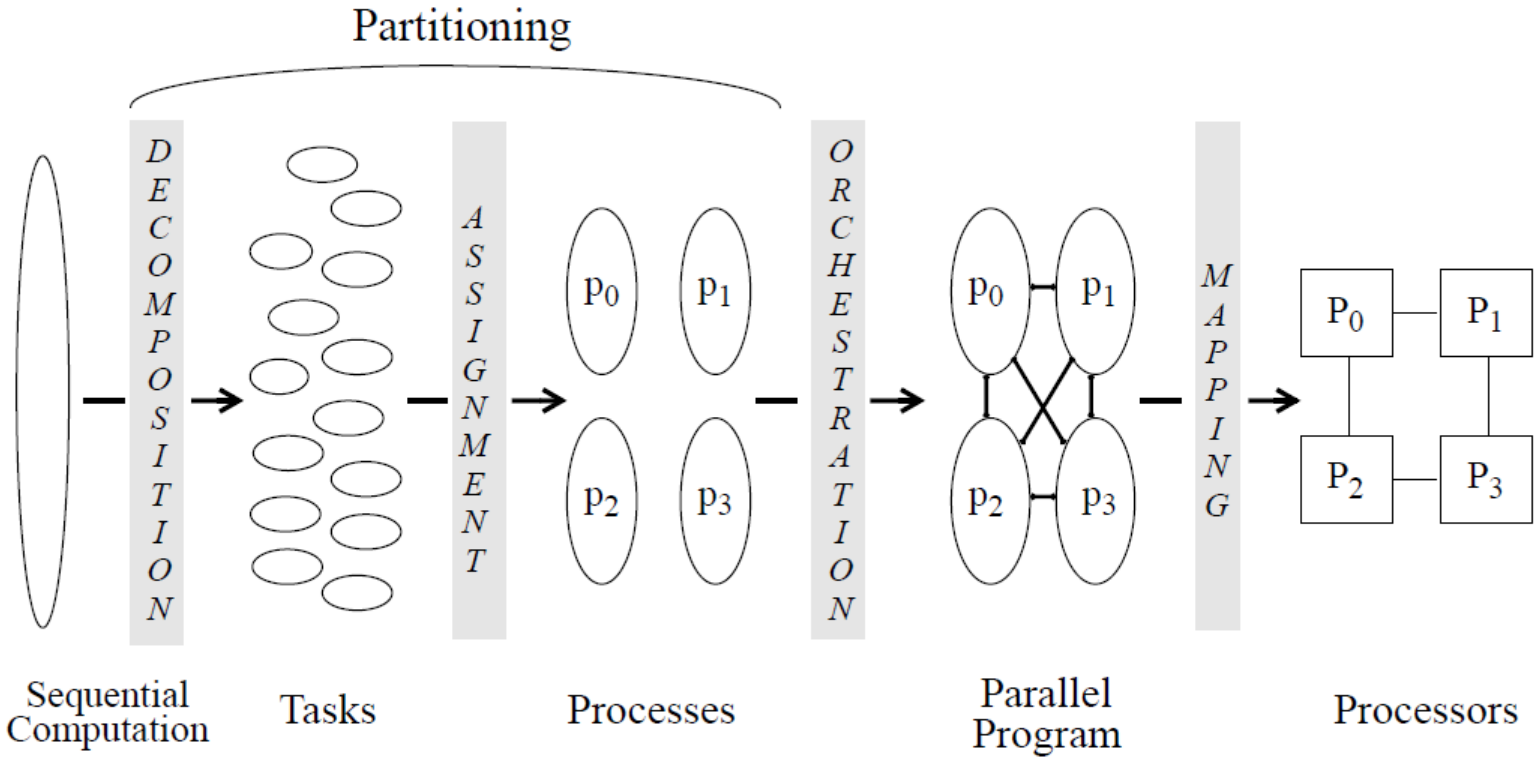
Parallelization

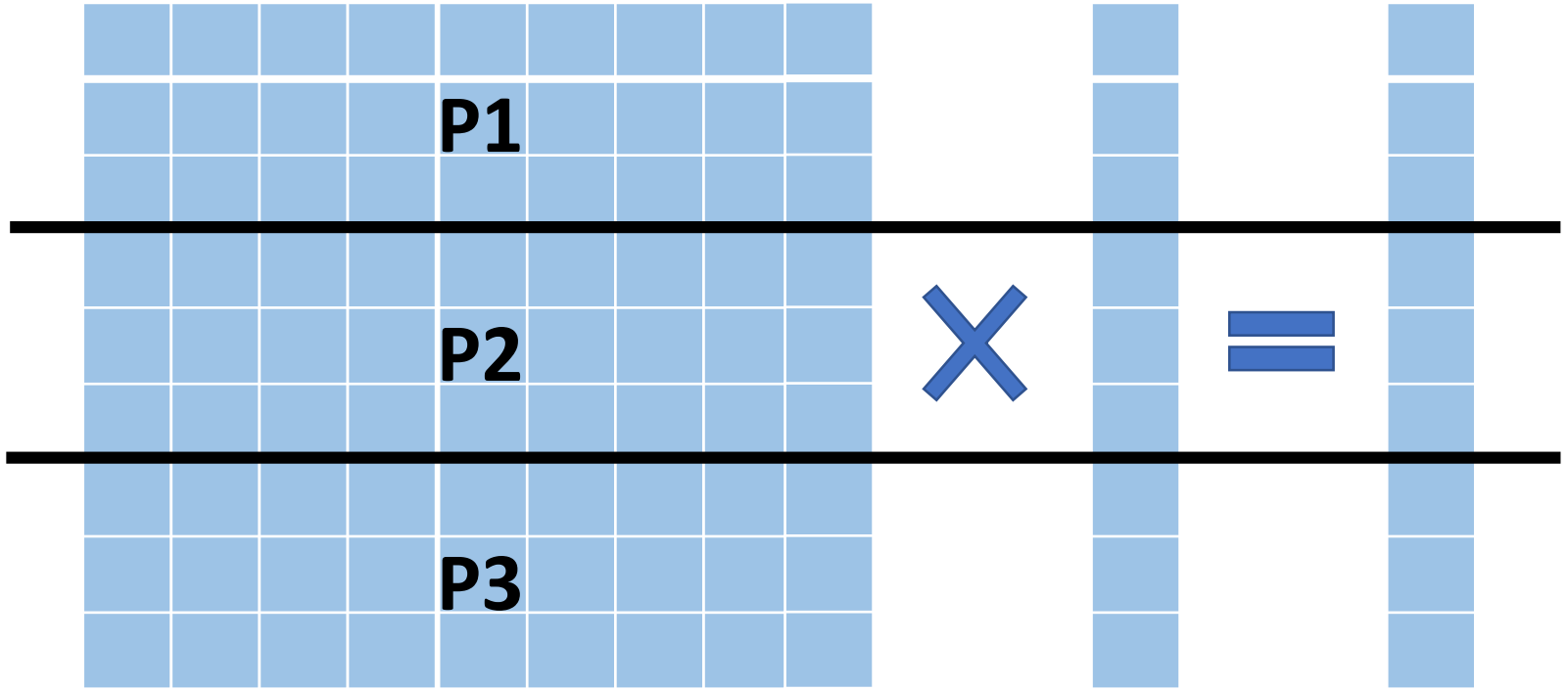
Jan 25, 2019

Illustration of Parallelization Steps



Source: Culler et al.

Parallelization – Matrix Vector Multiplication



P = 3

Decomposition

Assignment

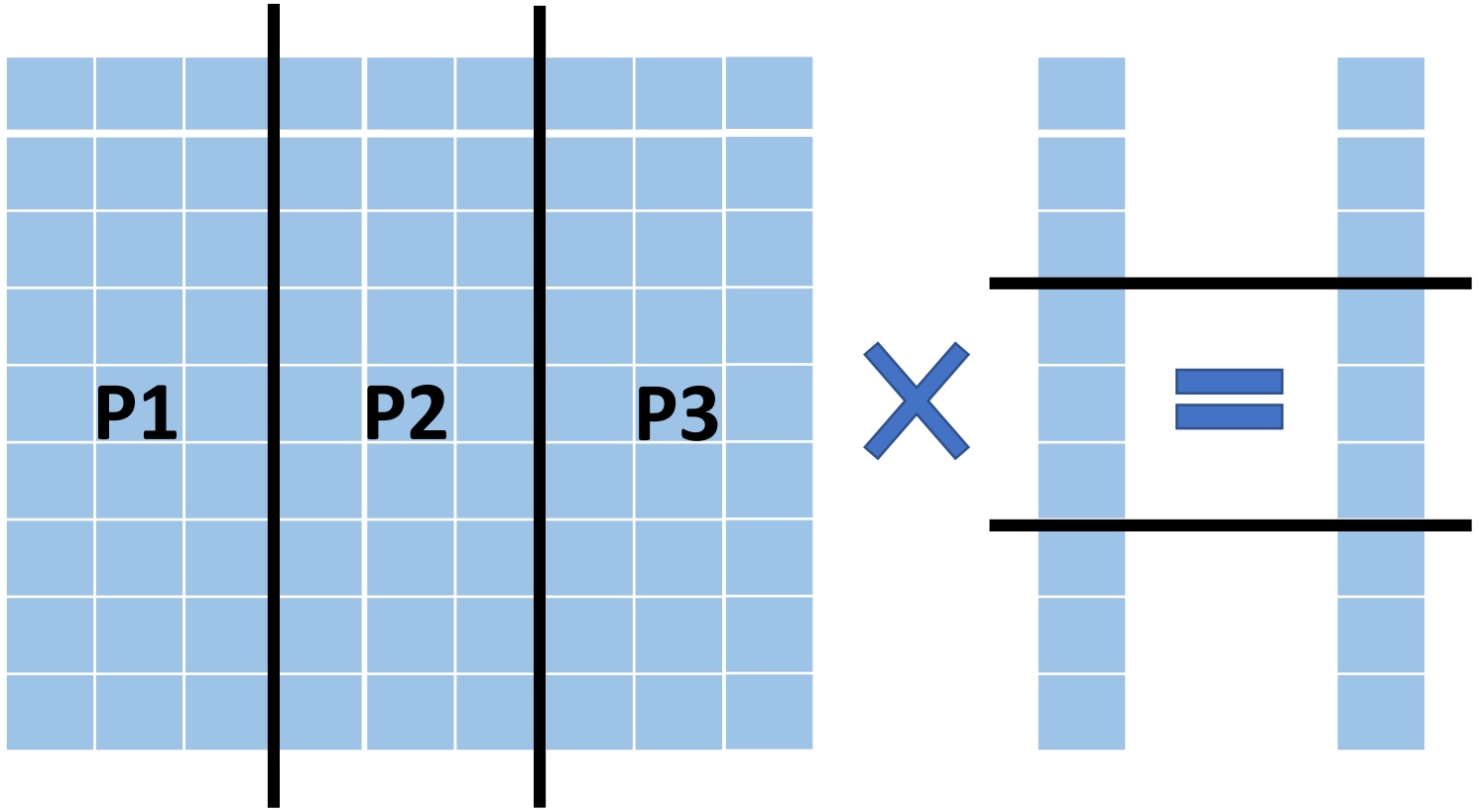
Orchestration

- Allgather
- Gather

Mapping

- What is the initial communication step ?
- Ways to distribute vector ?
- What are the differences between distribution and parallel reads?

Parallelization – Matrix Vector Multiplication



P = 3

Decomposition
Assignment
Orchestration
Mapping

What is the advantage of column-wise partitioning ?

