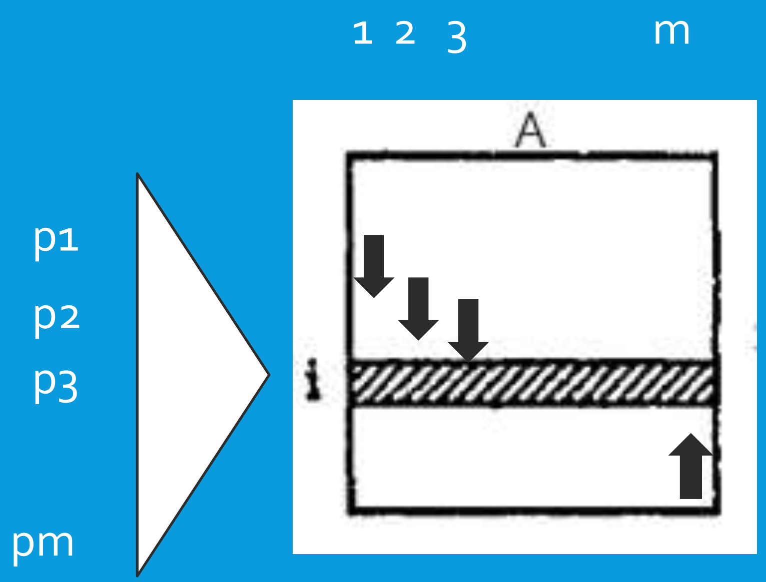
MATRIX MULTIPLICATION ON PRAM

- In order to avoid multiple accesses to the same matrix element, we skew the accesses so that P*i y* reads the elements of Row *i* beginning with *Ai y*.
- In this way, the entire Row *i* of *A* is read out in every cycle, although with the elements distributed differently to the processors in each cycle.

