Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Matrix-vector Multiplication

Michael Quinn

Parallel Programming with MPI and OpenMP

Material do autor

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

objetivos

- explorar operações MPI
- mostrar formas diferentes de dividir um mesmo programa sequencial

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Sequential Algorithm

2	1	0	4
3	2	1	1
4	3	1	2
3	0	2	0

$\times$

1
3
4
1

$=$

9
14
19
11

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

matrix decomposition

2	1	0	4
3	2	1	1
4	3	1	2
3	0	2	0

2	1	0	4
3	2	1	1
4	3	1	2
3	0	2	0

2	1	0	4
3	2	1	1
4	3	1	2
3	0	2	0

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Storing Vectors

- Divide vector elements among processes
- Replicate vector elements
- Vector replication acceptable because vectors have only n elements, versus n^2 elements in matrices

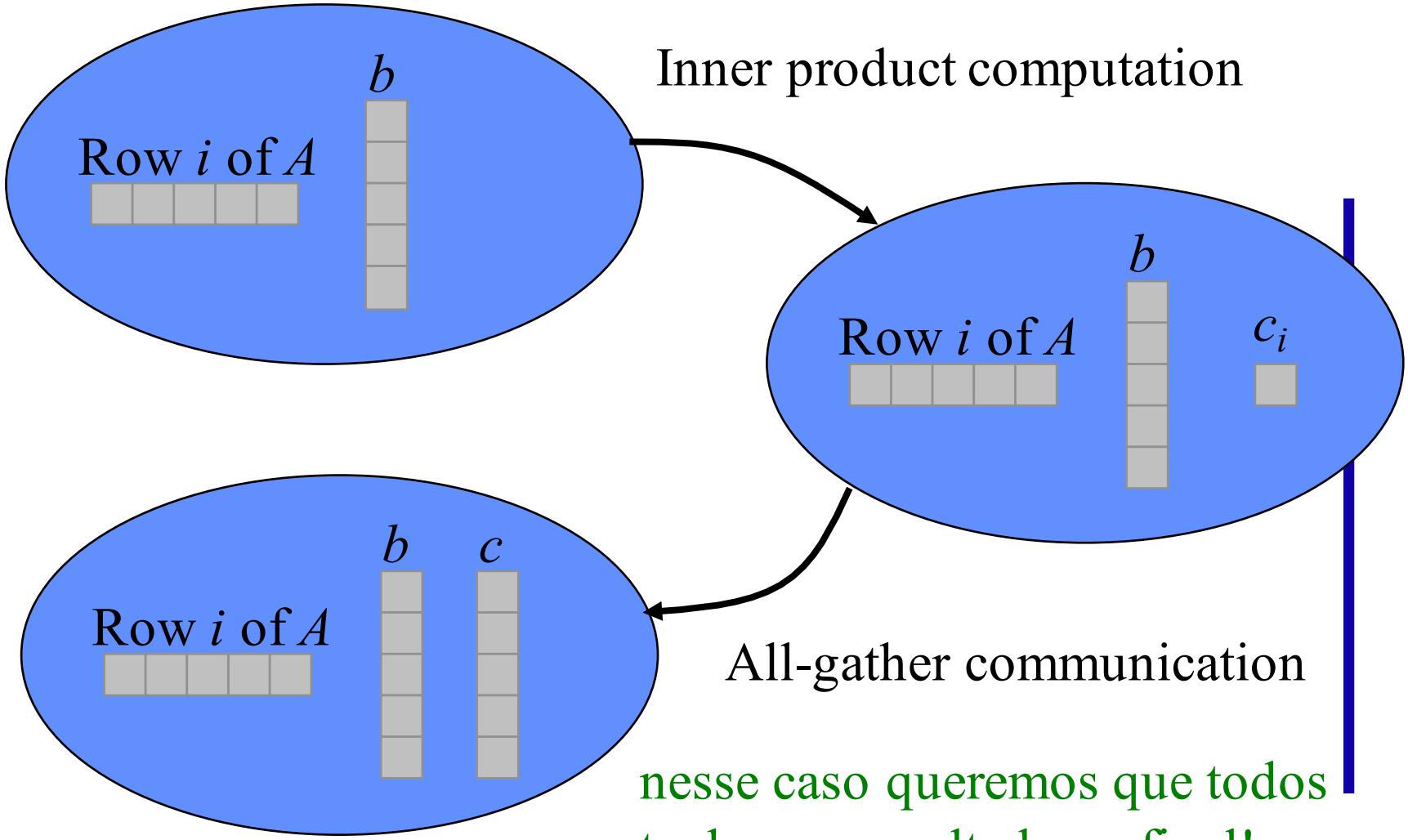
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Rowwise Block Striped Matrix

- Partitioning through domain decomposition
- Primitive task associated with
  - Row of matrix
  - Entire vector

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Phases of Parallel Algorithm - rowwise



nesse caso queremos que todos tenham o resultado ao final!

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Agglomeration and Mapping

- Static number of tasks
- Regular communication pattern (all-gather)
- Computation time per task is constant
- Strategy:
  - Agglomerate groups of rows
  - Create one task per MPI process

com matrizes esparsas poderíamos ter desbalanceamento!

