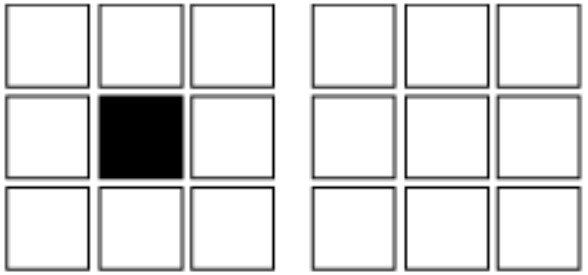
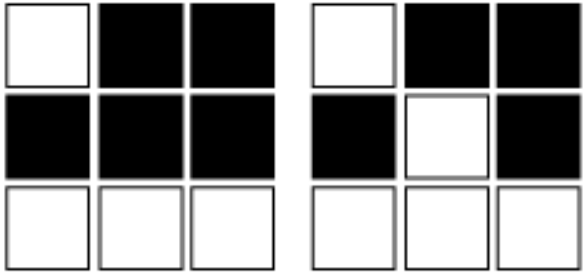
The Game of Life

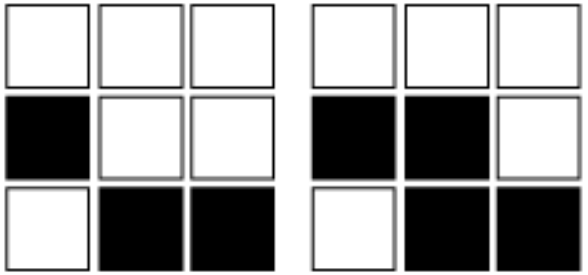
Each cell lives or dies based on the number of their neighbors.



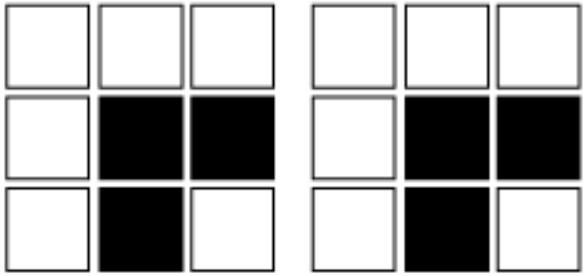
Loneliness:
A cell with less than 2 neighbors dies.



Overcrowding:
A cell with more than 3 neighbors dies.



Reproduction:
An empty cell with more than 3 neighbors comes to life.



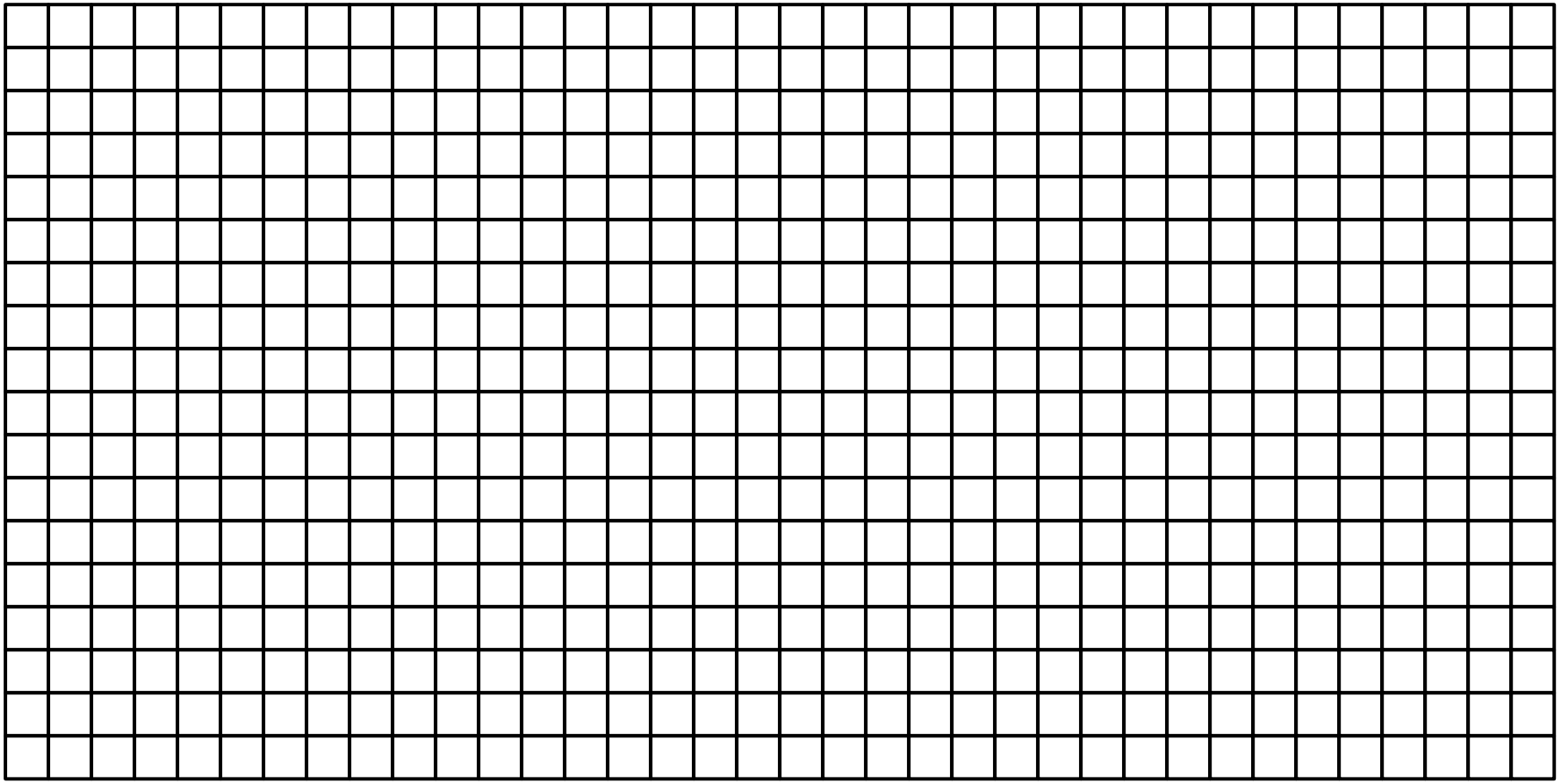
Stasis:
A cell with exactly 2 or 3 neighbors remains the alive.