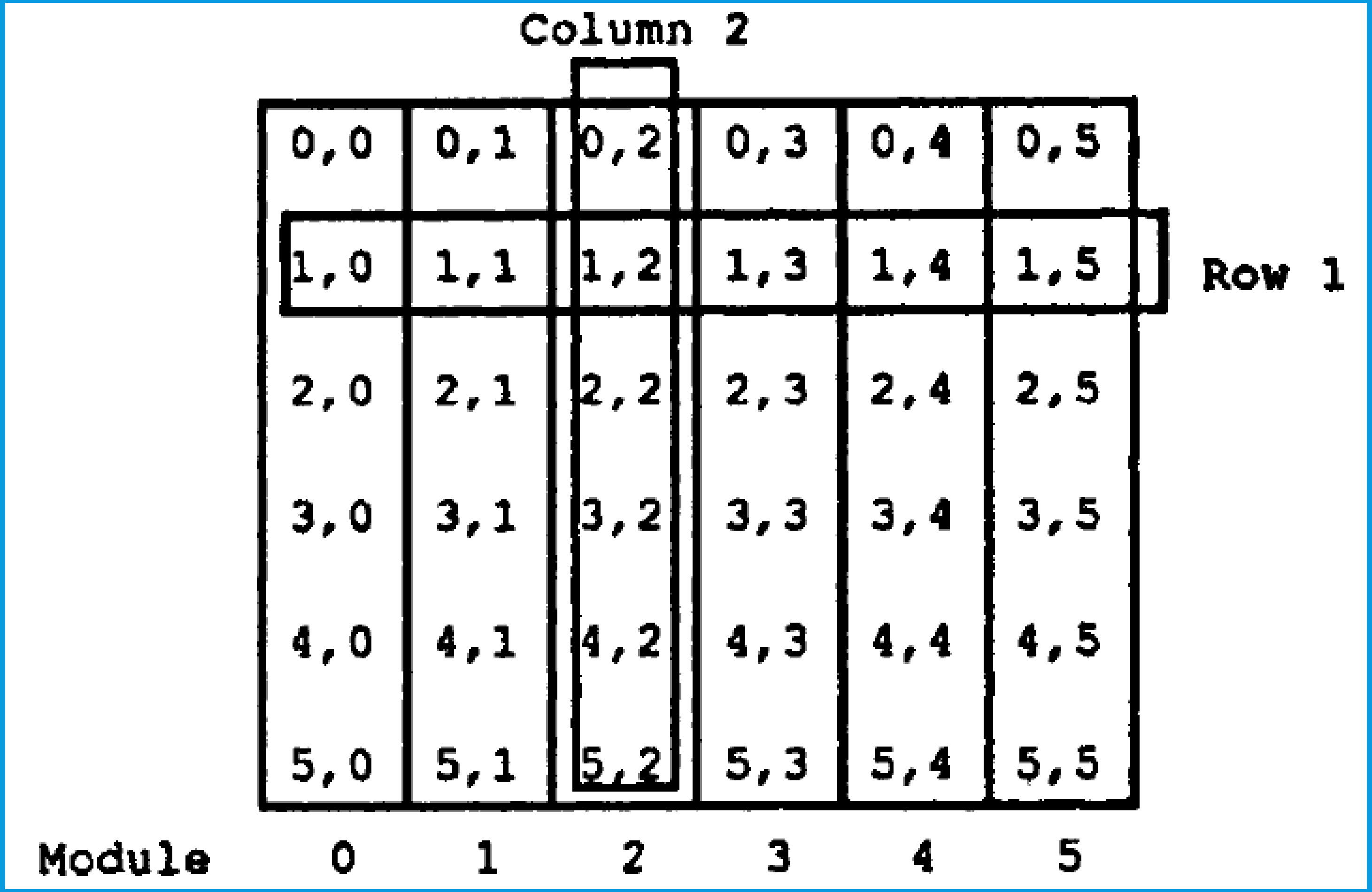


MATRIX MULTIPLICATION ON PRAM

- data layout must assign different columns of *A* to different memory banks. This is possible if we have at least *m* memory banks and corresponds to the data storage in column-major order