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Read matrix/distribute rows

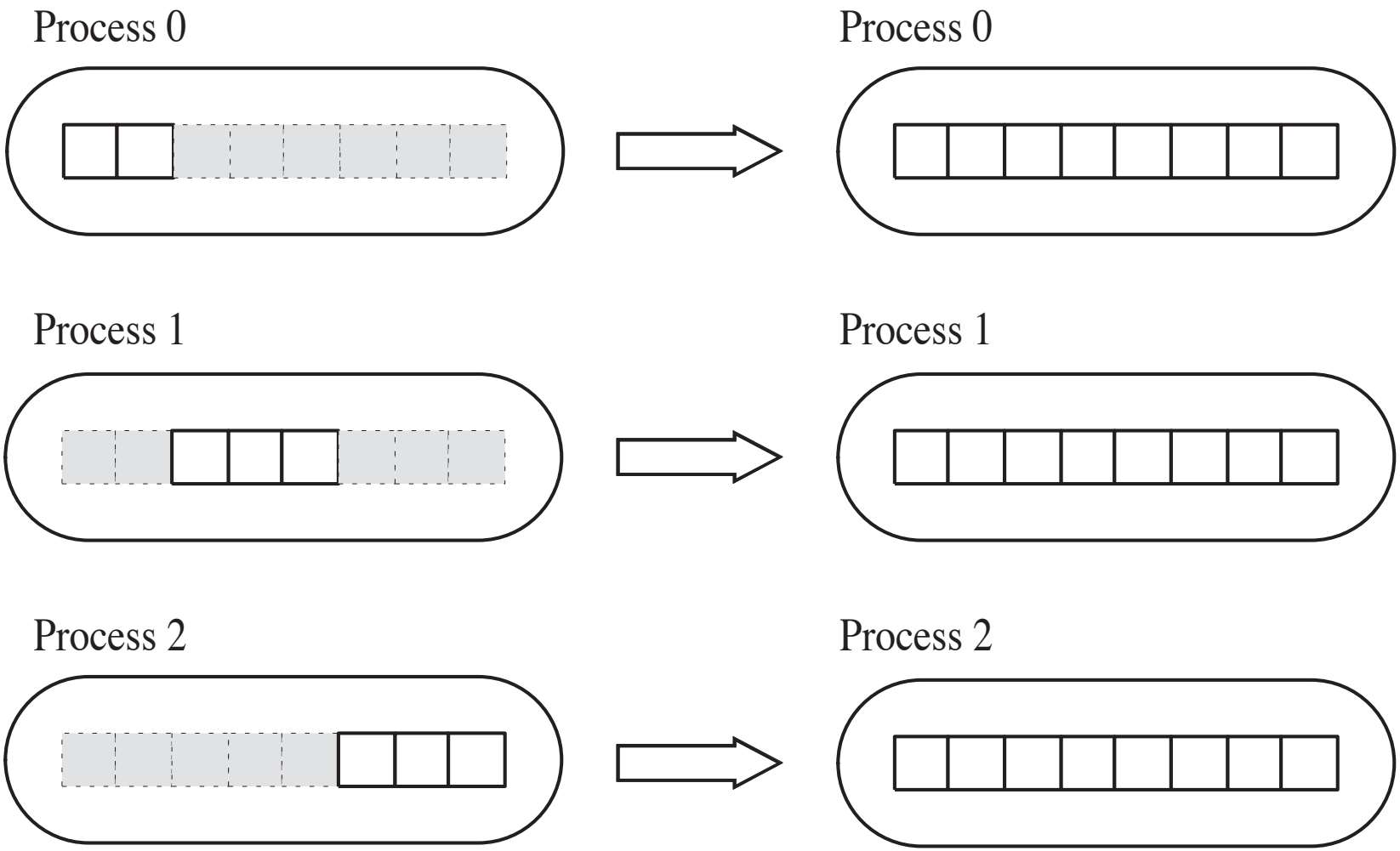
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Function read_replicated_vector

- Process *p*-1
  - Opens file
  - Reads vector length
- Broadcast vector length (root process = *p*-1)
- Allocate space for vector
- Process *p*-1 reads vector, closes file
- Broadcast vector

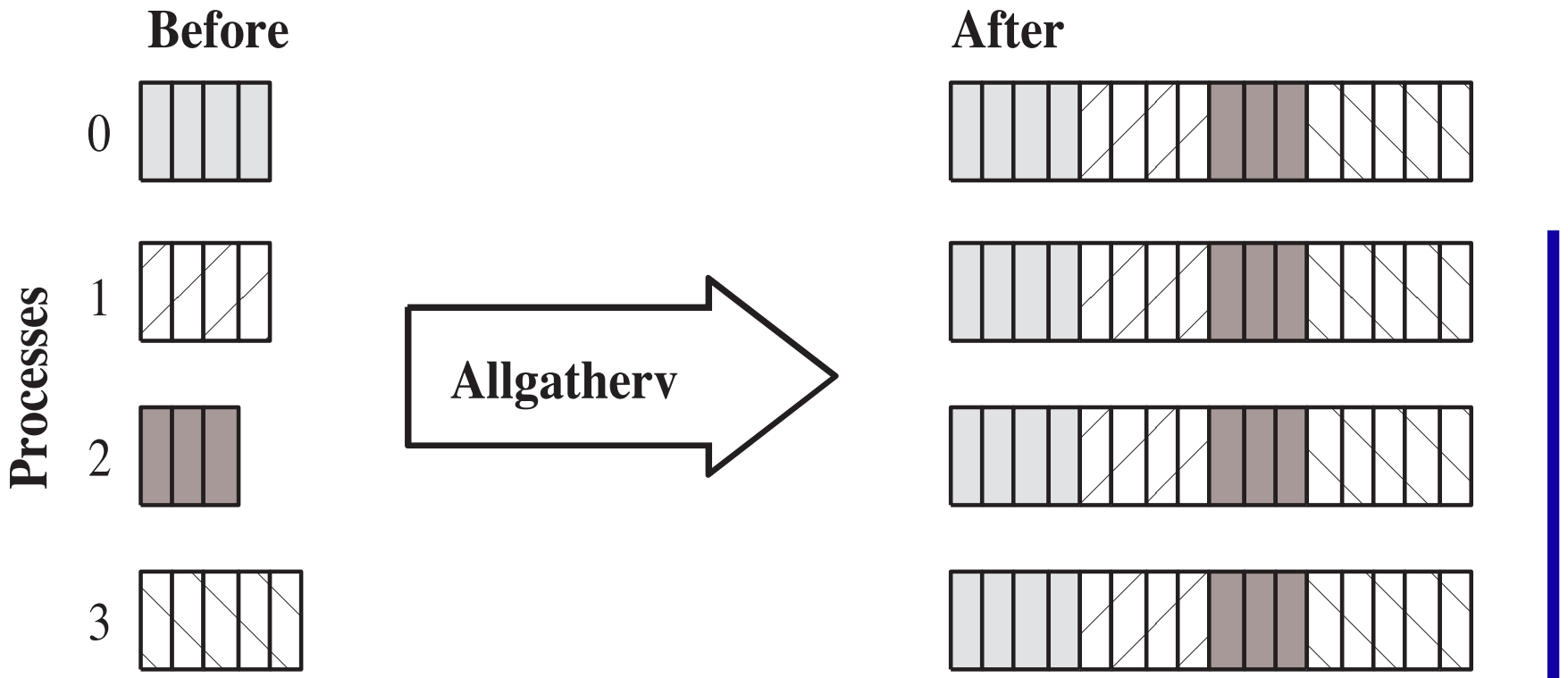
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Block-to-replicated Transformation - allgather



