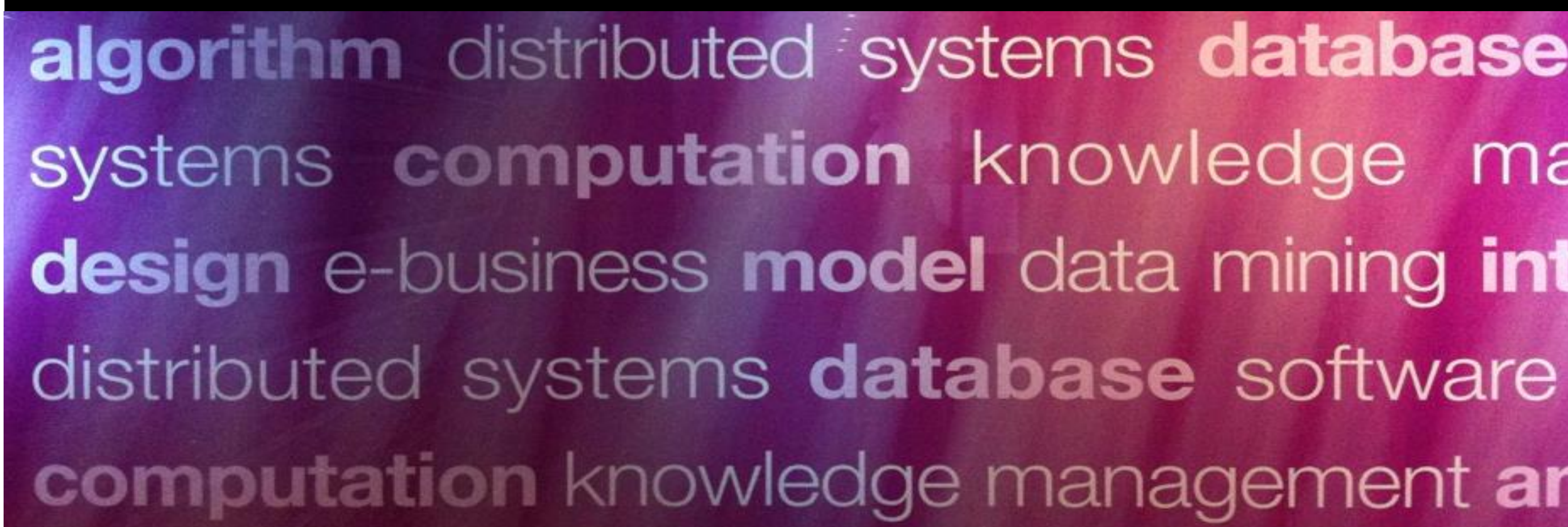
Information Technology

FIT3143 - LECTURE WEEK 8

PARALLEL ALGORITHM DESIGN -
PARTITIONING BASED ON MATRIX OPERATIONS



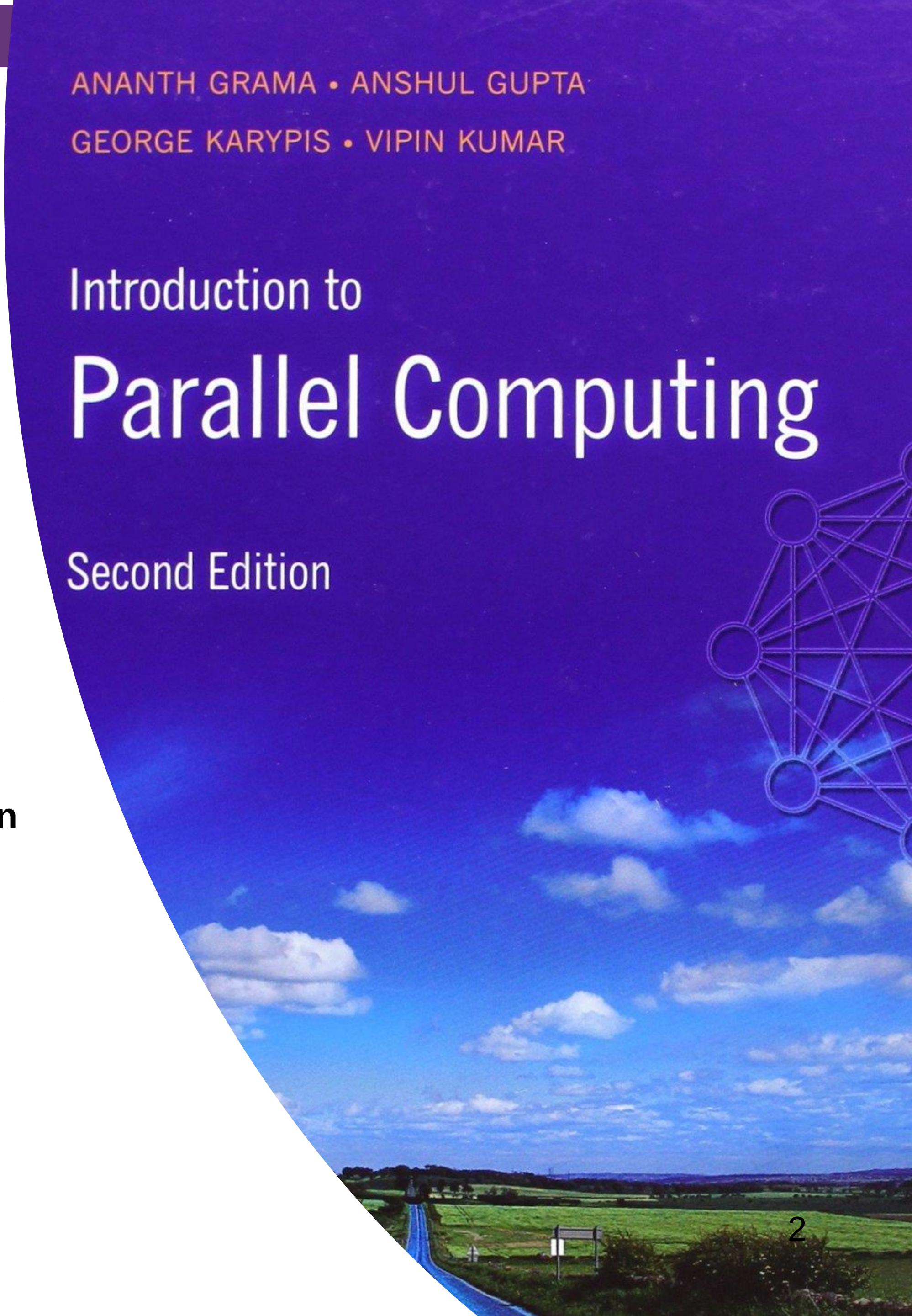
Topic Overview

- Matrix Algorithms & Problem Statement
- Decomposition
- Decomposition – Fox’s method

A portion of the content in the following slides were created by:

a) Gergel V.P., Nizhni Novgorod, **Introduction to Parallel Programming: Matrix Multiplication**, 2005.

b) Ananth Grama, Anshul Gupta, George Karypis, and Vipin Kumar, **“Introduction to Parallel Computing”, Addison Wesley, 2003.**



Matrix Algorithms: Introduction

- Due to their regular structure, parallel computations involving matrices and vectors readily lend themselves to data-decomposition.
- Typical algorithms rely on input, output, or intermediate data decomposition.
- Most algorithms use one- and two-dimensional block, cyclic, and block-cyclic partitionings.

Problem Statement

Matrix multiplication:

$$C = A \cdot B$$

or

$$\begin{pmatrix} c_{0,0}, & c_{0,1}, & \ldots, & c_{0,l-1} \\ & & \ldots & \\ c_{m-1,0}, & c_{m-1,1}, & \ldots, & c_{m-1,l-1} \end{pmatrix} = \begin{pmatrix} a_{0,0}, & a_{0,1}, & \ldots, & a_{0,n-1} \\ & & \ldots & \\ a_{m-1,0}, & a_{m-1,1}, & \ldots, & a_{m-1,n-1} \end{pmatrix} \begin{pmatrix} b_{0,0}, & b_{0,1}, & \ldots, & a_{0,l-1} \\ & & \ldots & \\ b_{n-1,0}, & b_{n-1,1}, & \ldots, & b_{n-1,l-1} \end{pmatrix}$$