Virtual Topology

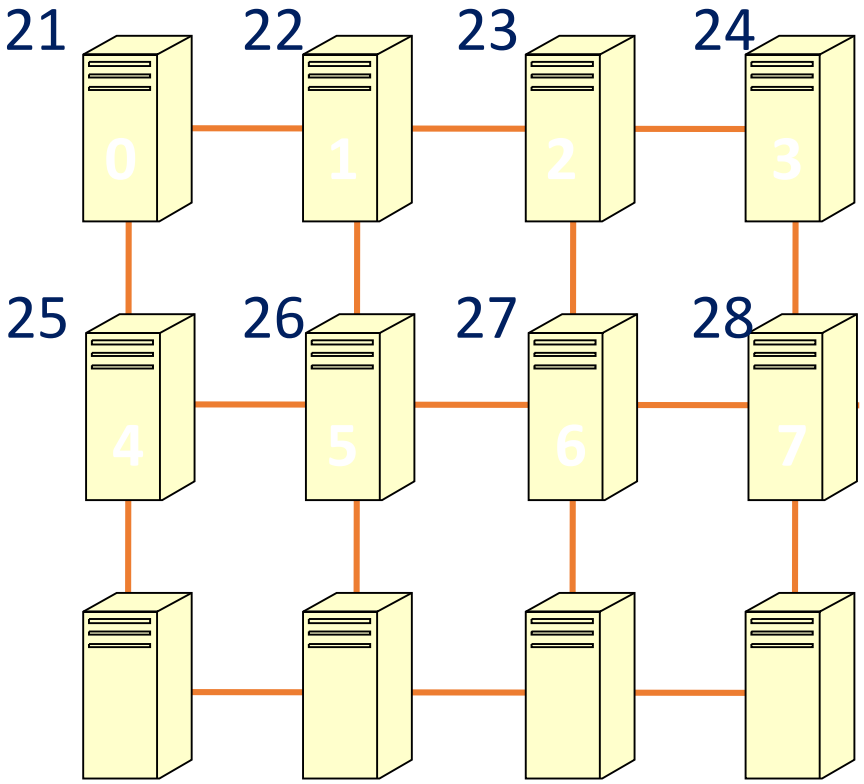
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

8 x 4 2D virtual process topology

- Communication pattern of MPI processes
- Graphical representation of communications
  - Nearest neighbor in a mesh
  - All-to-all
  - …
- Convenient way to represent communications

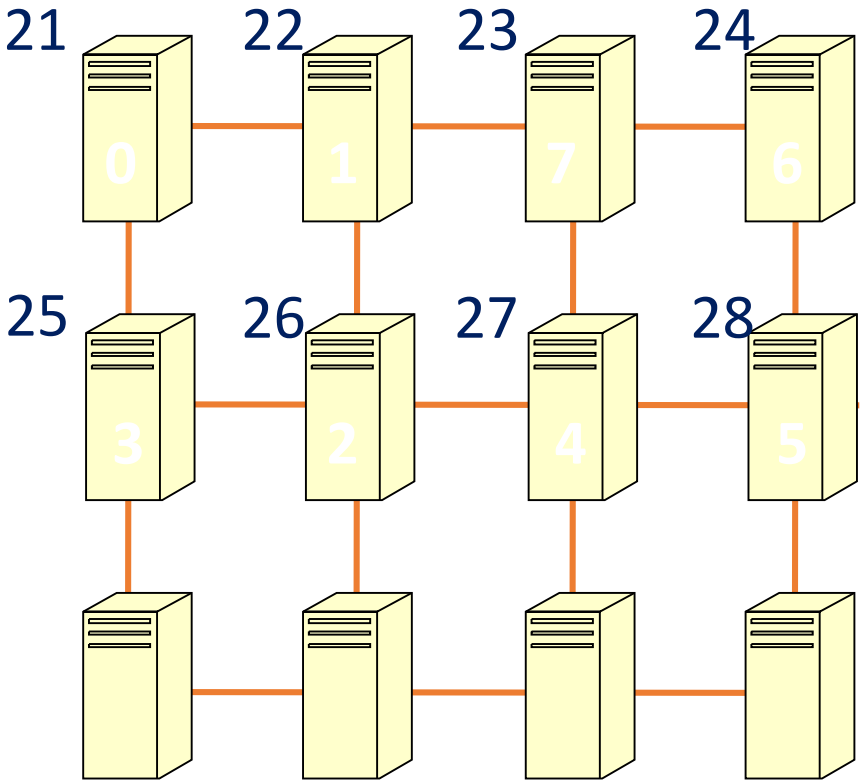
Note: Virtual topology set up before execution

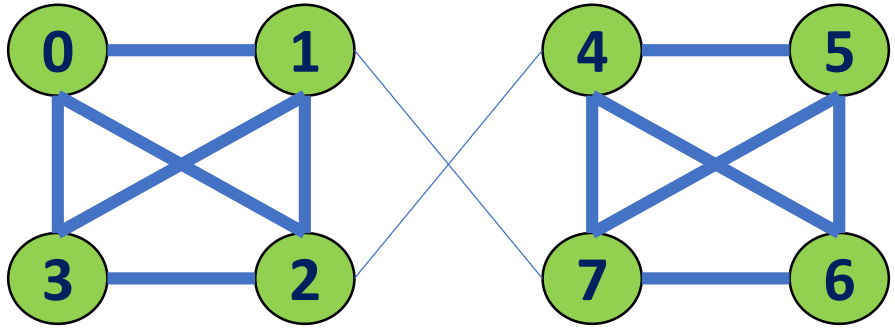
Physical Topology



- Connections between allocated cores
- Default placement of ranks based on node IDs
- Mapping: Placement of ranks onto cores
- Topology-aware mapping: Mapping that minimizes all communication times taking into account the physical topology

