Parallelization

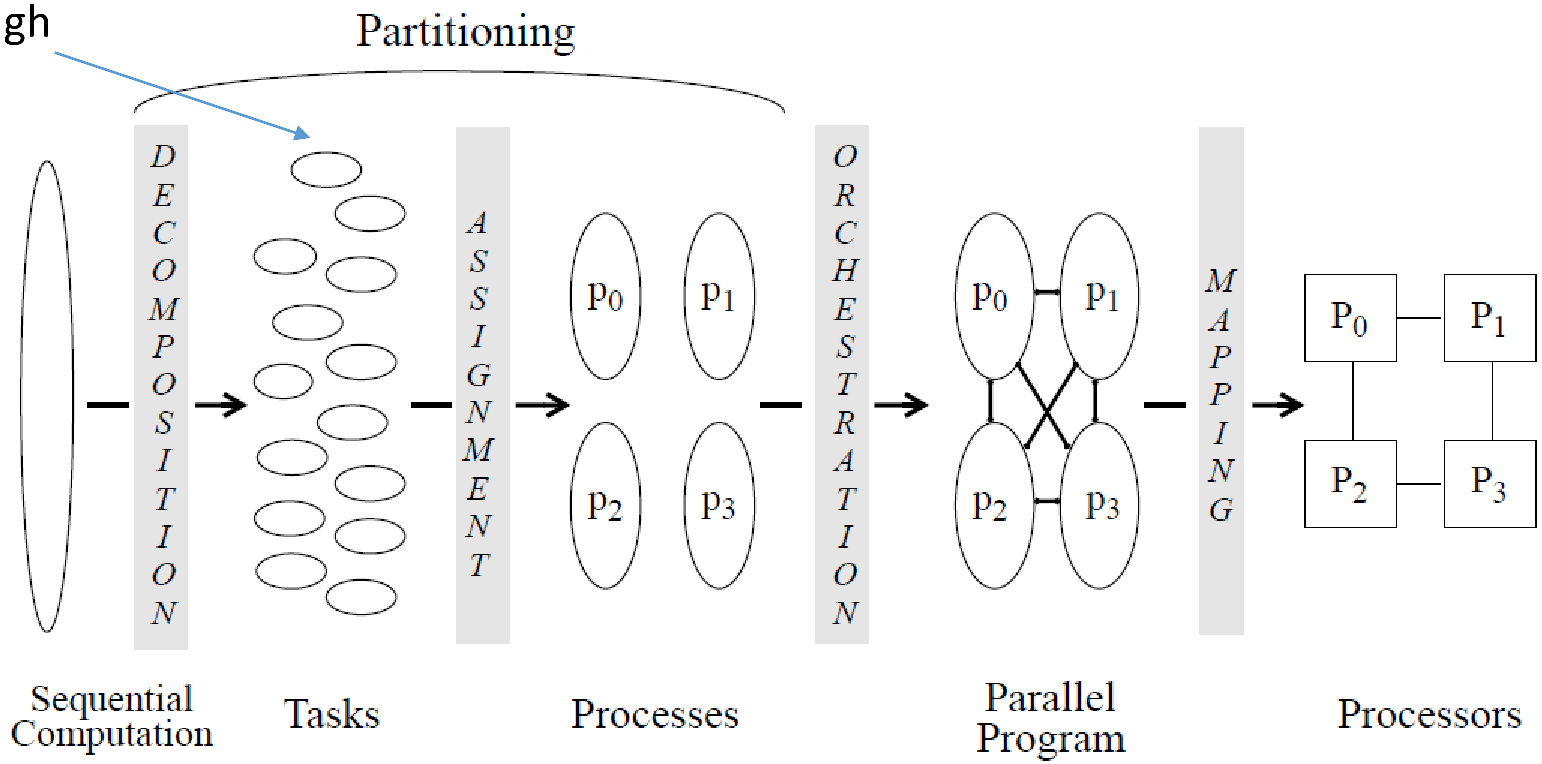
Mar 16, 2021

Parallelization Steps

1. *Decomposition* of computation into tasks
    - Identifying portions of the work that can be performed concurrently
2. *Assignment* of tasks to processes
    - Assigning concurrent pieces of work onto multiple processes running in parallel
3. *Orchestration* of data access, communication and synchronization among processes
    - Distributing the data associated with the program
    - Managing access to data shared by multiple processes
    - Synchronizing at various stages of the parallel program execution
4. *Mapping* of processes to processors
    - Placement of processes in the physical processor topology