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Allgatherv



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Allgatherv

```
int MPI_Allgatherv (
    void            *send_buffer,
    int              send_cnt,
    MPI_Datatype     send_type,
    void            *receive_buffer,
    int             *receive_cnt,
    int             *receive_disp,
    MPI_Datatype     receive_type,
    MPI_Comm         communicator)
```

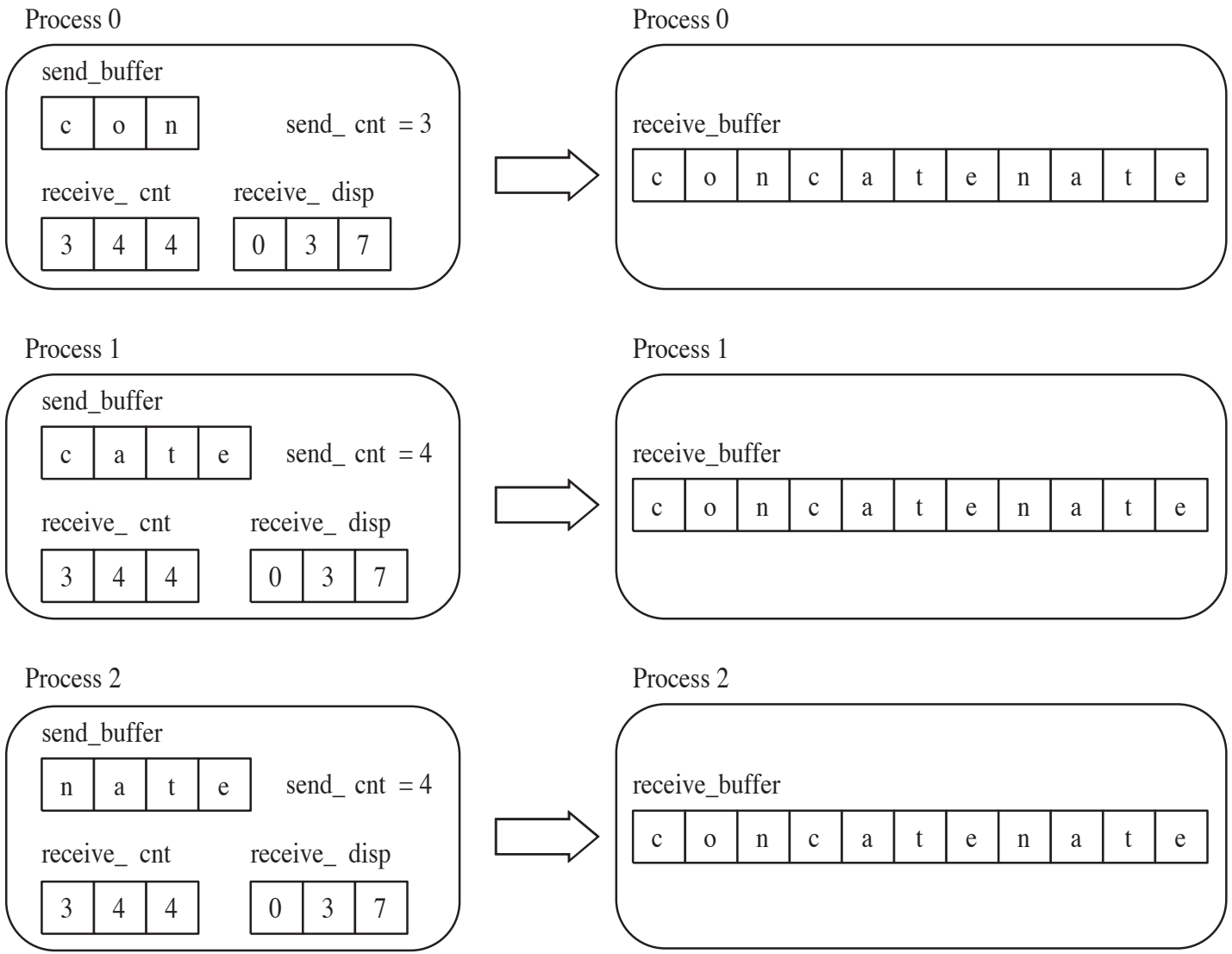
#itens q cada processo contribui (um inteiro)

#itens q espera de cada processo (um array)

índice onde colocar dados de cada processo

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Allgatherv in Action



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Columnwise Block Striped Matrix

- Partitioning through domain decomposition
- Task associated with
  - Column of matrix
  - Vector element