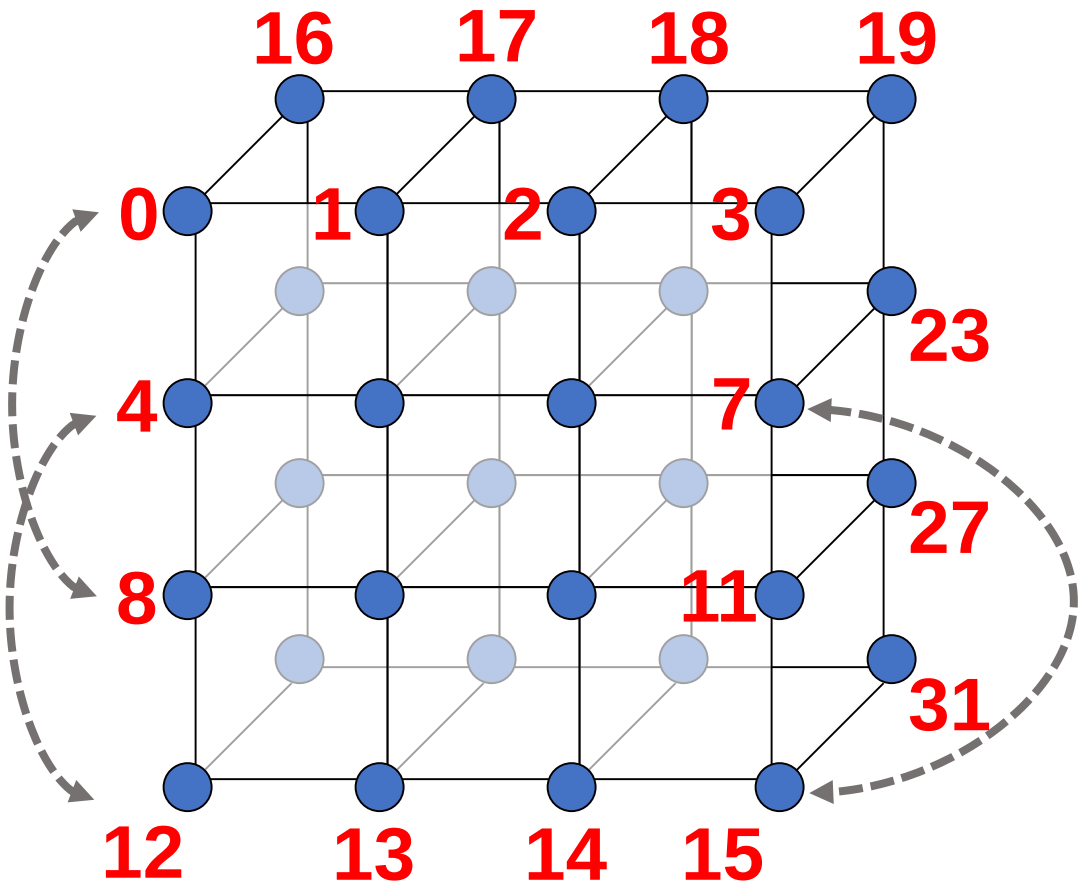
Rank placement





- May place ranks anywhere in the allocated nodes based on the communication pattern