Goal: Run in Parallel

- The serial code operates on a 2D grid of variable size.
- Our goal is to split the grid among a number of MPI tasks and compute many game of life generations.
- A large amount of bookkeeping may be required.

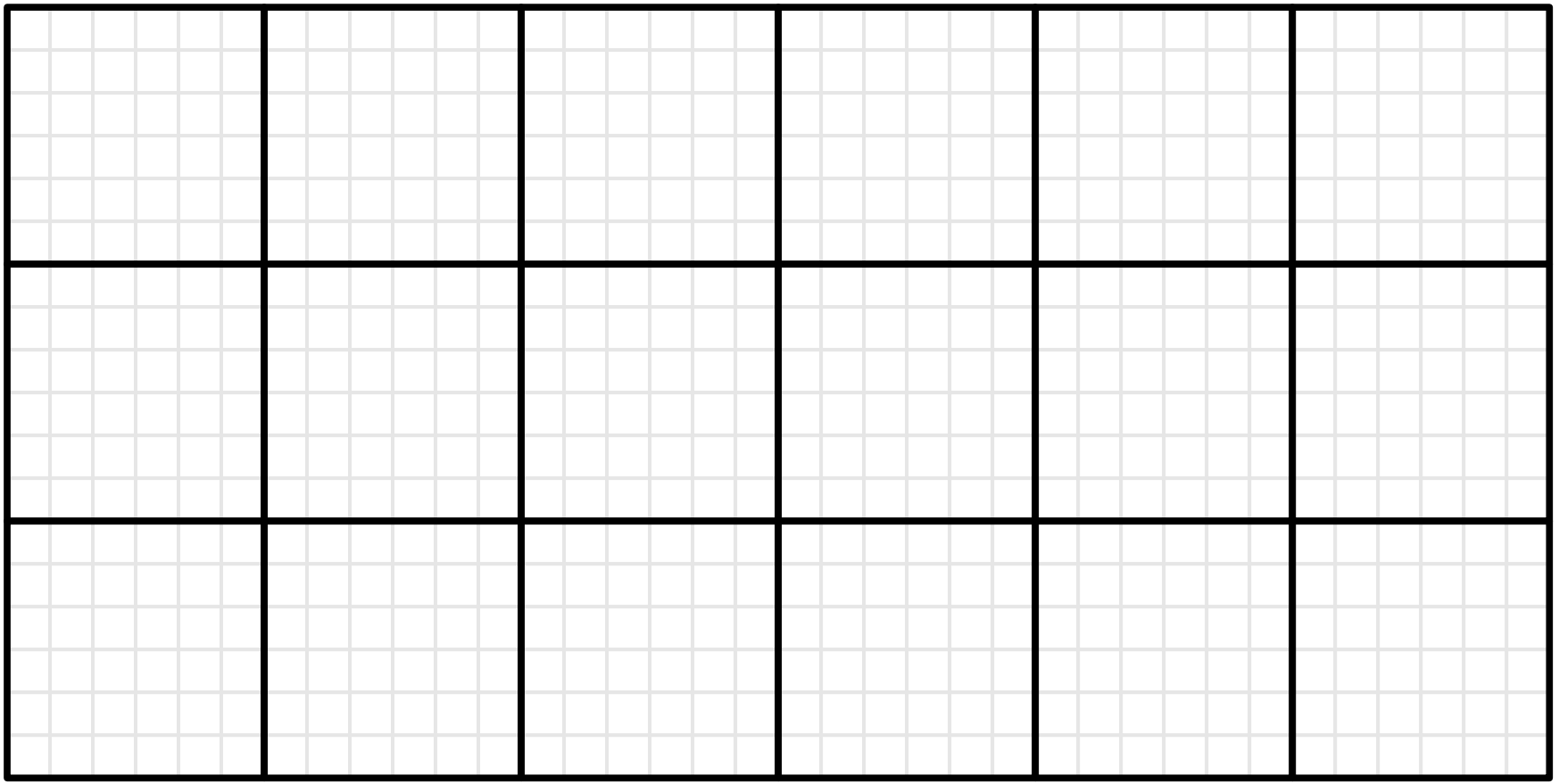
Domain Decomposition

Lets say we have 18 processors , how can we split our large grid up?