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Matrix-Vector Multiplication

$$
\begin{aligned}
c_0 &= a_{0,0}\,b_0 + a_{0,1}\,b_1 + a_{0,2}\,b_2 + a_{0,3}\,b_3 + a_{4,4}\,b_4 \\
c_1 &= a_{1,0}\,b_0 + a_{1,1}\,b_1 + a_{1,2}\,b_2 + a_{1,3}\,b_3 + a_{1,4}\,b_4 \\
c_2 &= a_{2,0}\,b_0 + a_{2,1}\,b_1 + a_{2,2}\,b_2 + a_{2,3}\,b_3 + a_{2,4}\,b_4 \\
c_3 &= a_{3,0}\,b_0 + a_{3,1}\,b_1 + a_{3,2}\,b_2 + a_{3,3}\,b_3 + b_{3,4}\,b_4 \\
c_4 &= a_{4,0}\,b_0 + a_{4,1}\,b_1 + a_{4,2}\,b_2 + a_{4,3}\,b_3 + a_{4,4}\,b_4
\end{aligned}
$$

Proc 4

Proc 3

Proc 2

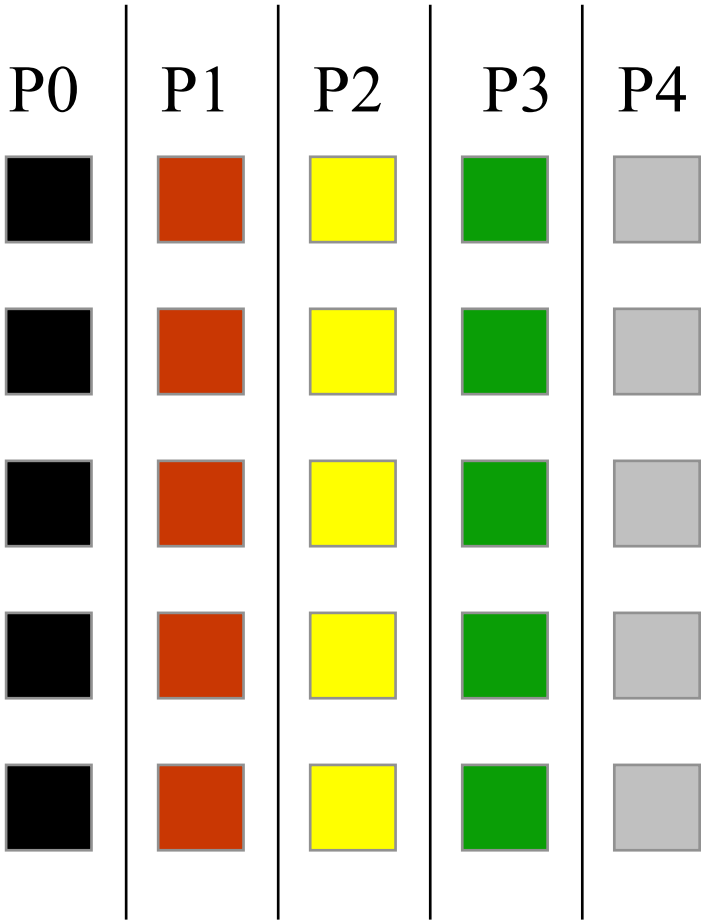
Processor 1's initial computation

Processor 0's initial computation

evita replicação do vetor B

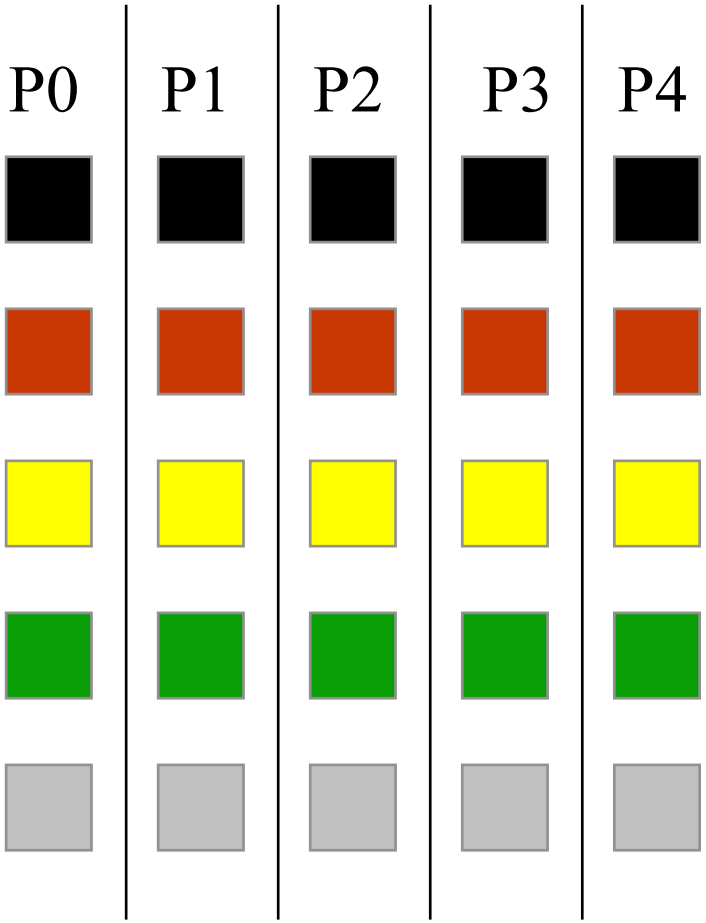
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

All-to-all Exchange (before)