Process-to-processor Mapping

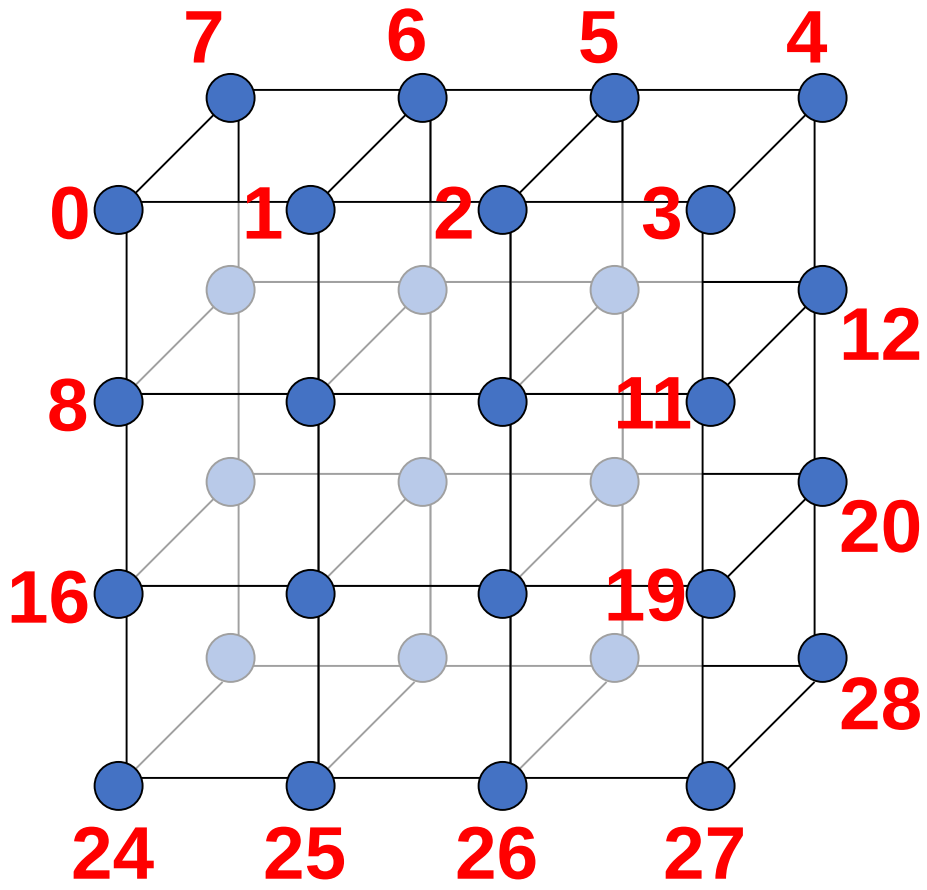


4 x 4 x 2 3D torus

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

8 x 4 2D virtual process topology

Topology-aware Mapping

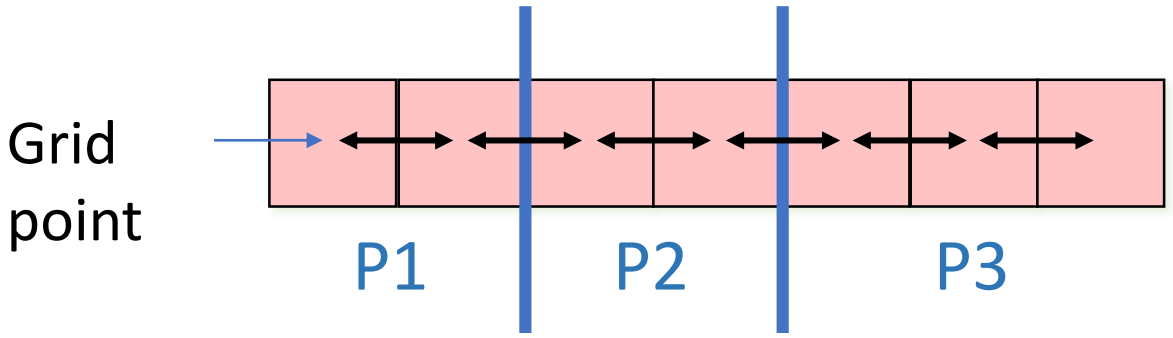


4 x 4 x 2 3D torus

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

8 x 4 2D virtual process topology

1D Domain decomposition



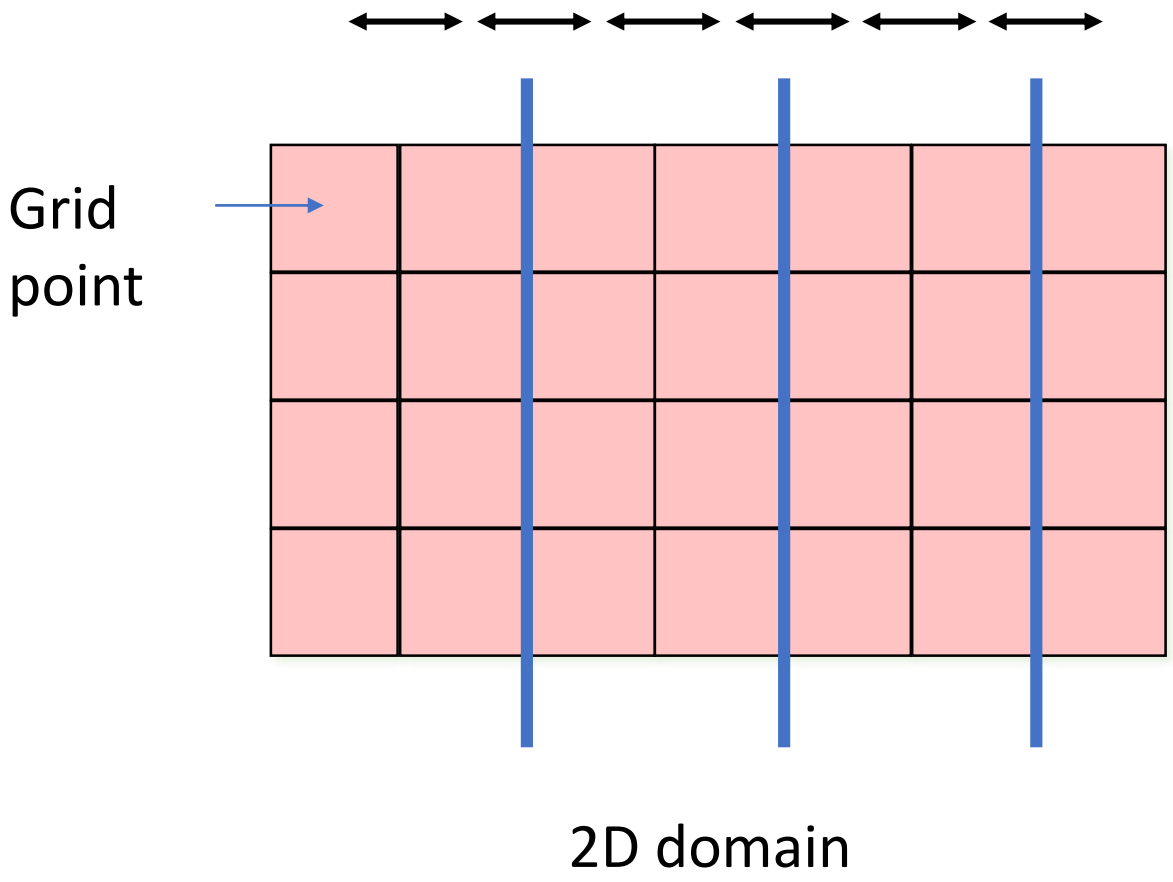
1D domain

N grid points
P processes
N/P points per process

Communications?

2 sends()
2 recvs()

1D Domain decomposition



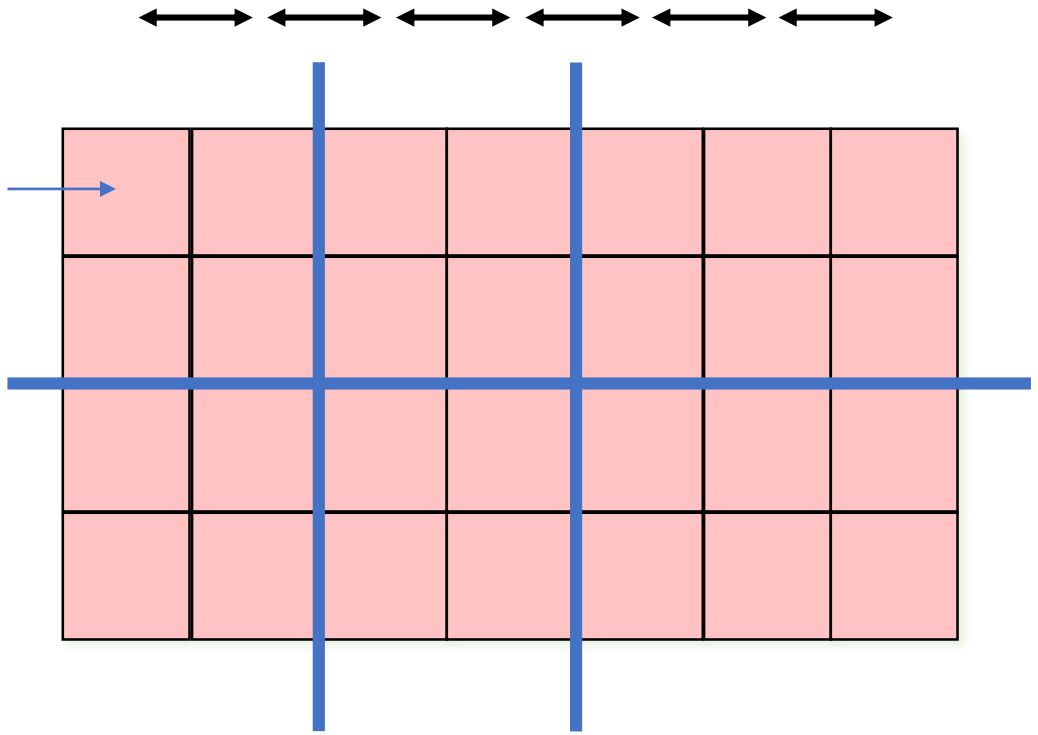
2D domain

N grid points
P processes
N/P points per process

Decomposition
Assignment
Orchestration
Mapping

Q1: Tunable parameters?
Q2: One drawback?