Illustration of Parallelization Steps

Expose enough concurrency



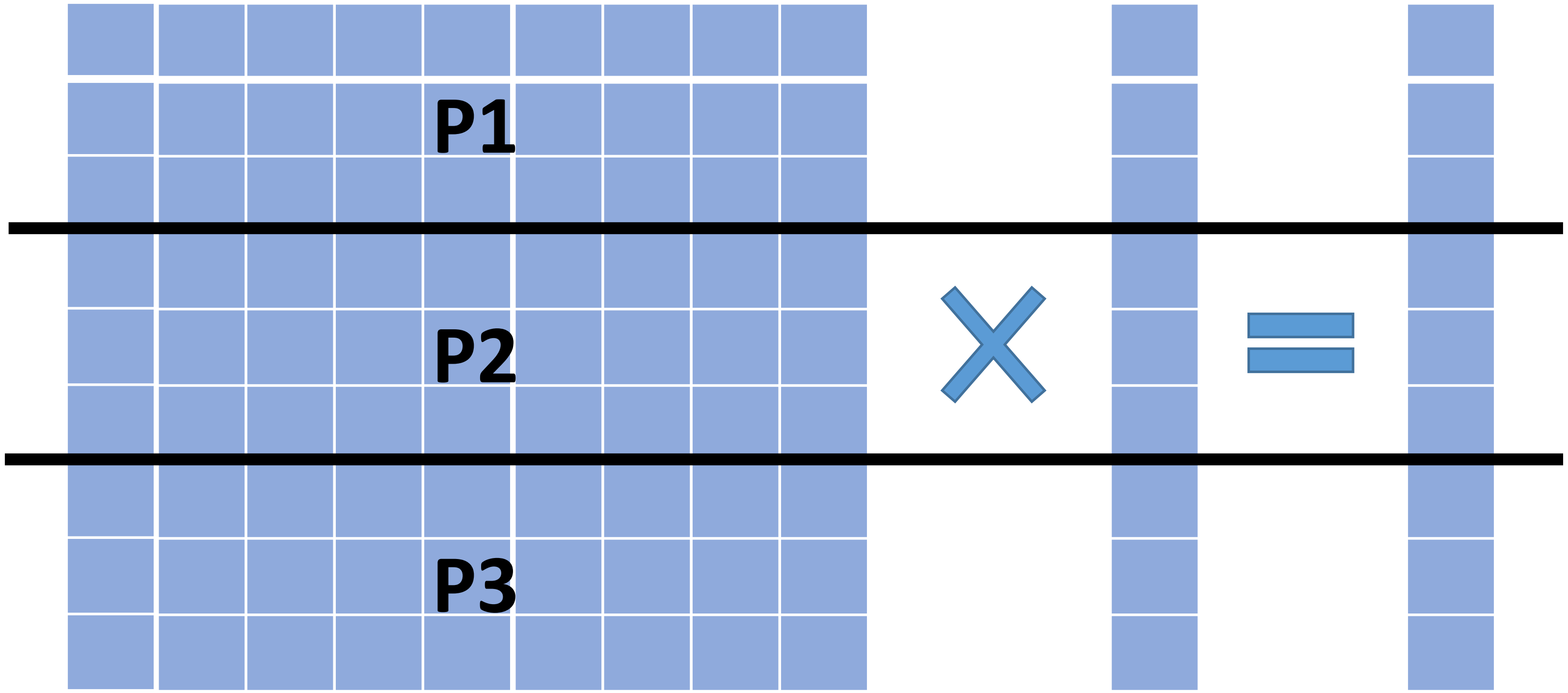
Q: What if number of processes != available processors?

Source: Culler et al.

Performance Goals

- Expose concurrency
- Reduce inter-process communications
- Load-balance
- Reduce synchronization
- Reduce idling
- Reduce management overhead
- Preserve data locality
- Exploit network topology

Matrix Vector Multiplication – Decomposition



P = 9 ?