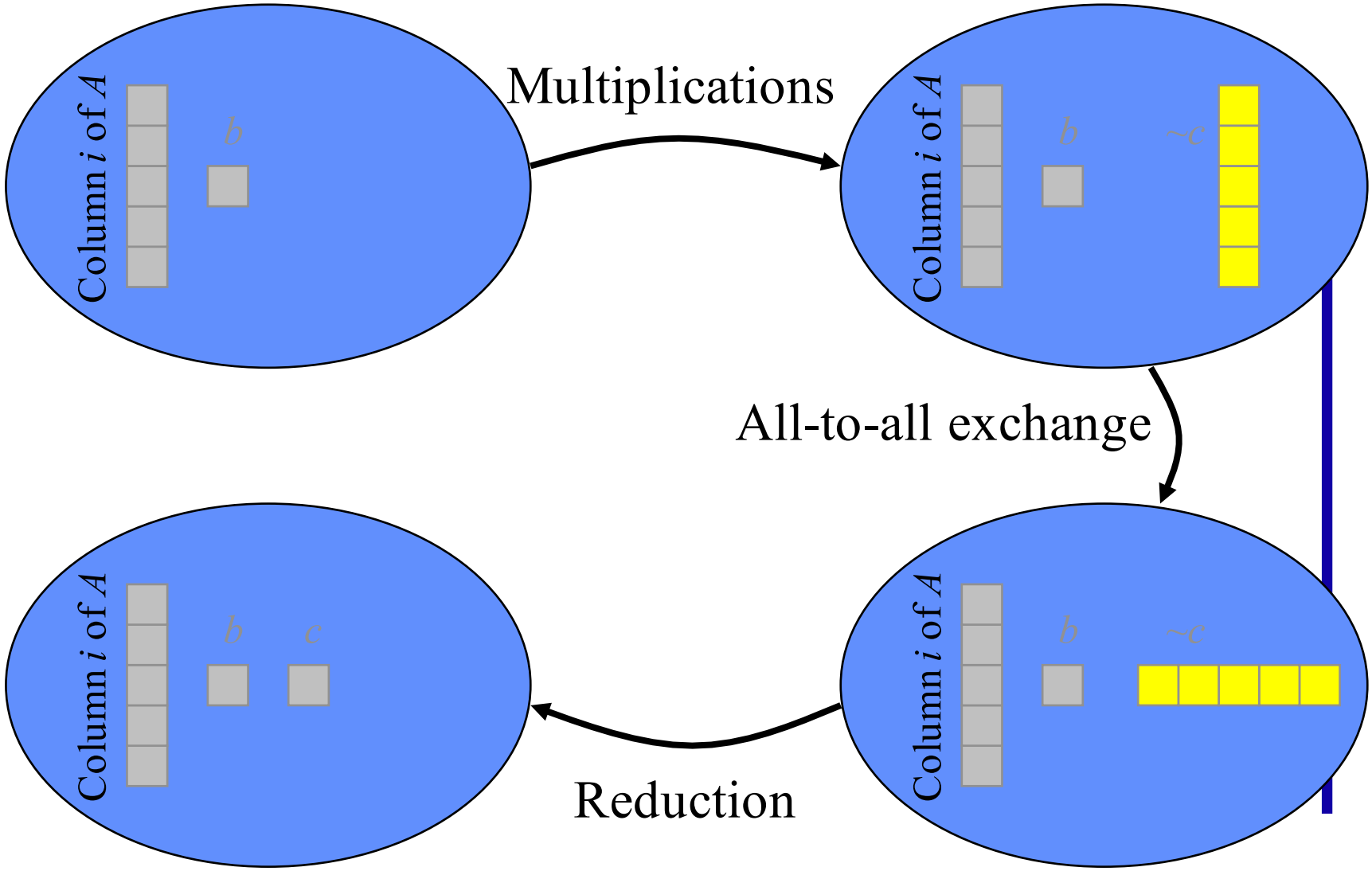
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

All-to-all Exchange (after)



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Phases of Parallel Algorithm



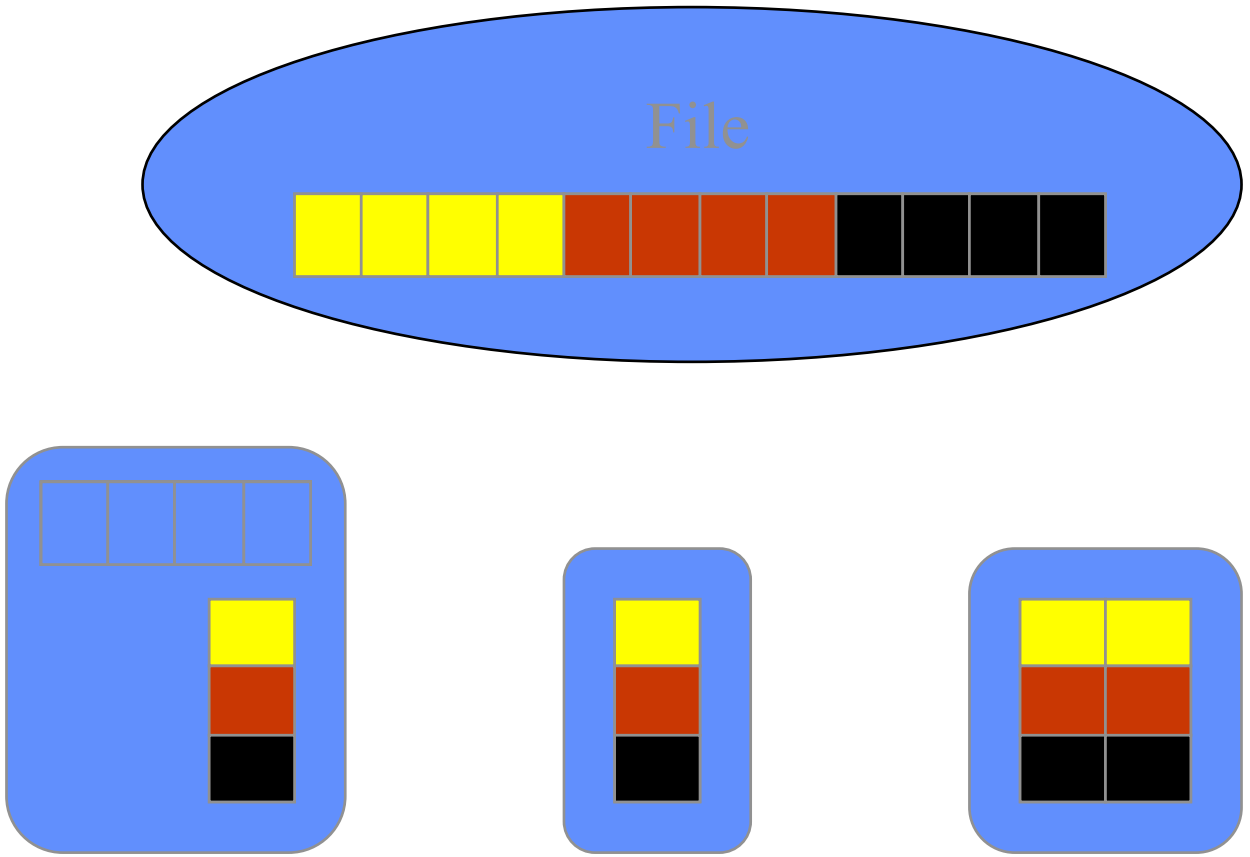
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Agglomeration and Mapping