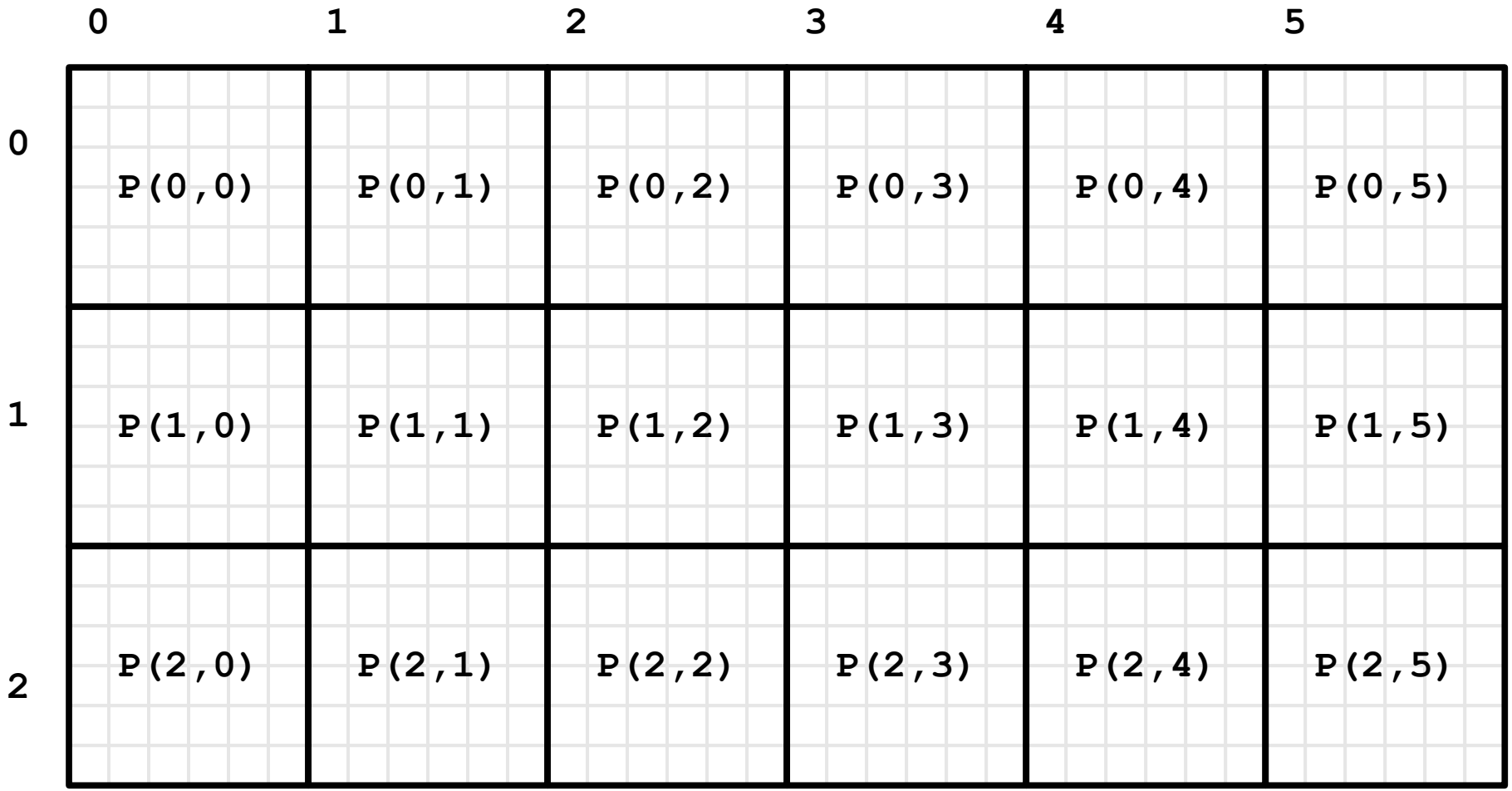
Domain Decomposition

Divide the grid into 18 segments.



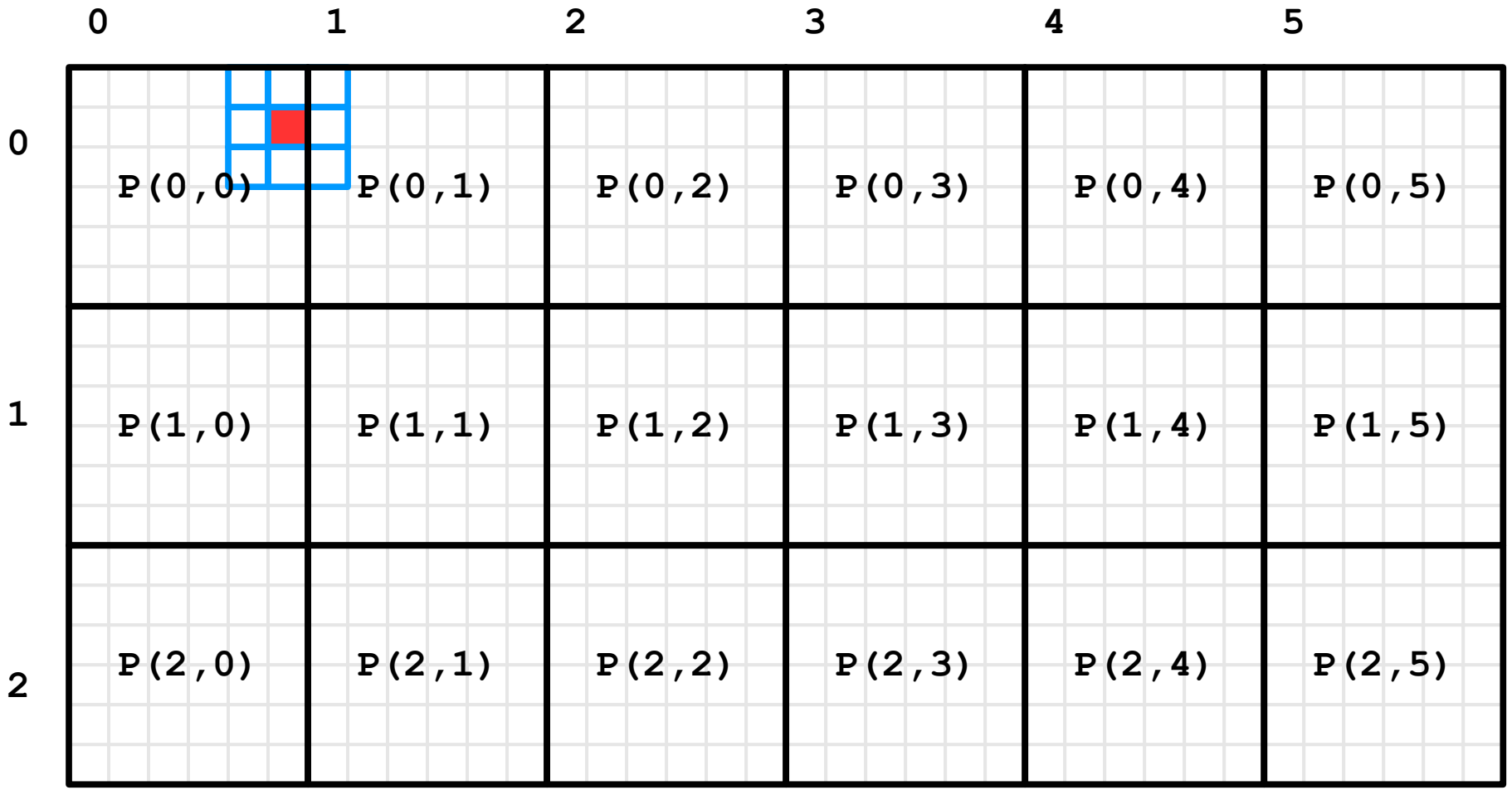
Domain Decomposition

Assign each segment to a processor.