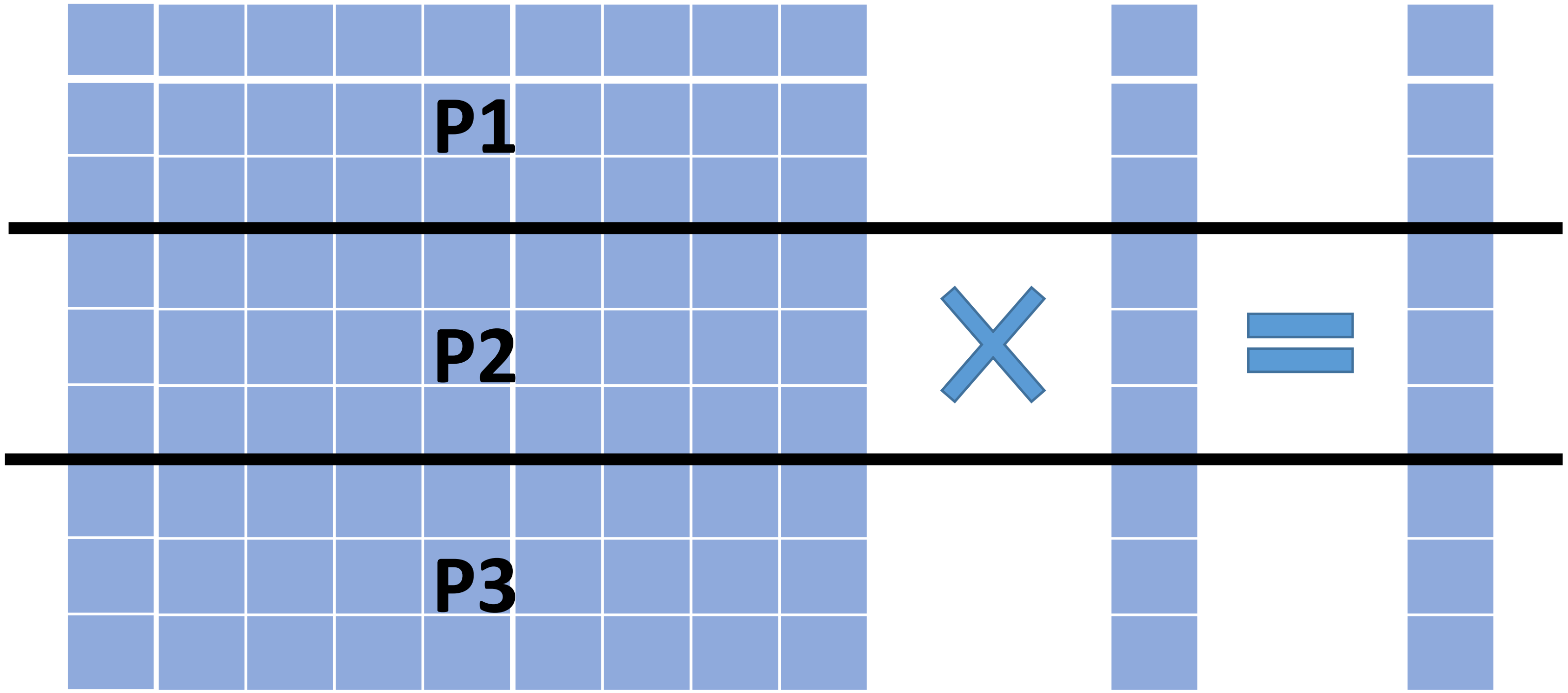
P = 3 ?

Decomposition

Identifying portions of the work that can be performed concurrently

Assignment

Matrix Vector Multiplication – Orchestration



P = 3

Decomposition

Assignment

Orchestration

- Allgather/Bcast
- Gather

- What is the initial communication step?
- Ways to distribute vector (if its present in 1 process)?
- What are the differences between distribution and parallel reads?