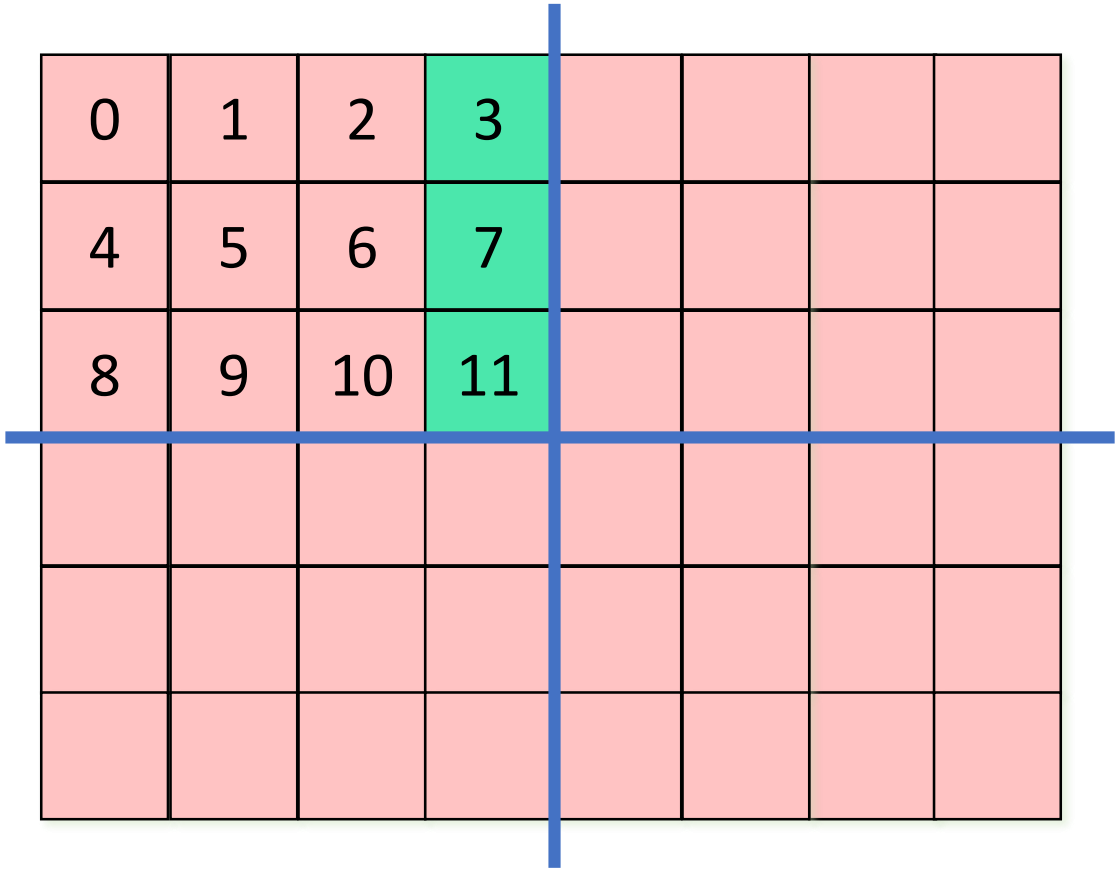
2D Domain decomposition



N grid points
P processes
N/P points per process

+ Several parallel communications
+ Lower communication volume/process

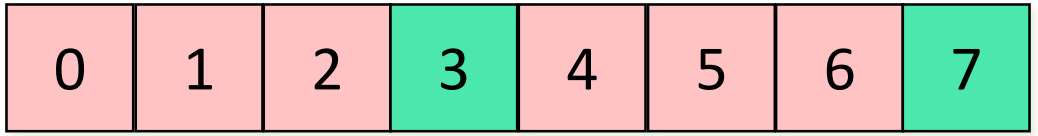
Send / Recv Options



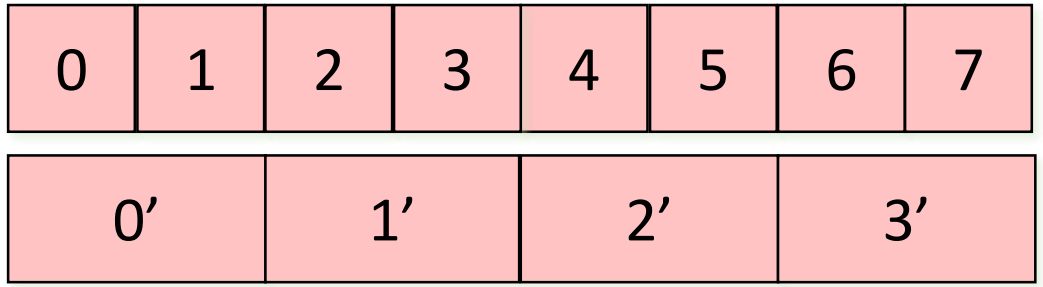
MPI_Pack (buf)
MPI_Pack (buf)
MPI_Pack (buf)
MPI_Send (buf)

MPI_Recv (buf)
MPI_Unpack (buf)
MPI_Unpack (buf)
MPI_Unpack (buf)

Create a new datatype

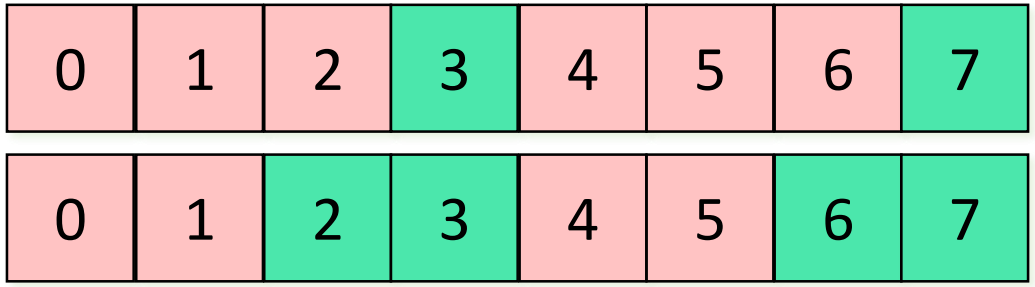


MPI Derived Datatypes



MPI_Type_contiguous

Count = ?