- Static number of tasks
- Regular communication pattern (all-to-all)
- Computation time per task is constant
- Strategy:
  - Agglomerate groups of columns
  - Create one task per MPI process

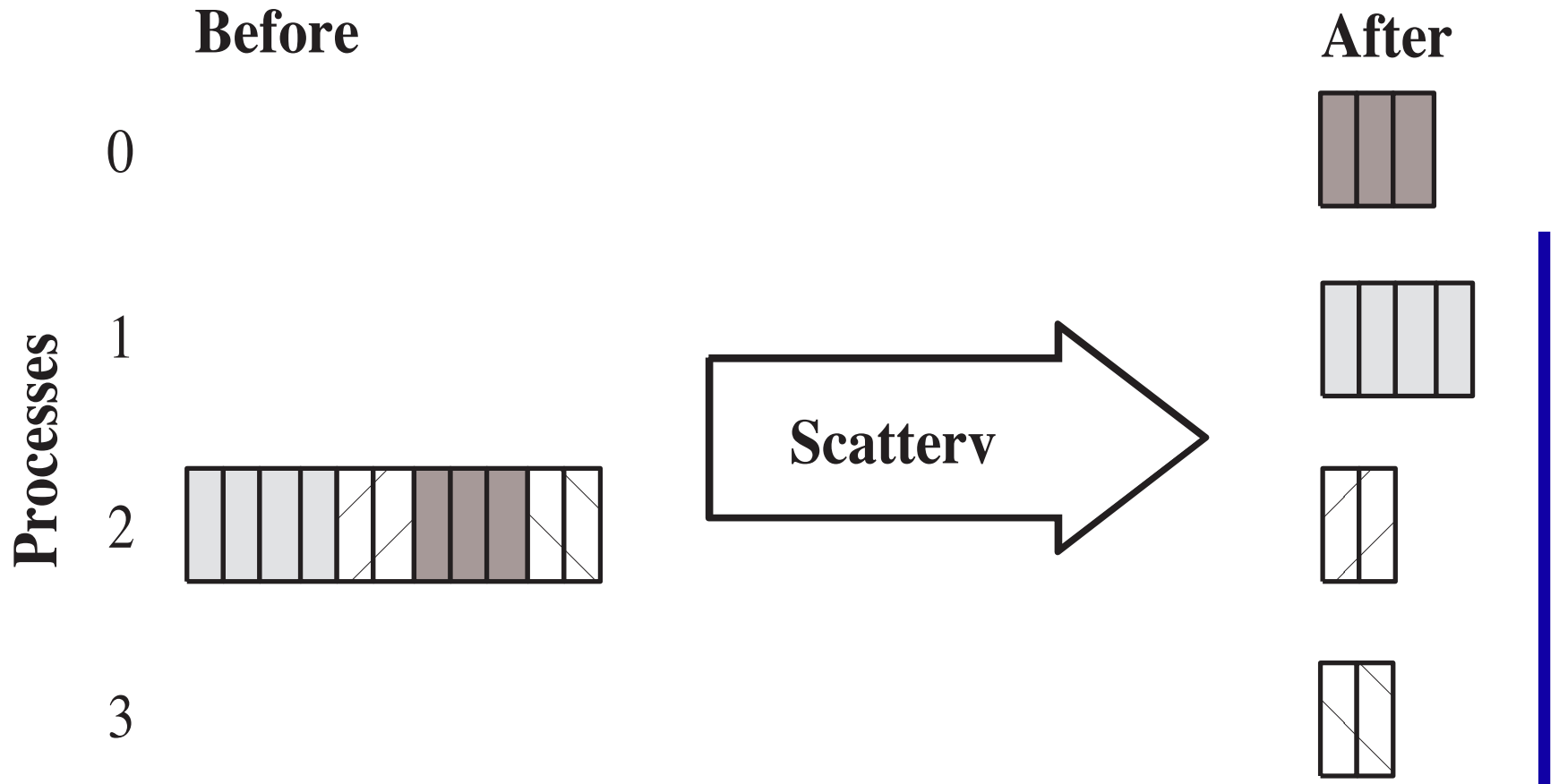
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Reading a Block-Column Matrix



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Scatterv



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Header for MPI_Scatterv

```
int MPI_Scatterv (
    void            *send_buffer,
    int             *send_cnt,
    int             *send_disp,
    MPI_Datatype     send_type,
    void            *receive_buffer,
    int              receive_cnt,
    MPI_Datatype     receive_type,
    int              root,
    MPI_Comm         communicator)
```