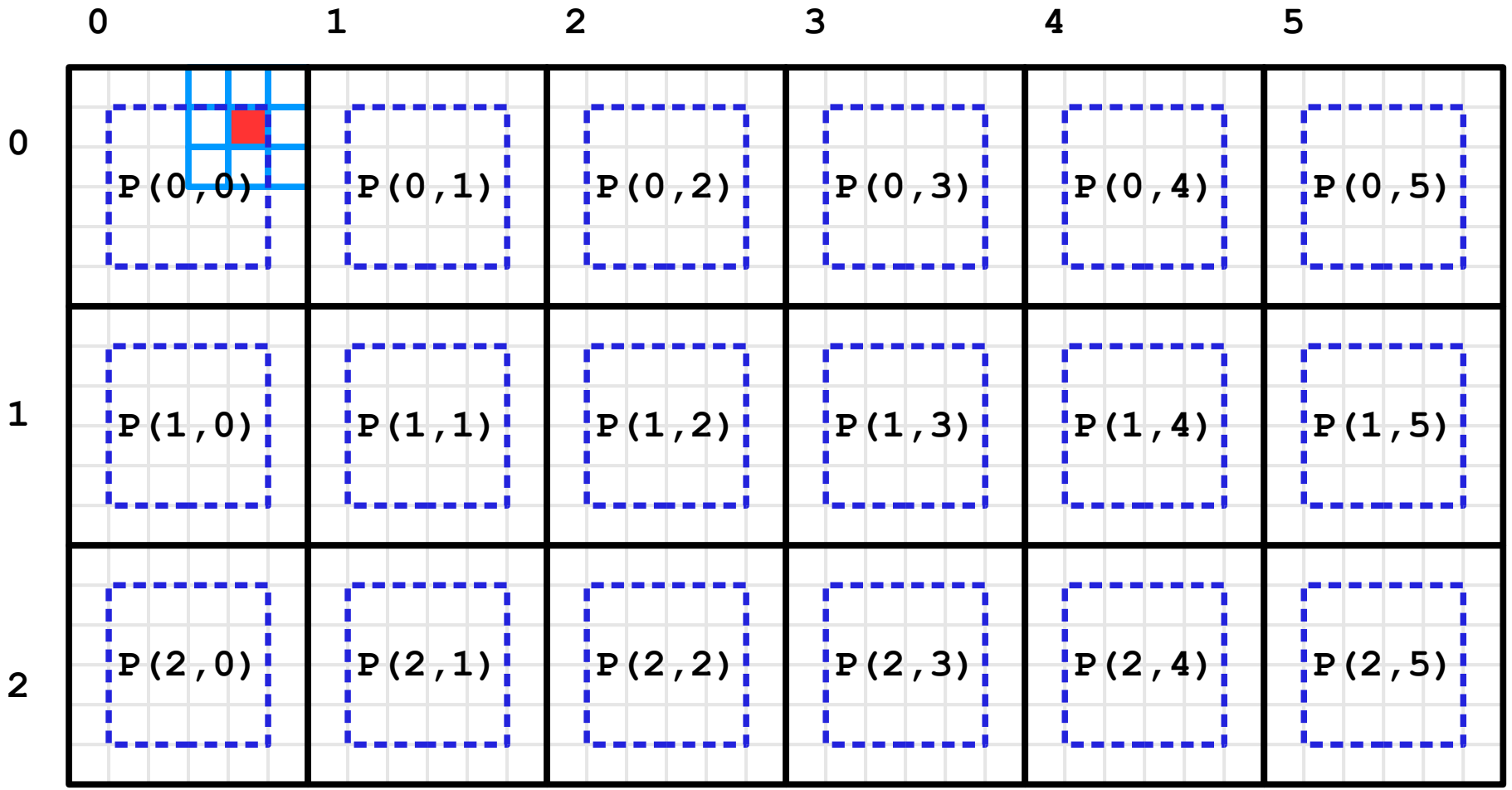
Domain Decomposition

What happens when the red cell needs to check it's neighbors?
They are on another processor.