MATRIX MULTIPLICATION ON PRAM



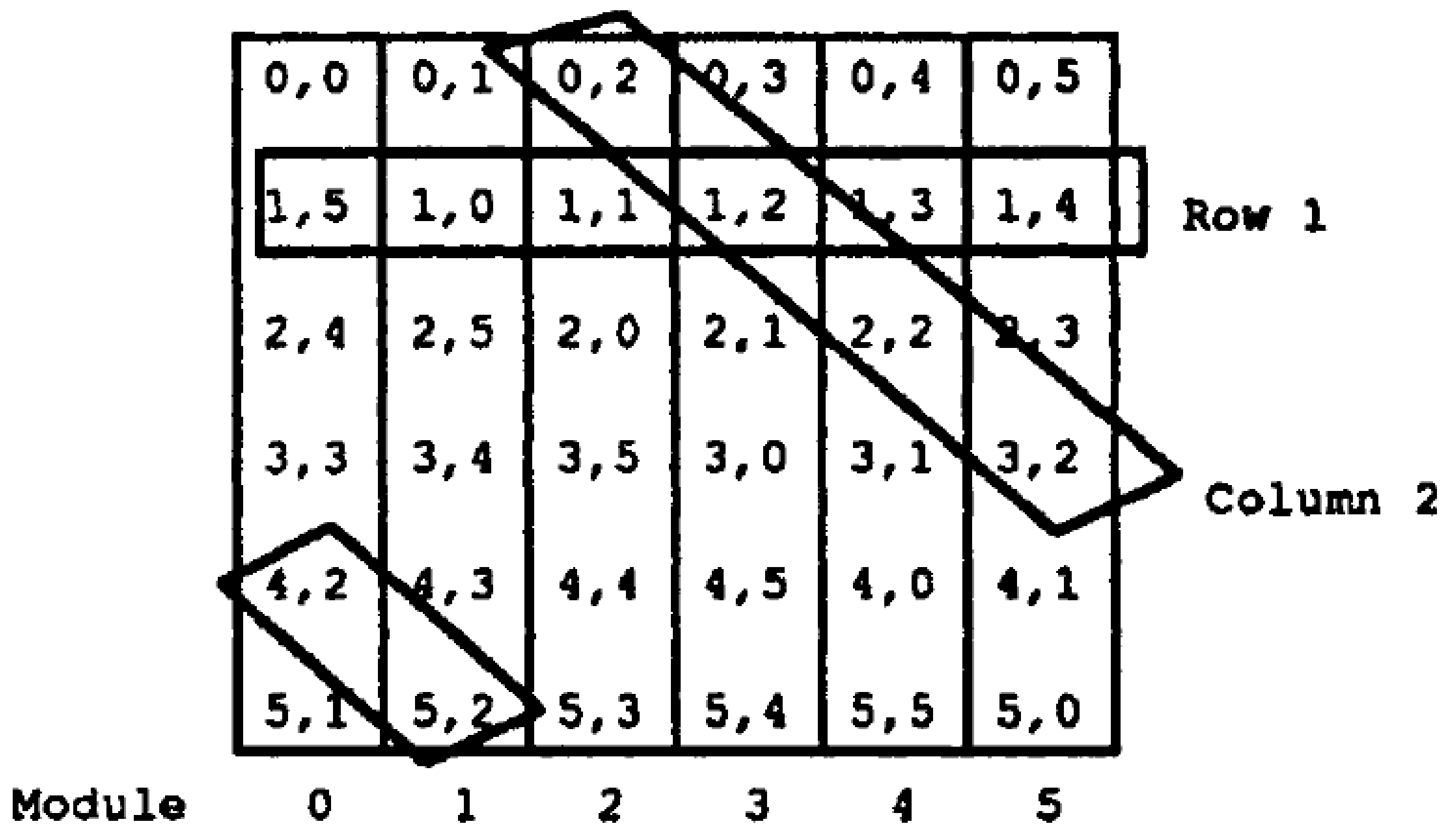
MATRIX MULTIPLICATION ON PRAM

- note that Processors $P_{x\,j}$, $0 \le x < m$, all access the jth column of matrix B. Therefore, the column-major storage scheme is not suitable and will lead to memory bank conflicts for all such accesses to the columns of B.
- If we store B in row-major order to avoid such conflicts. We may later need to use B in a different matrix multiplication B x D.
- Should we rearrange B in memory or change the algorithm??

MATRIX MULTIPLICATION ON PRAM

- A matrix can be laid out in memory in such a way that both columns and rows are accessible in parallel without memory bank conflicts.
- This is called skewed storage scheme that allows conflict-free access to both rows and columns of a matrix.
- matrix element (i, j) is found in location i of module $(i + j) \bmod B$ *(B number of memory banks).*
- It is clear from this formulation that all elements (i, y), $0 \leq y < m$, will be found in different modules, as are all elements (x, j), $0 \leq x < m$, provided that $B \geq m$.

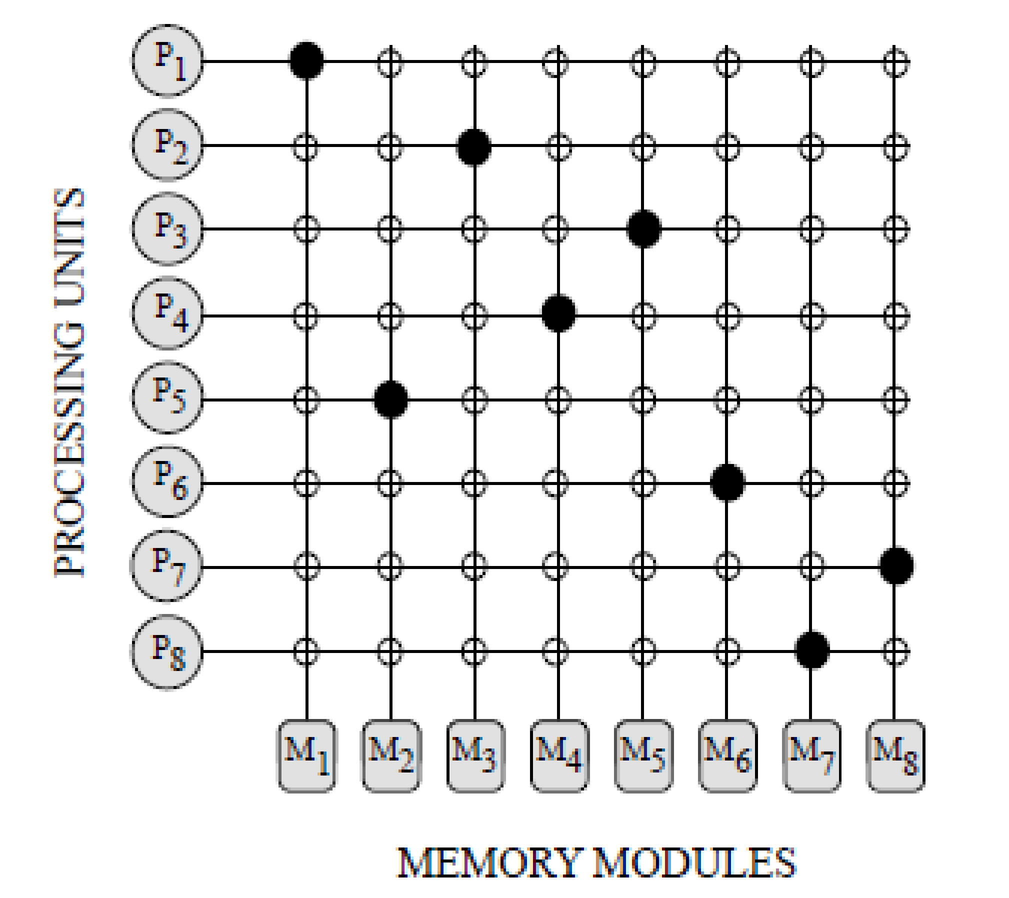
MATRIX MULTIPLICATION ON PRAM



MATRIX MULTIPLICATION ON PRAM

- A multiple memory access requests must be directed from the processors to the associated memory banks.
- With a large number of processors and memory banks, this is a nontrivial problem.
- Ideally, the memory access network should be a permutation network that can connect each processor to any memory bank as long as the connection is a permutation.