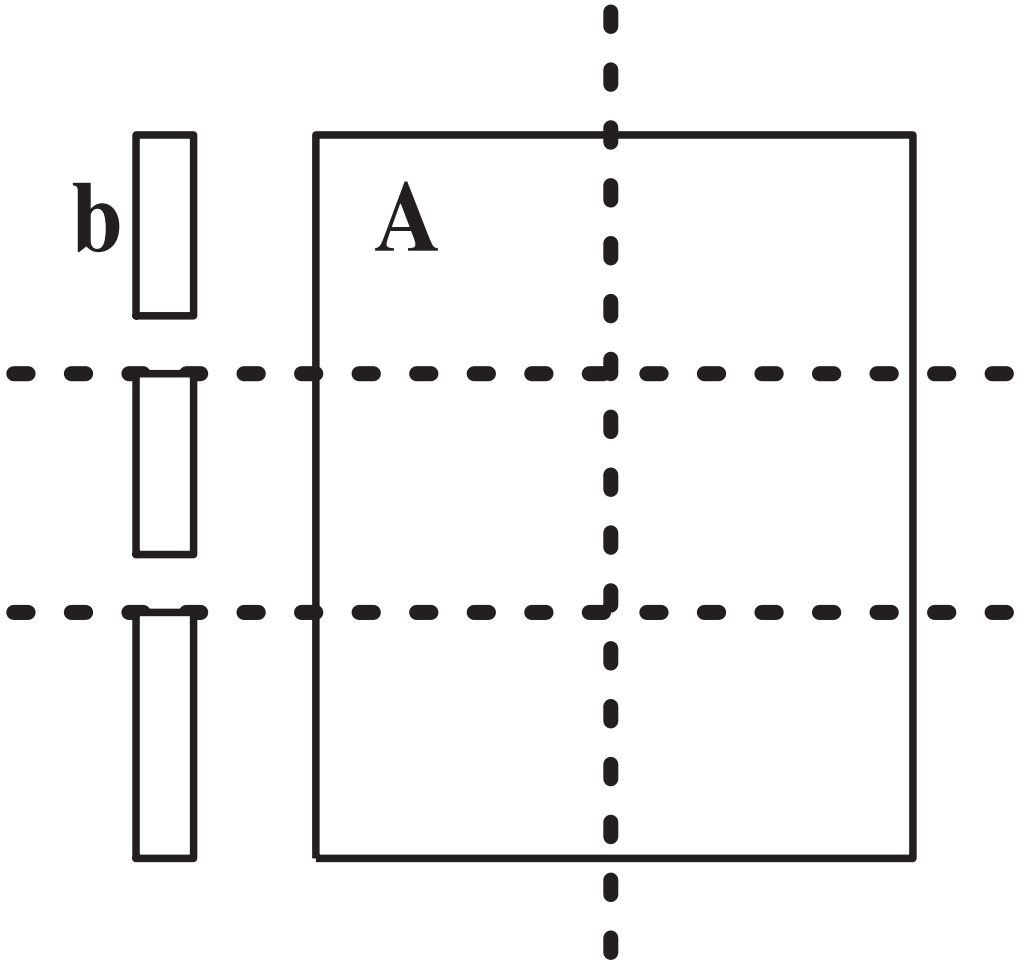
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Checkerboard Block Decomposition

- Associate primitive task with each element of the matrix **A**
- Each primitive task performs one multiply
- Agglomerate primitive tasks into rectangular blocks
- Processes form a 2-D grid
- Vector **b** distributed by blocks among processes in first column of grid

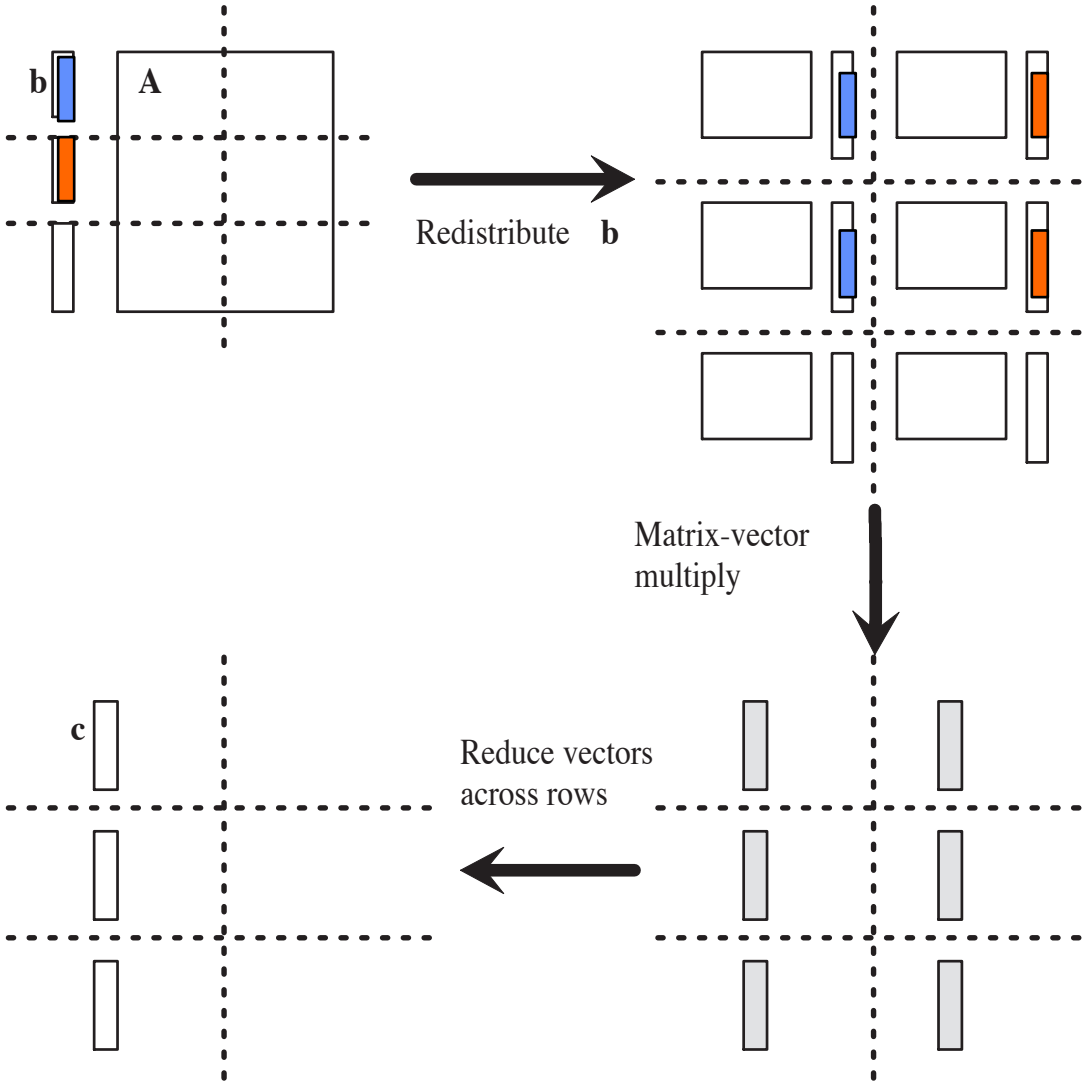
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Tasks after Agglomeration