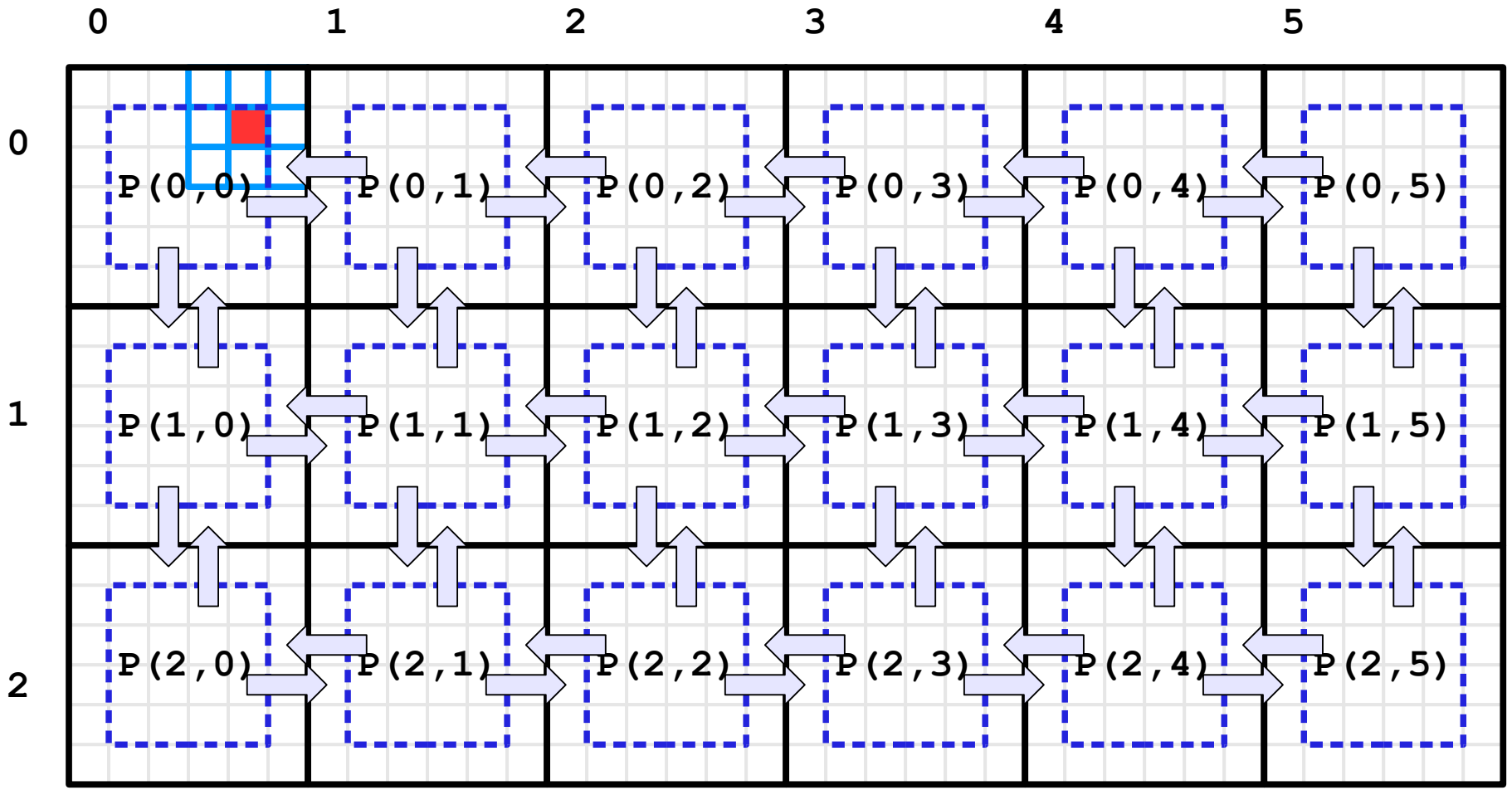
Domain Decomposition

Solution: Pad the data with ghost rows and columns.



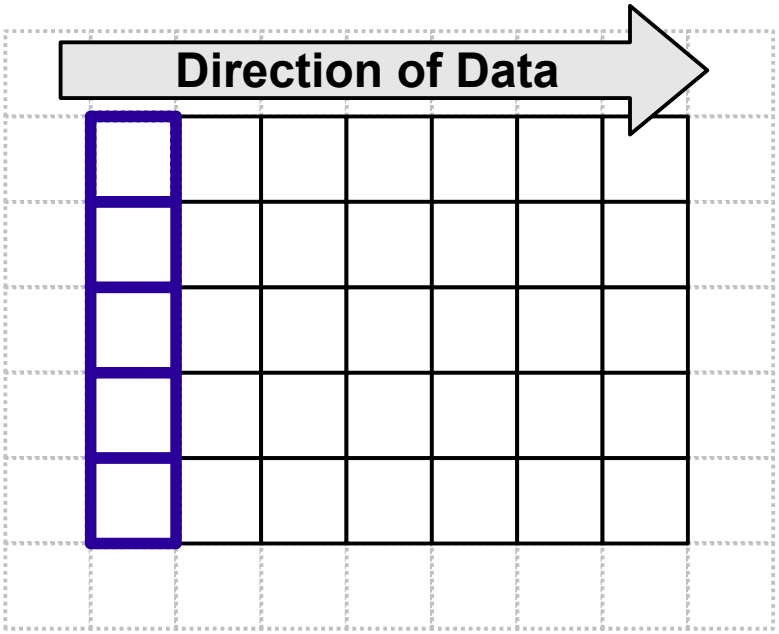
Domain Decomposition

Processors must now exchange boundary data.