The matrix multiplication problem can be reduced to the execution of m·l independent operations of matrix A rows and matrix B columns inner product calculation

$$c_{ij} = \left(a_i, b_j^T\right) = \sum_{k=0}^{n-1} a_{ik} \cdot b_{kj}, \ 0 \le i < m, \ 0 \le j < l$$

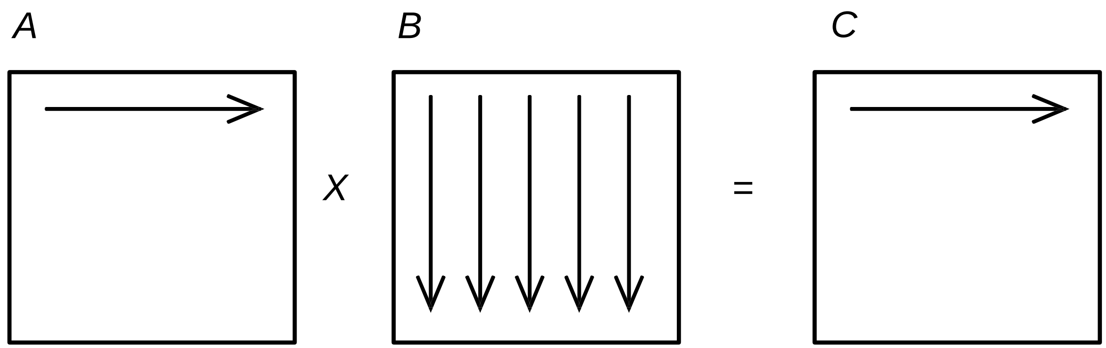
Data parallelism can be exploited to design parallel computations

Sequential Algorithm

```
// Sequential algorithm of matrix multiplication
double MatrixA[Size][Size];
double MatrixB[Size][Size];
double MatrixC[Size][Size];
int i,j,k;
...
for (i=0; i<Size; i++){
  for (j=0; j<Size; j++){
    MatrixC[i][j] = 0;
    for (k=0; k<Size; k++){
      MatrixC[i][j] = MatrixC[i][j] + MatrixA[i][k]*MatrixB[k][j];
    }
  }
}
```

Sequential Algorithm

- *Algorithm performs the matrix **C** rows calculation sequentially*
- *At every iteration of the outer loop on **i** variable a single row of matrix **A** and all columns of matrix **B** are processed*



- ***m·l** inner products are calculated to perform the matrix multiplication*
- *The complexity of the matrix multiplication is O(mnl).*

Block-Striped Composition

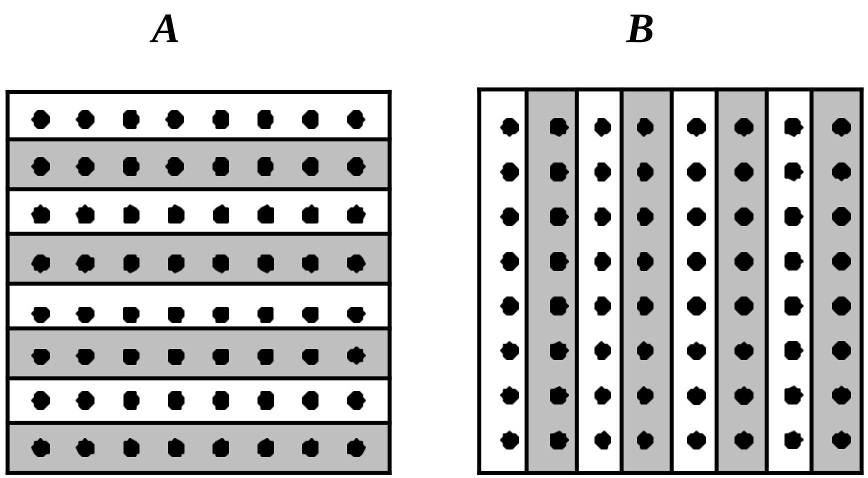
- ***A fine-grained approach*** *– the basic subtask is calculation of one element of matrix* ***C***

$$c_{ij} = \left(a_i, b_j^T\right),\; a_i = \left(a_{i0}, a_{i1}, \ldots, a_{in-1}\right),\; b_j^T = \left(b_{0j}, b_{1j}, \ldots, b_{n-1j}\right)^T$$