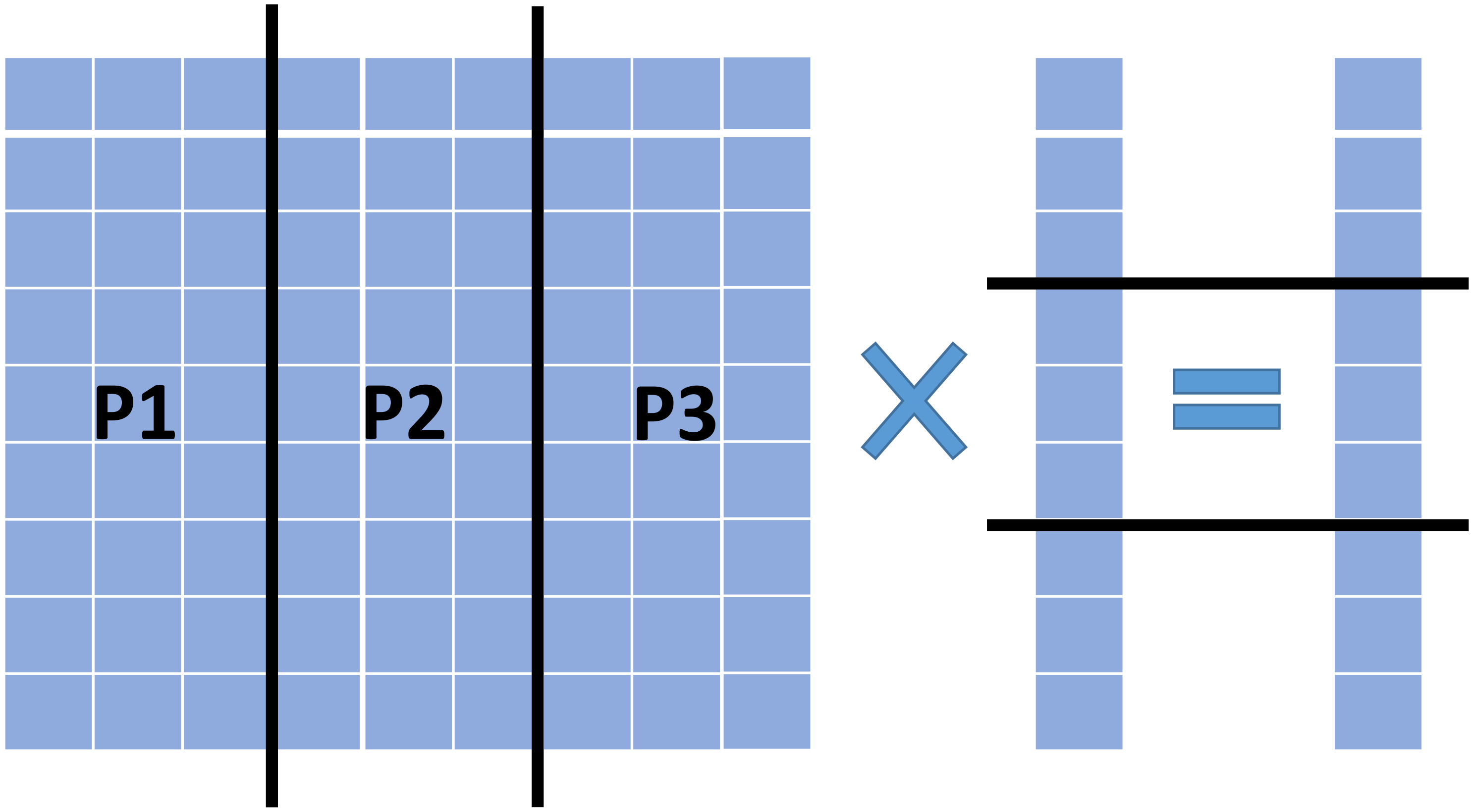
Distribute using Bcast vs. Allgather

```
MPI_Init( &argc, &argv );
MPI_Comm_rank( MPI_COMM_WORLD, &myrank );
MPI_Comm_size( MPI_COMM_WORLD, &commsize );

time = MPI_Wtime();
MPI_Bcast (buf, N, MPI_FLOAT, 0, MPI_COMM_WORLD);
etime = MPI_Wtime() - time;
MPI_Reduce (&etime, &maxtime, 1, MPI_DOUBLE, MPI_MAX, 0, MPI_COMM_WORLD);
if (!myrank) printf ("Time to bcast: %11.3lf\n", maxtime);

int count = N/commsize;
time = MPI_Wtime();
MPI_Allgather (buf, count, MPI_FLOAT, recvbuf, count, MPI_FLOAT, MPI_COMM_WORLD);
etime = MPI_Wtime() - time;
MPI_Reduce (&etime, &maxtime, 1, MPI_DOUBLE, MPI_MAX, 0, MPI_COMM_WORLD);
if (!myrank) printf ("Time to allgather: %7.3lf\n", maxtime);
```