MPI_Type_vector

count = 2, blocklength = 1, stride = 4

MPI_Type_vector (count, blocklength, stride, MPI_INT, newtype)

MPI Derived Datatypes

MPI_Datatype newtype

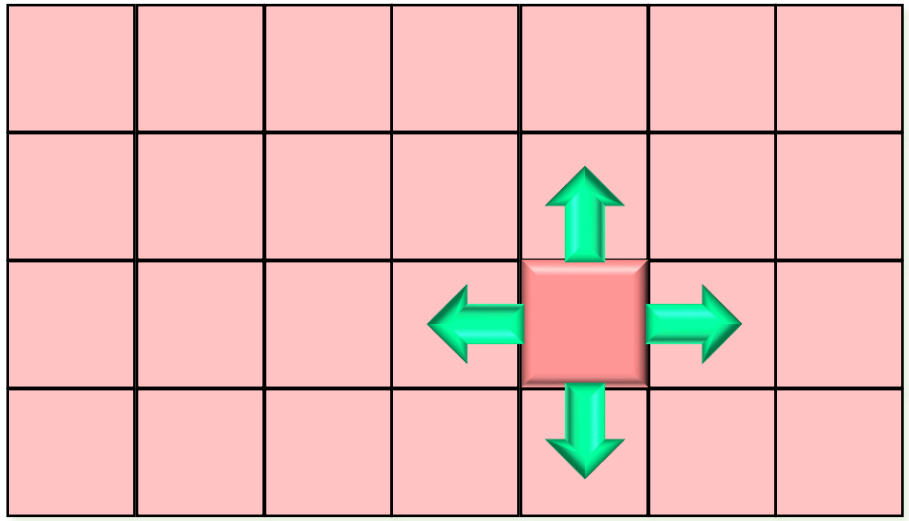
- MPI_Type_contiguous (count, oldtype, newtype)
- MPI_Type_vector (count, blocklength, stride, oldtype, newtype)
- MPI_Type_create_subarray (ndims, array_of_sizes, array_of_subsizes, array_of_starts, order, oldtype, newtype)
- MPI_Type_create_struct (count, array_of_blocklengths, array_of_displacements, array_of_types, newtype)

MPI_Type_commit (newtype)

……

MPI_Type_free (newtype)

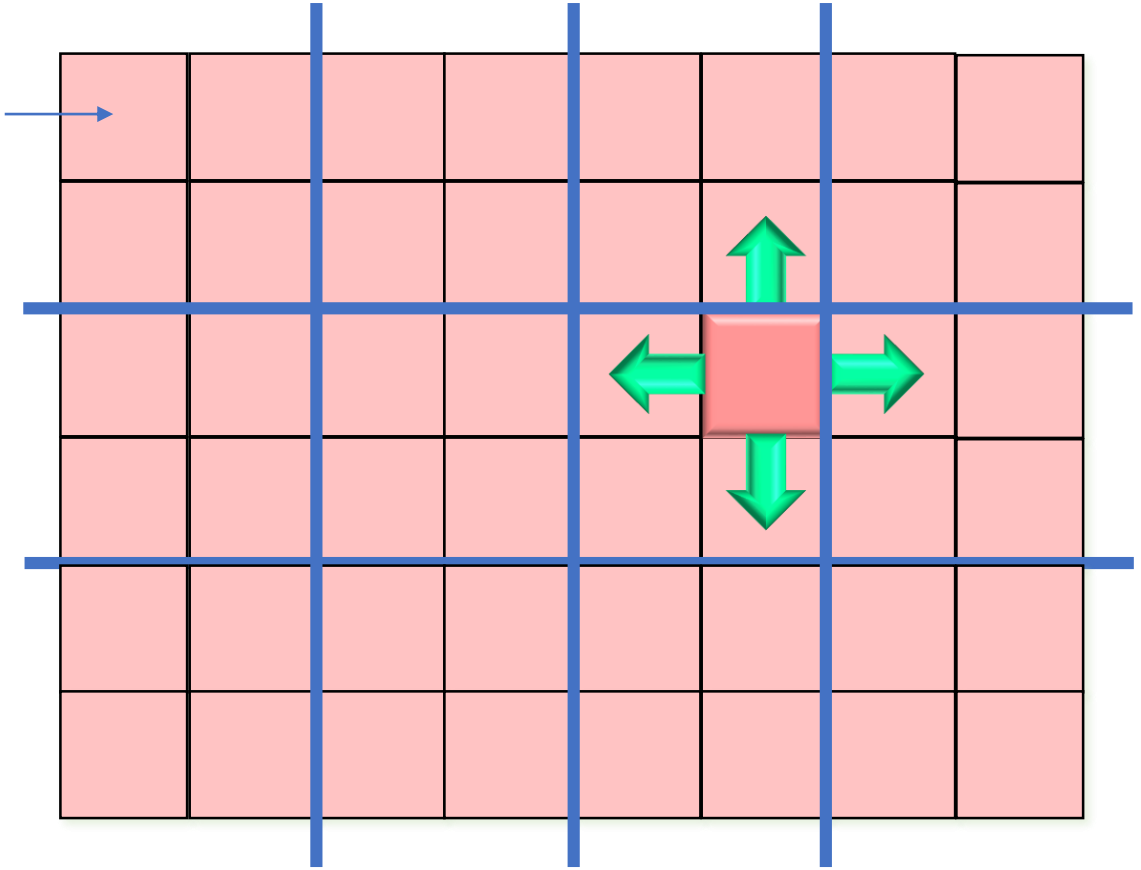
Stencils



Five-point stencil: Each grid point new value is the average of itself and its four neighbors’