- *Number of basic subtasks is equal to* ***n².***
- *As a rule, the number of available processors is less then* ***n²*** *(**p<n²**), so it will be necessary to perform the subtask scaling*

Block-Striped Decomposition

- ***The aggregated subtask*** *– the calculation of one row of matrix **C** (the number of subtasks is **n**)*
- ***Data distribution*** *– rowwise block-striped decomposition for matrix **A** and columnwise block-striped decomposition for matrix **B***



Block-Striped Decomposition

Analysis of Information Dependencies

*Each subtask hold one row of matrix **A** and one column of matrix **B**,*

- *At every iteration each subtask performs the inner product calculation of its row and column, as a result the corresponding element of matrix **C** is obtained*
- *Then every subtask i, $0 \le i < n$, transmits its column of matrix **B** for the subtask with the number (i+1) mod n.*

*After all algorithm iterations all the columns of matrix **B** were come within each subtask one after another*

Block-Striped Decomposition

Aggregating and Distributing the Subtasks among the Processors:

- *In case when the number of processors **p** is less than the number of basic subtasks **n**, calculations can be aggregated in such a way that each processor would execute several inner products of matrix **A** rows and matrix **B** columns. In this case after the completion of computation, each aggregated basic subtask determines several rows of the result matrix **C.***
- *Under such conditions the initial matrix **A** is decomposed into **p** horizontal stripes and matrix **B** is decomposed into **p** vertical stripes.*
- *Subtasks distribution among the processors have to meet the requirements of effective representation of the ring structure of subtask information dependencies.*

Efficiency Analysis…

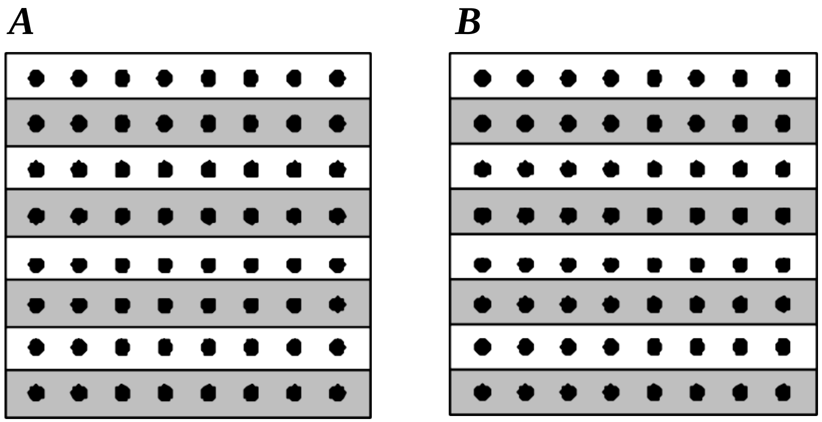
– *Speed-up and Efficiency generalized estimates*

$$S_p = \frac{n^3}{(n^3/p)} = p \qquad E_p = \frac{n^3}{p \cdot (n^3/p)} = 1$$

Developed method of parallel computations allows to achieve ideal speed-up and efficiency characteristics

Block-Striped Decomposition

- *Another possible approach for the data distribution is the rowwise block-striped decomposition for matrices **A** and **B***