Bcast vs. Allgather

```
class for i in `seq 1 3` ; do mpirun -np 10 -hosts csews2,csews5,csews20 ./bcast-allgather 10000; echo ; done
Time to bcast:       0.014
Time to allgather:   0.021

Time to bcast:       0.018
Time to allgather:   0.009

Time to bcast:       0.012
Time to allgather:   0.007

class for i in `seq 1 3` ; do mpirun -np 10 -hosts csews2,csews5,csews20 ./bcast-allgather 100000; echo ; done
Time to bcast:       0.034
Time to allgather:   0.011

Time to bcast:       0.027
Time to allgather:   0.023

Time to bcast:       0.026
Time to allgather:   0.011

class for i in `seq 1 3` ; do mpirun -np 10 -hosts csews2,csews5,csews20 ./bcast-allgather 1000000; echo ; done
Time to bcast:       0.187
Time to allgather:   0.347

Time to bcast:       0.176
Time to allgather:   0.111

Time to bcast:       0.155
Time to allgather:   0.112

class for i in `seq 1 3` ; do mpirun -np 10 -hosts csews2,csews5,csews20 ./bcast-allgather 10000000; echo ; done
Time to bcast:       1.421
Time to allgather:   1.121

Time to bcast:       1.618
Time to allgather:   1.282

Time to bcast:       1.674
Time to allgather:   1.583

class for i in `seq 1 3` ; do mpirun -np 10 -hosts csews2,csews5,csews20 ./bcast-allgather 100000000; echo ; done
Time to bcast:      18.061
Time to allgather:  15.616

Time to bcast:      23.447
Time to allgather:  17.005

Time to bcast:      16.875
Time to allgather:  11.085
```