2D Domain decomposition

Grid point



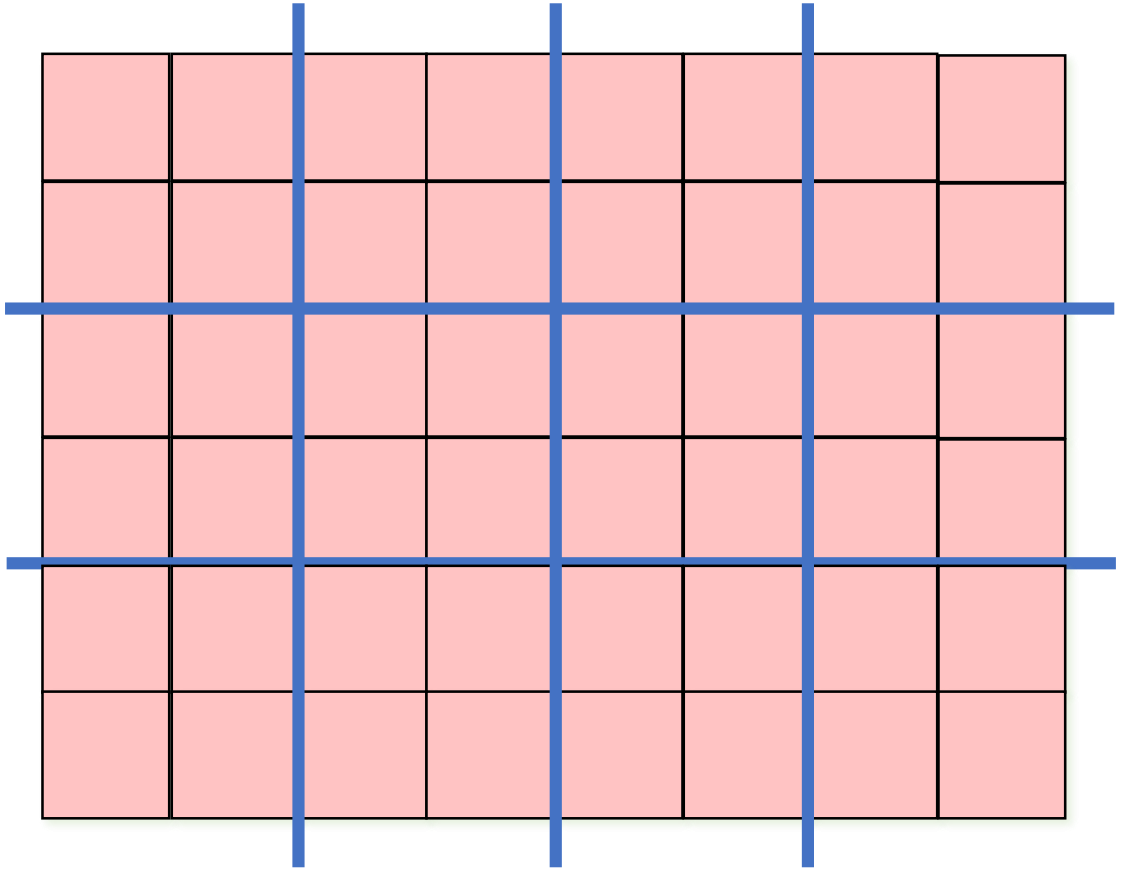
Communications?

4 Isends()
4 Irecvs()

Design considerations for multiple variables?

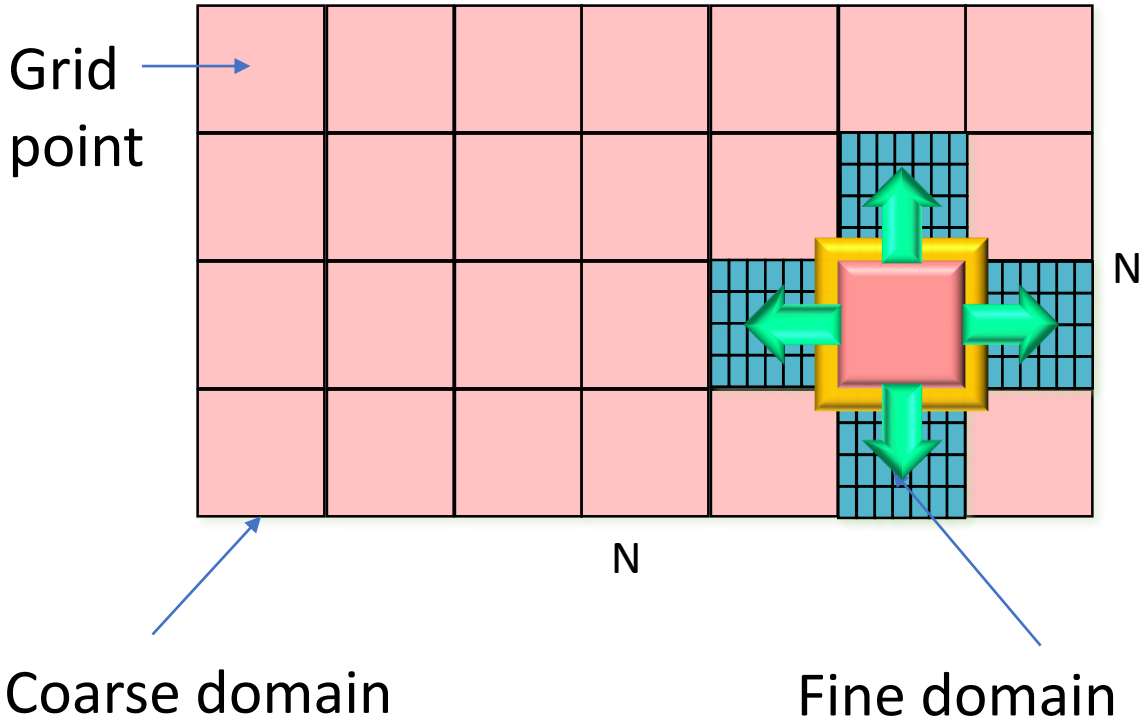
Pack
Unpack

2D Domain decomposition



Does the order of processes matter in 2D?

Domain refinement



Halo exchange

- Each cell has some ghost regions
- Communication with neighbors

#Computations and communication volume of each cell?