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Algorithm's Phases



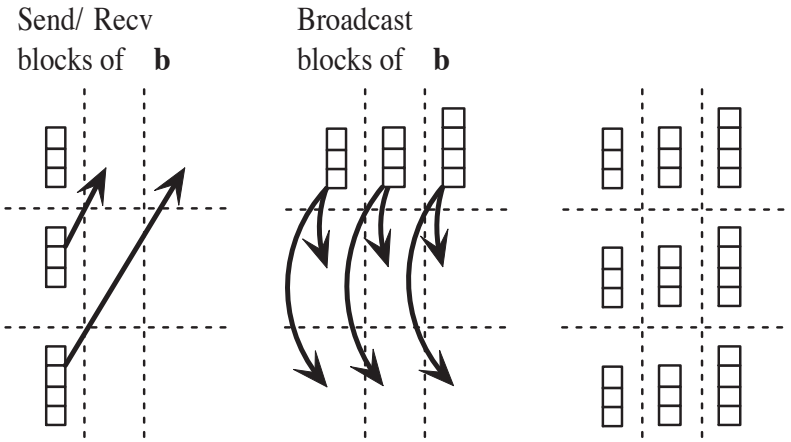
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Redistributing Vector b

- Step 1: Move **b** from processes in first row to processes in first column
  - If *p* square
    - » First column/first row processes send/receive portions of **b**
  - If *p* not square
    - » Gather **b** on process 0, 0
    - » Process 0, 0 broadcasts to first row procs
- Step 2: First row processes scatter **b** within columns

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Redistributing Vector b



(a)

when *p* is a square number

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Creating Communicators

- Want processes in a virtual 2-D grid
- Create a custom communicator to do this
- Collective communications involve all processes in a communicator
- We need to do broadcasts, reductions among subsets of processes
- We will create communicators for processes in same row or same column

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

What's in a Communicator?

- Process group
- Context
- Attributes
  - Topology (lets us address processes another way)
  - Others we won't consider

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Creating 2-D Virtual Grid of Processes

- MPI_Dims_create
  - Input parameters
    - » Total number of processes in desired grid
    - » Number of grid dimensions
  - Returns number of processes in each dim
- MPI_Cart_create
  - Creates communicator with cartesian topology

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Dims_create

```
int MPI_Dims_create (
    int nodes,
        /* Input - Procs in grid */

    int dims,
        /* Input - Number of dims */

    int *size)
        /* Input/Output - Size of
           each grid dimension */
```

- cria um array de dimensões para uso posterior
- size: na entrada pode indicar uma dimensão desejada

```
int p = …; int size(2) = {0.0};
MPI_Dims_Create (p, 2, size);
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Cart_create

```
int MPI_Cart_create (
   MPI_Comm old_comm, /* Input - old communicator */

   int dims, /* Input - grid dimensions */

   int *size, /* Input - # procs in each dim */

   int *periodic,
       /* Input - periodic[j] is 1 if dimension j
          wraps around; 0 otherwise */

   int reorder,
       /* 1 if process ranks can be reordered */

   MPI_Comm *cart_comm)
       /* Output - new communicator */
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Using MPI_Dims_create and MPI_Cart_create

```
MPI_Comm cart_comm;
int p;
int periodic[2];
int size[2];
...
size[0] = size[1] = 0;
MPI_Dims_create (p, 2, size);
periodic[0] = periodic[1] = 0;
MPI_Cart_create (MPI_COMM_WORLD, 2, size,
                 1, &cart_comm);
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Useful Grid-related Functions

- MPI_Cart_rank
  - Given coordinates of process in Cartesian communicator, returns process rank
- MPI_Cart_coords
  - Given rank of process in Cartesian communicator, returns process' coordinates

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Header for MPI_Cart_rank

```
int MPI_Cart_rank (
    MPI_Comm comm,
        /* In - Communicator */
    int *coords,
        /* In - Array containing process'
                grid location */
    int *rank)
        /* Out - Rank of process at
           specified coords */
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Header for MPI_Cart_coords

```
int MPI_Cart_coords (
    MPI_Comm comm,
        /* In - Communicator */
    int rank,
        /* In - Rank of process */
    int dims,
        /* In - Dimensions in virtual grid */
    int *coords)
        /* Out - Coordinates of specified
           process in virtual grid */
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

MPI_Comm_split

- Partitions the processes of a communicator into one or more subgroups
- Constructs a communicator for each subgroup
- Allows processes in each subgroup to perform their own collective communications
- Needed for columnwise scatter and rowwise reduce

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Header for MPI_Comm_split

```
int MPI_Comm_split (
    MPI_Comm old_comm,
        /* In - Existing communicator */

    int partition, /* In - Partition number */

    int new_rank,
        /* In - Ranking order of processes
           in new communicator */

    MPI_Comm *new_comm)
        /* Out - New communicator shared by
           processes in same partition */
```

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Example: Create Communicators for Process Rows

```
MPI_Comm grid_comm; /* 2-D process grid */

MPI_Comm grid_coords[2];
         /* Location of process in grid */

MPI_Comm row_comm;
                  /* Processes in same row */

MPI_Comm_split (grid_comm, grid_coords[0],
   grid_coords[1], &row_comm);
```