Matrix Vector Multiplication – Decomposition

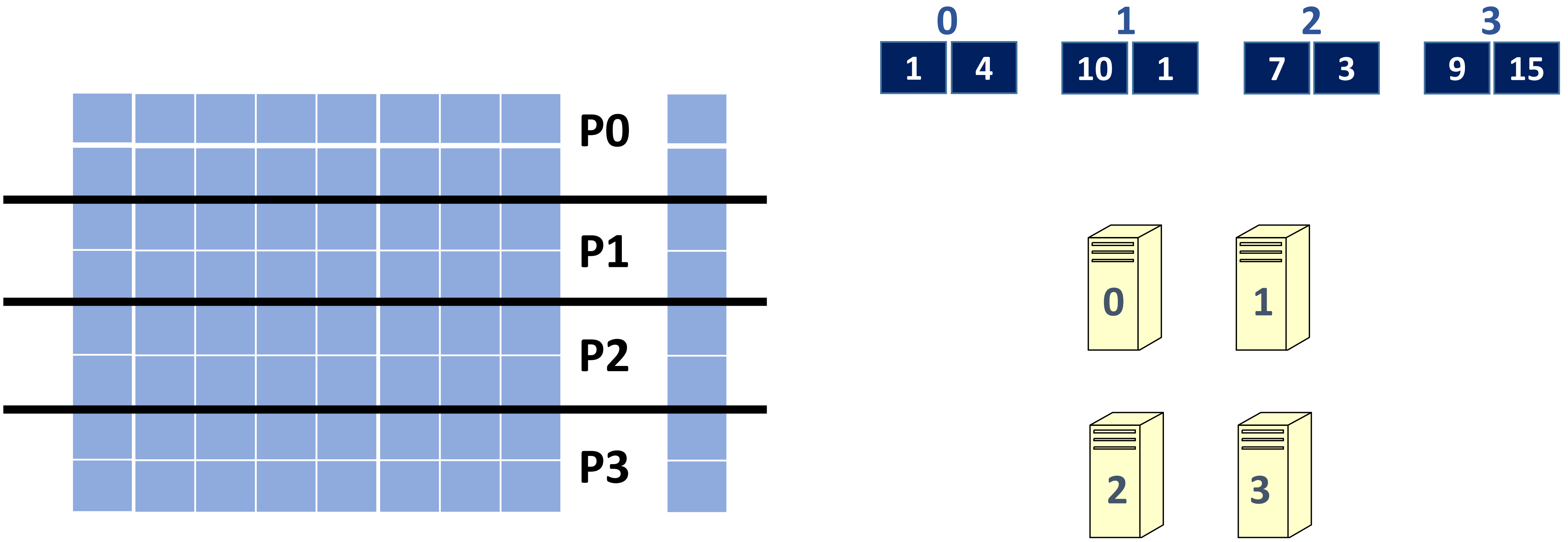


P = 3

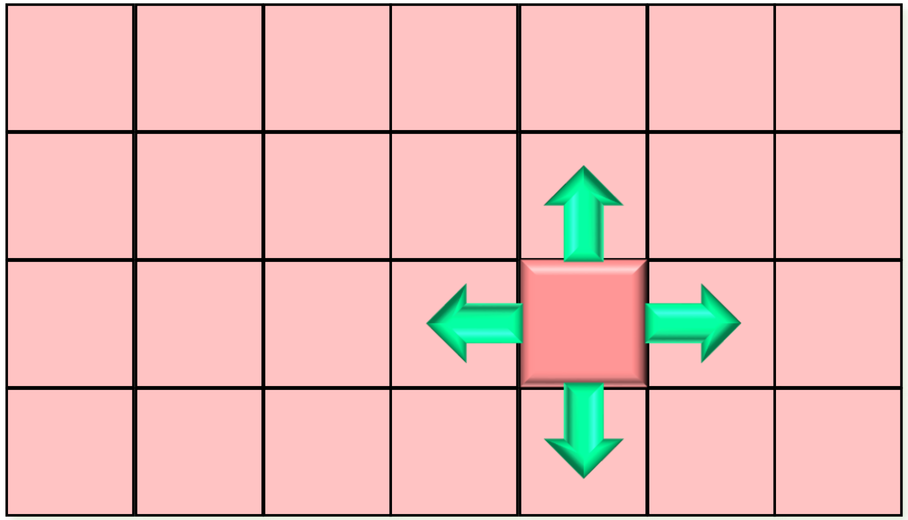
Decomposition
Assignment
Orchestration

What is the advantage of column-wise partitioning ?

Matrix Vector Multiplication – Mapping



Stencils



Five-point stencil

Virtual Topology

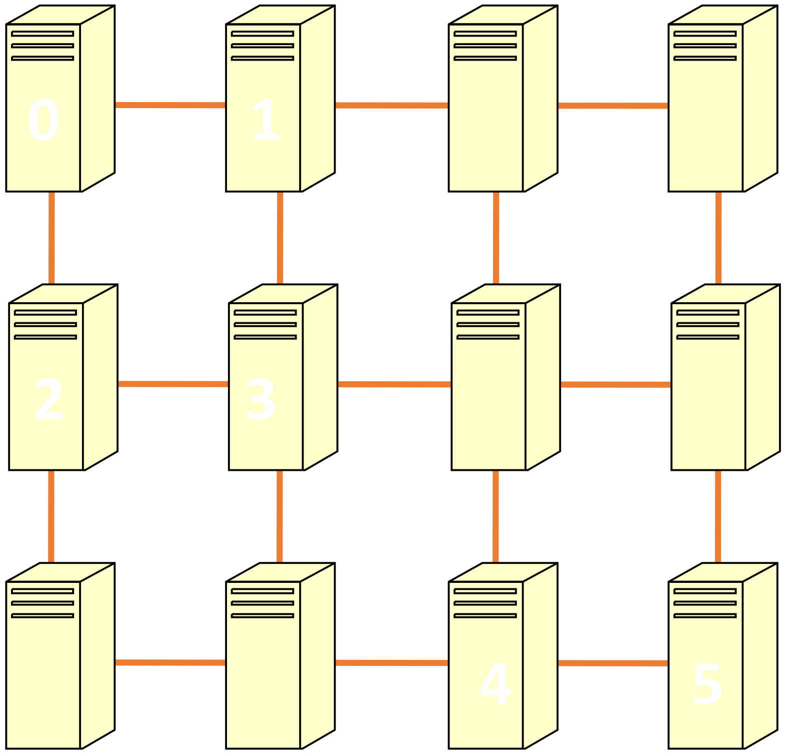
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