Block-Striped Decomposition

Analysis of Information Dependencies

– *Each subtask hold one row of matrix **A** and one row of matrix **B**,*

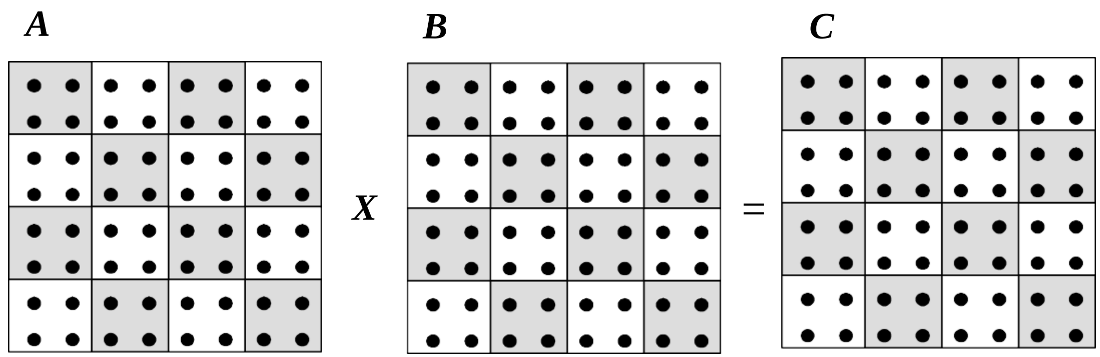
– *At every iteration the subtasks perform the element-to-element multiplications of the rows; as a result the row of partial results for matrix **C** is obtained,*

– *Then every subtask i, 0≤ i<n, transmits its row of matrix **B** for the subtask with the number (i+1) mod n.*

*After all algorithm iterations all rows of matrix **B** were come within every subtask one after another*

Block-Striped Decomposition – Fox's method

Data distribution – *checkerboard scheme*



Basic subtask *is a procedure, that calculates all elements of one block of matrix* ***C***

$$\begin{pmatrix} A_{00}A_{01}\ldots A_{0q-1} \\ \cdots \\ A_{q-10}A_{q-11}\ldots A_{q-1q-1} \end{pmatrix} \times \begin{pmatrix} B_{00}B_{01}\ldots B_{0q-1} \\ \cdots \\ B_{q-10}B_{q-11}\ldots B_{q-1q-1} \end{pmatrix} = \begin{pmatrix} C_{00}C_{01}\ldots C_{0q-1} \\ \cdots \\ c_{q-10}C_{q-11}\ldots C_{q-1q-1} \end{pmatrix}, \quad C_{ij} = \sum_{s=0}^{q-1} A_{is}B_{sj}$$

Block-Striped Decomposition – Fox's method

Analysis of Information Dependencies

- *Subtask with (i,j) number calculates the block* $\boldsymbol{C_{ij}}$, *of the result matrix* ***C****. As a result, the subtasks form the* $\boldsymbol{q \times q}$ *two-dimensional grid,*
- *Each subtask holds 4 matrix blocks:*
  - *block* $\boldsymbol{C_{ij}}$ *of the result matrix* ***C****, which is calculated in the subtask,*
  - *block* $\boldsymbol{A_{ij}}$ *of matrix* ***A****, which was placed in the subtask before the calculation starts,*
  - *blocks* $\boldsymbol{A_{ij}'}$ *and* $\boldsymbol{B_{ij}'}$ *of matrix* ***A*** *and matrix* ***B****, that are received by the subtask during calculations.*

Block-Striped Decomposition – Fox's method

***Analysis of Information Dependencies** – during iteration l, $0 \le l < q$, algorithm performs:*

- *The subtask (i,j) transmits its block* $\mathbf{A_{ij}}$ *of matrix* **A** *to all subtasks of the same horizontal row* i *of the grid; the* j *index, which detemines the position of the subtask in the row, can be obtained using equation:*