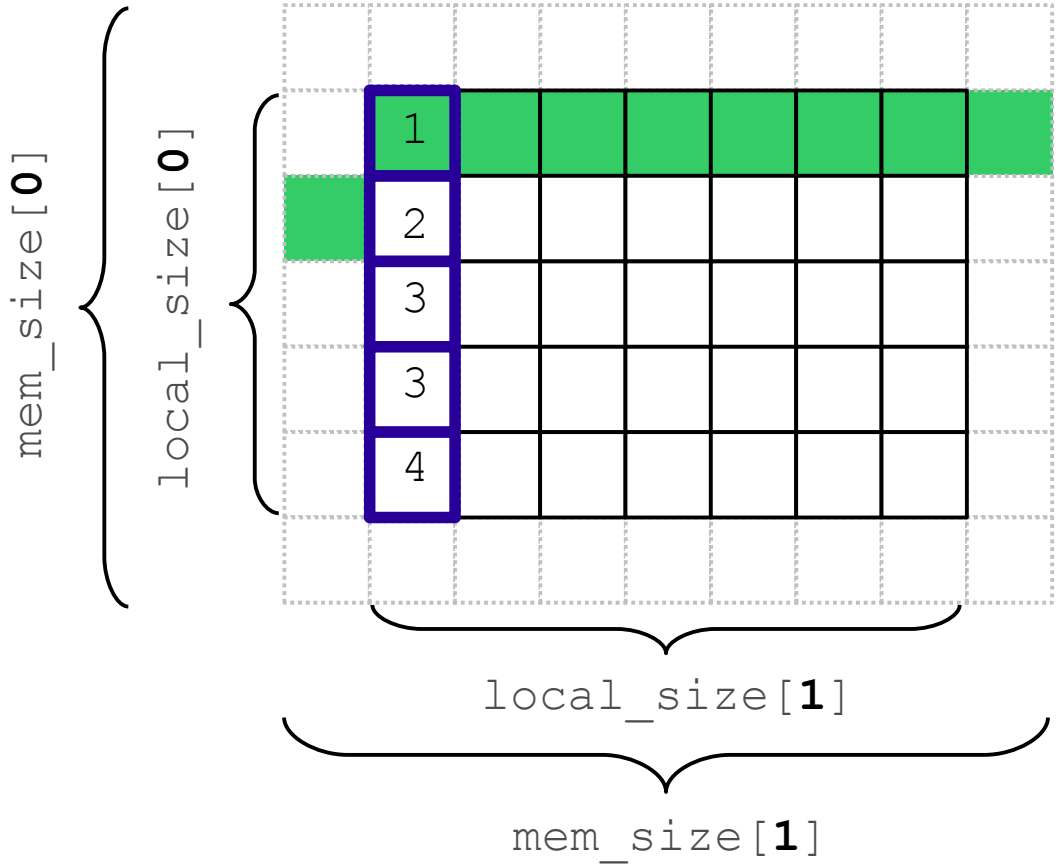
Column Data Type

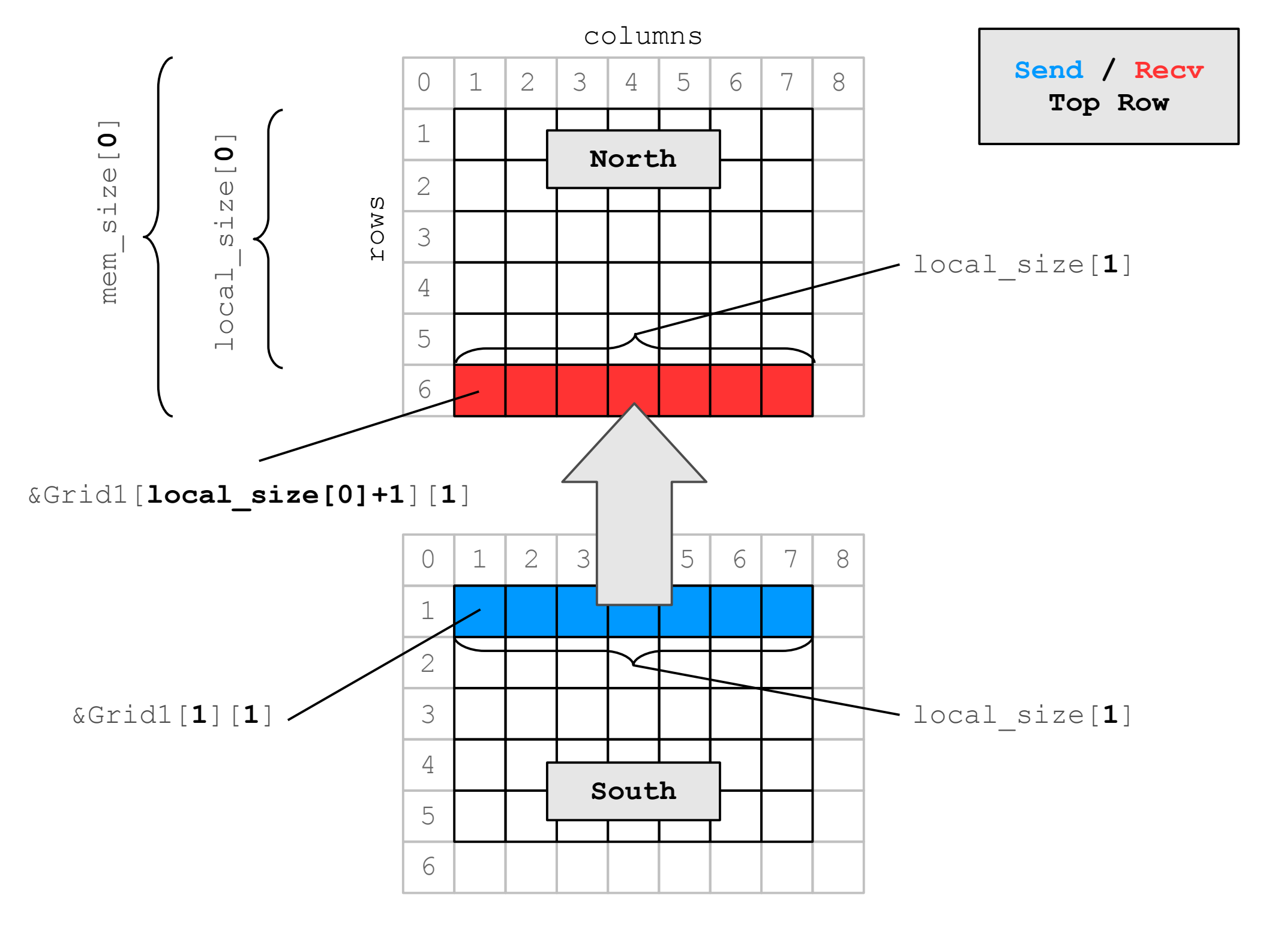
Column data is non-contiguous in memory.
We must construct a custom data type to send and receive columns.