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

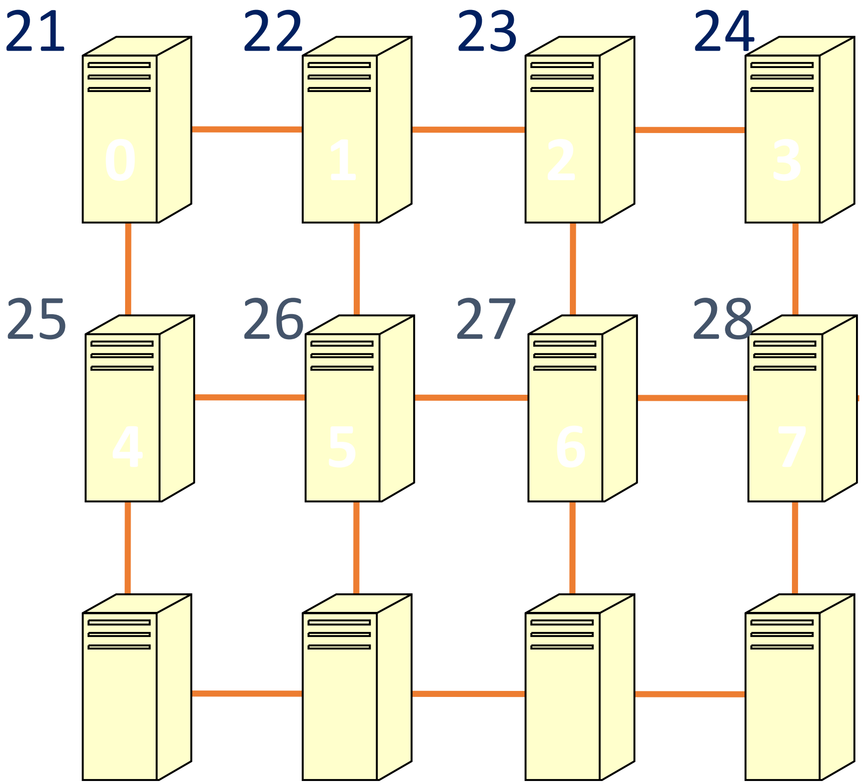
Note: Virtual topology is set up before execution

Physical Topology



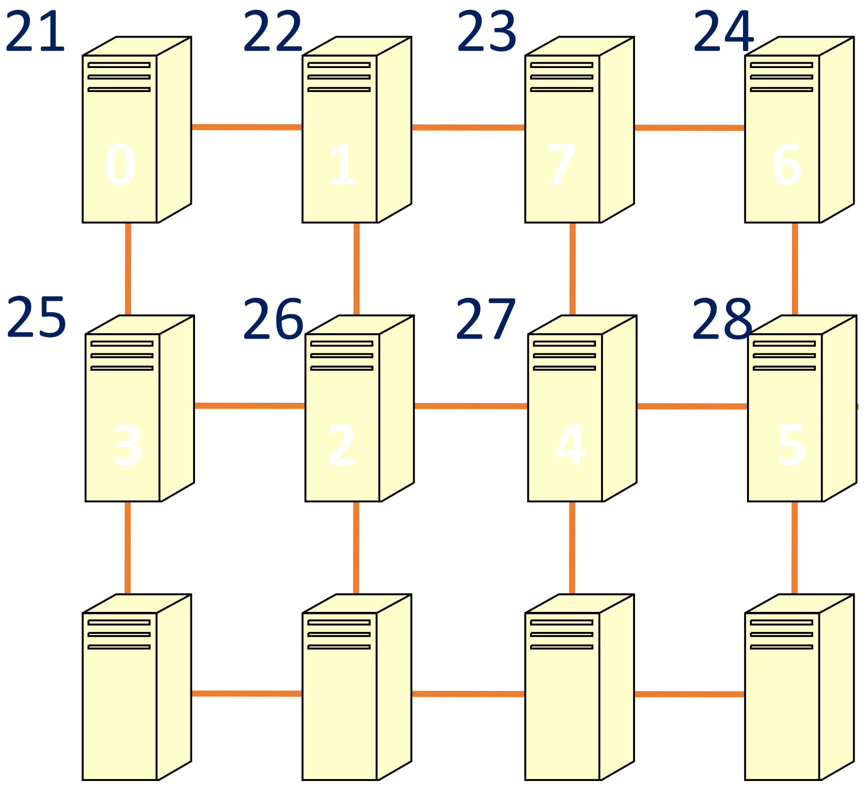
- Connections between allocated nodes
- Mapping: Placement of ranks onto cores