  $$j = (i+l) \bmod q,$$

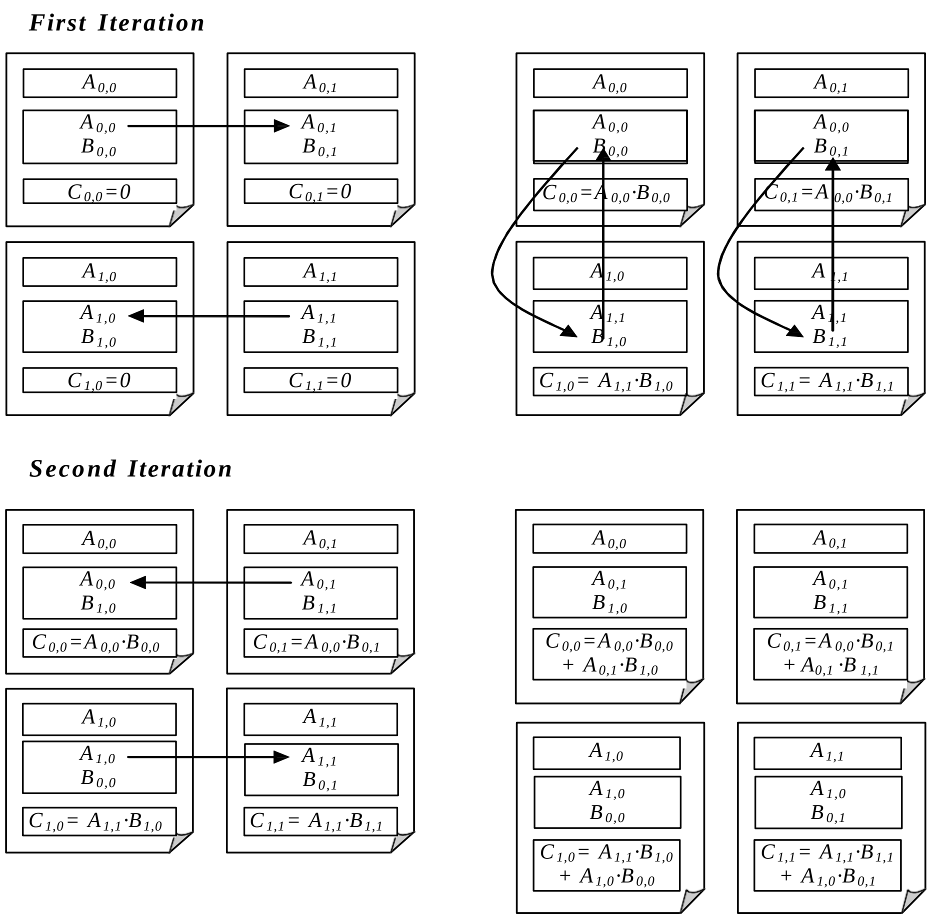
  where mod operation is the procedure of calculating the remainder of integer-valued division,
- *Every subtask performs the multiplication of received blocks* $\mathbf{A_{ij}'}$ *and* $\mathbf{B_{ij}'}$ *and adds the result to the block* $\mathbf{C_{ij}}$

  $$C_{ij} = C_{ij} + A'_{ij} \times B'_{ij}$$

- *Every subtask (i,j) transmits its block* $\mathbf{B_{ij}'}$ *to the neighbor, which is previous in the same vertical line (the blocks of subtasks of the first row are transmitted to the subtasks of the last row of the grid).*

Block-Striped Decomposition – Fox's method

Scheme of Information Dependences



*Gergel V.P., Nizhni Novgorod, **Introduction to Parallel Programming: Matrix Multiplication**, 2005*

Block-Striped Decomposition – Fox's method

Scaling and Distributing the Subtasks among the Processors

- *The sizes of the matrices blocks can be selected so that the number of subtasks will coincides the number of available processors* ***p****,*
- *The most efficient execution of the parallel the Fox's algorithm can be provided when the communication network topology is a two-dimensional grid,*
- *In this case the subtasks can be distributed among the processors in a natural way: the subtask (i,j) has to be placed to the* $\boldsymbol{p}_{i,j}$ *processor*

Block-Striped Decomposition – Fox's method

Scaling and Distributing the Subtasks among the Processors

- *The sizes of the matrices blocks can be selected so that the number of subtasks will coincides the number of available processors **p**,*
- *The most efficient execution of the parallel the Fox's algorithm can be provided when the communication network topology is a two-dimensional grid,*
- *In this case the subtasks can be distributed among the processors in a natural way: the subtask (i,j) has to be placed to the $\boldsymbol{p_{i,j}}$ processor*

In depth discussion & example

- Please refer to the enclosed report attached with these slides, "*Design and Implementation of Parallel Matrix Multiplication Algorithms using Message Passing Interface*" by Chin-Kit Ng for further in-depth discussion and code examples.
  - Serial matrix multiplication example
  - Bernstein analysis for data dependency
  - Parallel matrix multiplication examples using POSIX and MPI