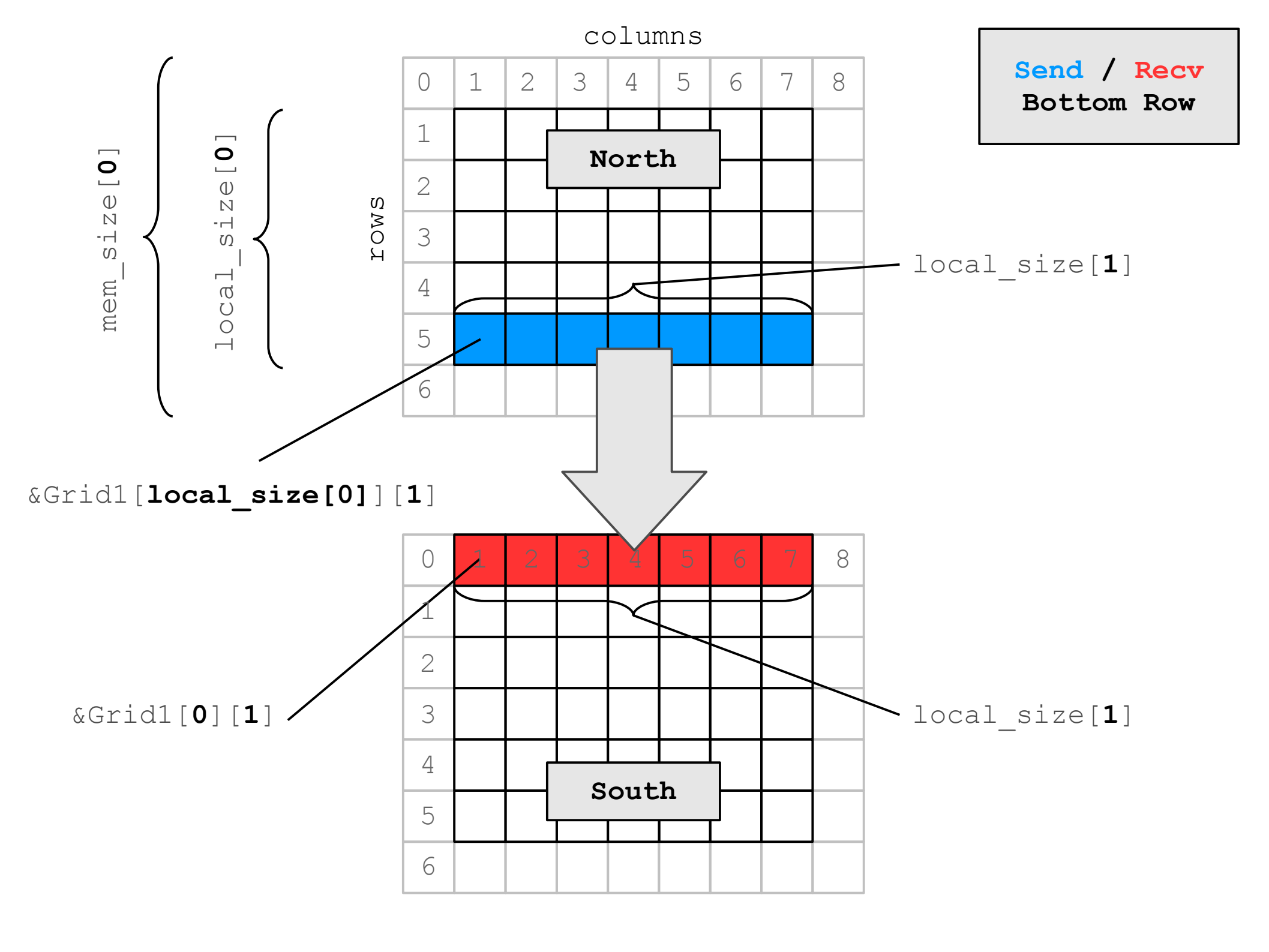


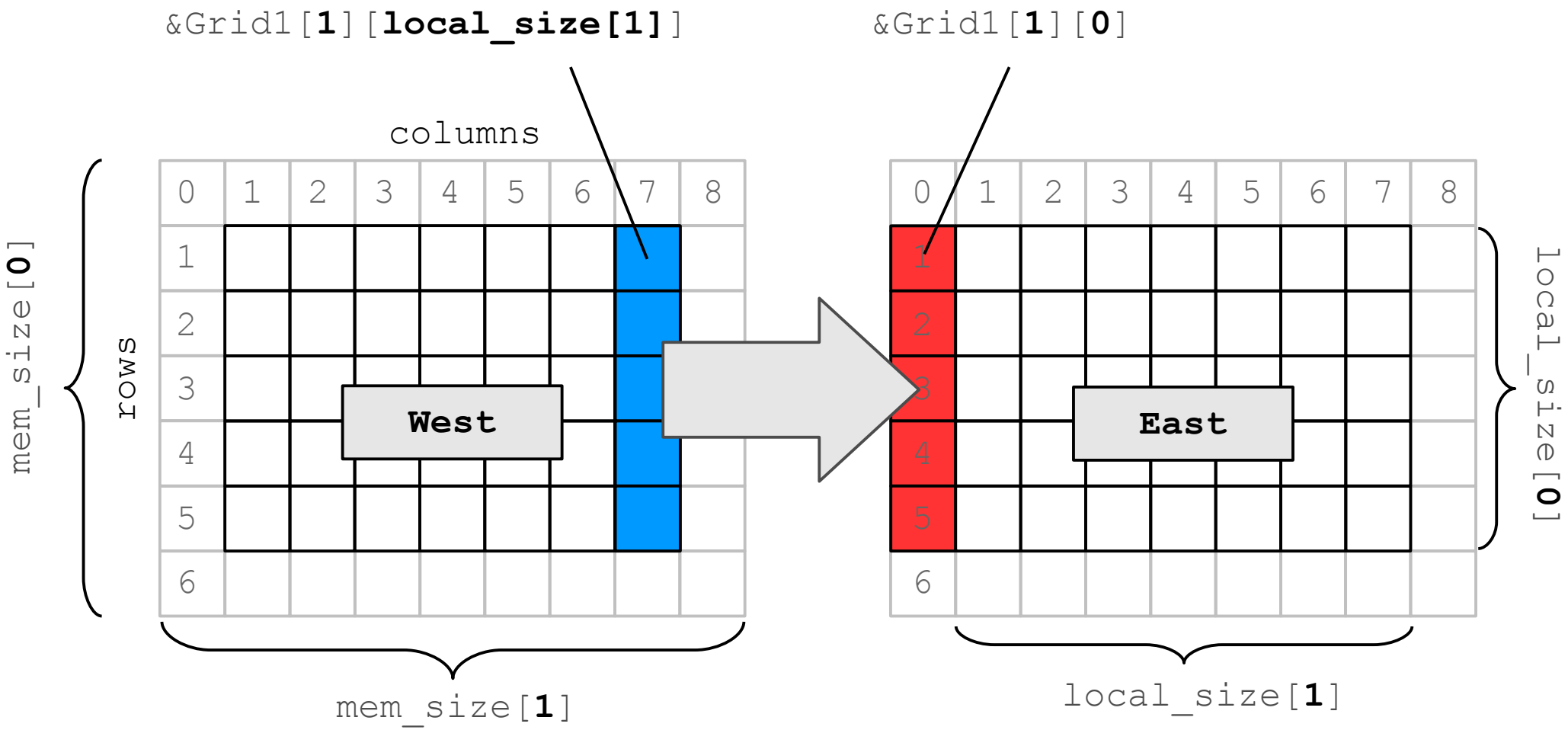
Column Data Type

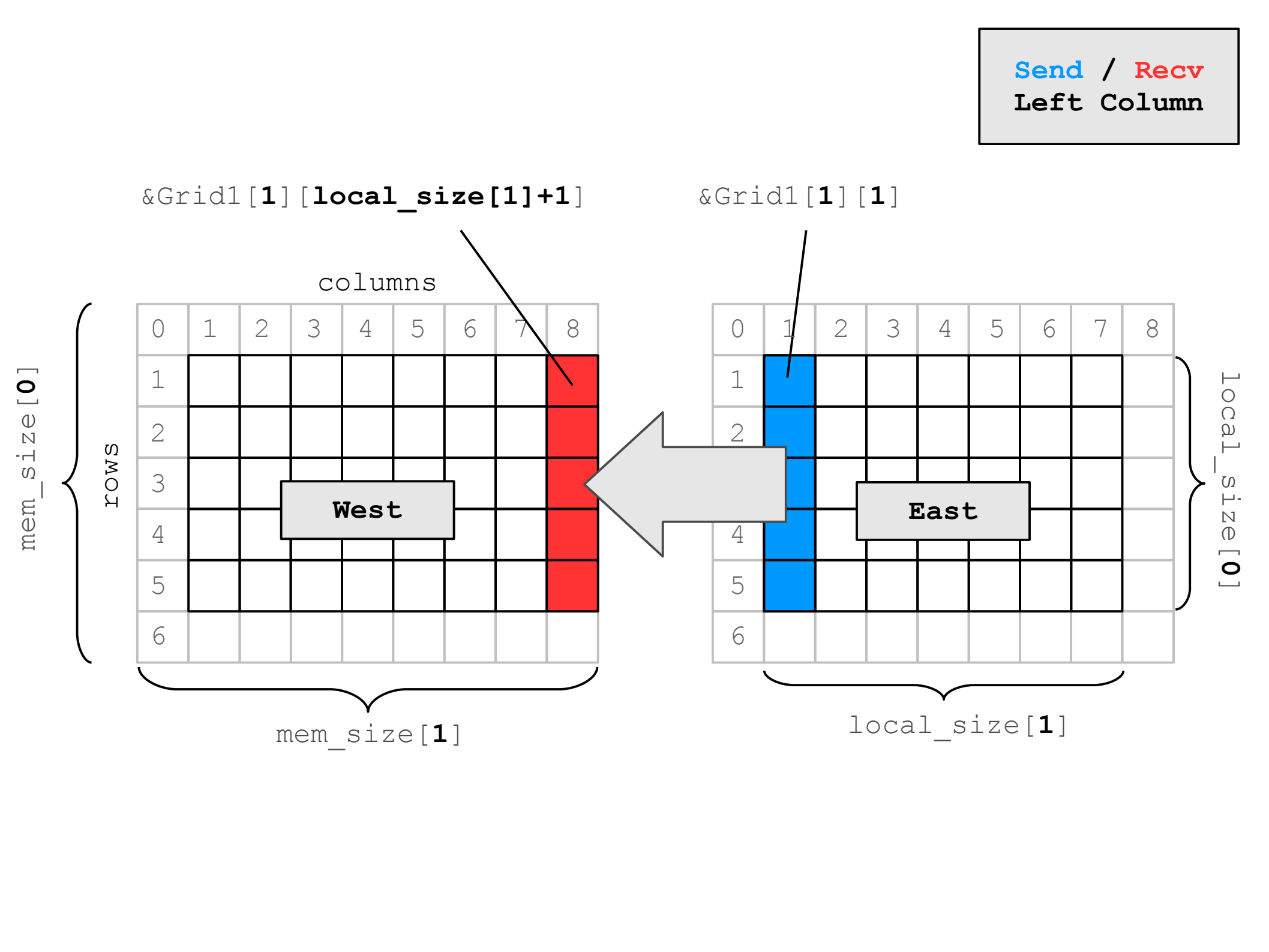
```
MPI_Type_vector(local_size[0], 1, mem_size[1], MPI_CHAR, &col_type);
                      |        |        |
                  Count,   Size,   Stride
```




columns
rows
0 1 2 3 4 5 6 7 8
1
2
3
4
5
6
North
South
mem_size[0]
local_size[0]
local_size[1]
&Grid1[local_size[0]+1][1]
&Grid1[1][1]
local_size[1]
Send / Recv
Top Row


columns
rows
0 1 2 3 4 5 6 7 8
1
2
3
4
5
6
North
South
Send / Recv
Bottom Row
mem_size[0]
local_size[0]
local_size[1]
&Grid1[local_size[0]][1]
&Grid1[0][1]
local_size[1]


Send / Recv
Right Column
&Grid1[1][local_size[1]]
&Grid1[1][0]
columns
rows
mem_size[0]
mem_size[1]
local_size[0]
local_size[1]
West
East
0 1 2 3 4 5 6 7 8
1 2 3 4 5 6


Send / Recv
Left Column
&Grid1[1][local_size[1]+1]
&Grid1[1][1]
columns
rows
mem_size[0]
mem_size[1]
local_size[0]
local_size[1]
West
East
0 1 2 3 4 5 6 7 8
1
2
3
4
5
6