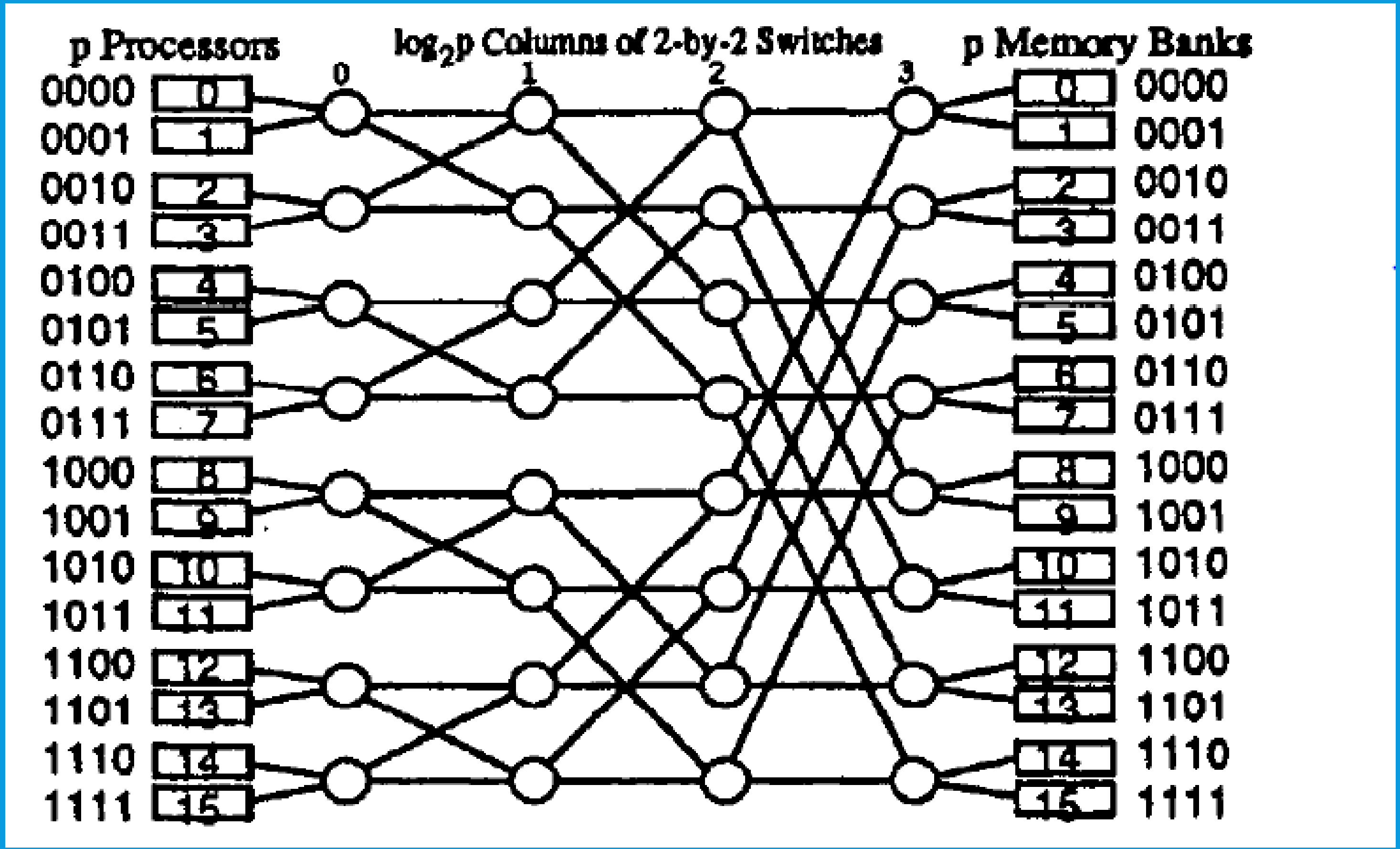
MATRIX MULTIPLICATION ON PRAM



MATRIX MULTIPLICATION ON PRAM

- Permutation networks are quite expensive to implement and difficult to control.
- We usually settle for networks that do not possess full permutation capability.
- Multistage interconnection network as an example of a compromise solution.
- The following is the Butterfly interconnection network.

MATRIX MULTIPLICATION ON PRAM



MATRIX MULTIPLICATION ON PRAM