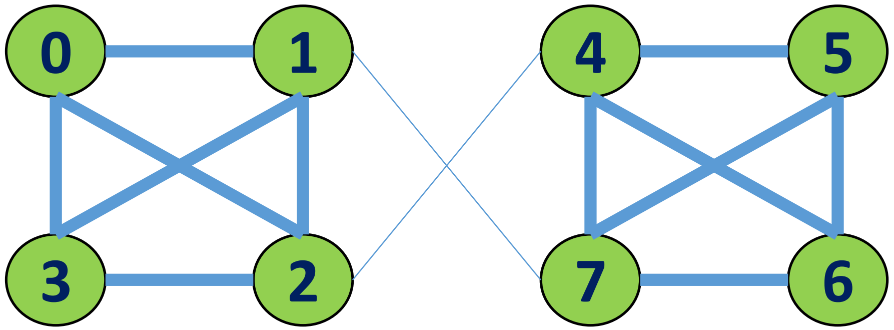
Physical Topology



- Connections between allocated cores
- Mapping: Placement of ranks onto cores
- Default placement of ranks based on node IDs

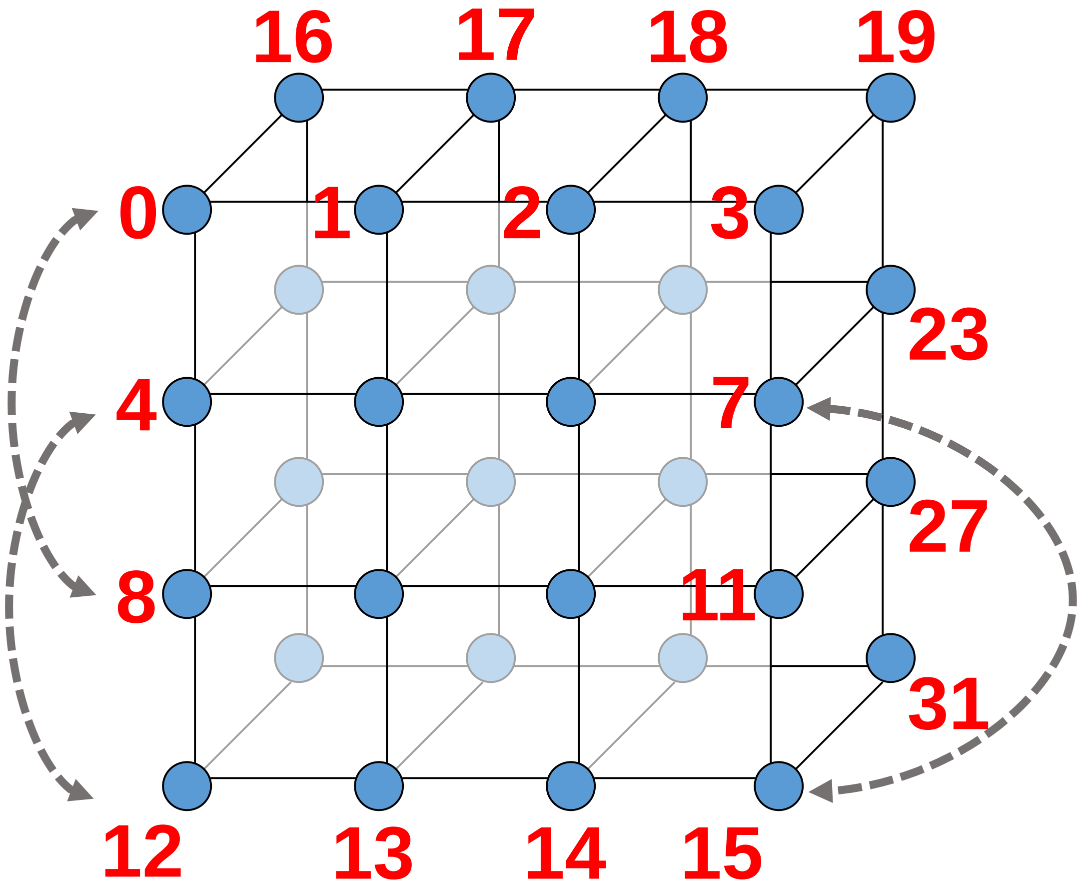
Rank placement





- May place ranks anywhere in the allocated nodes based on the communication pattern
- Topology-aware mapping: Mapping that minimizes all communication times taking into account the physical topology

Process-to-processor Mapping

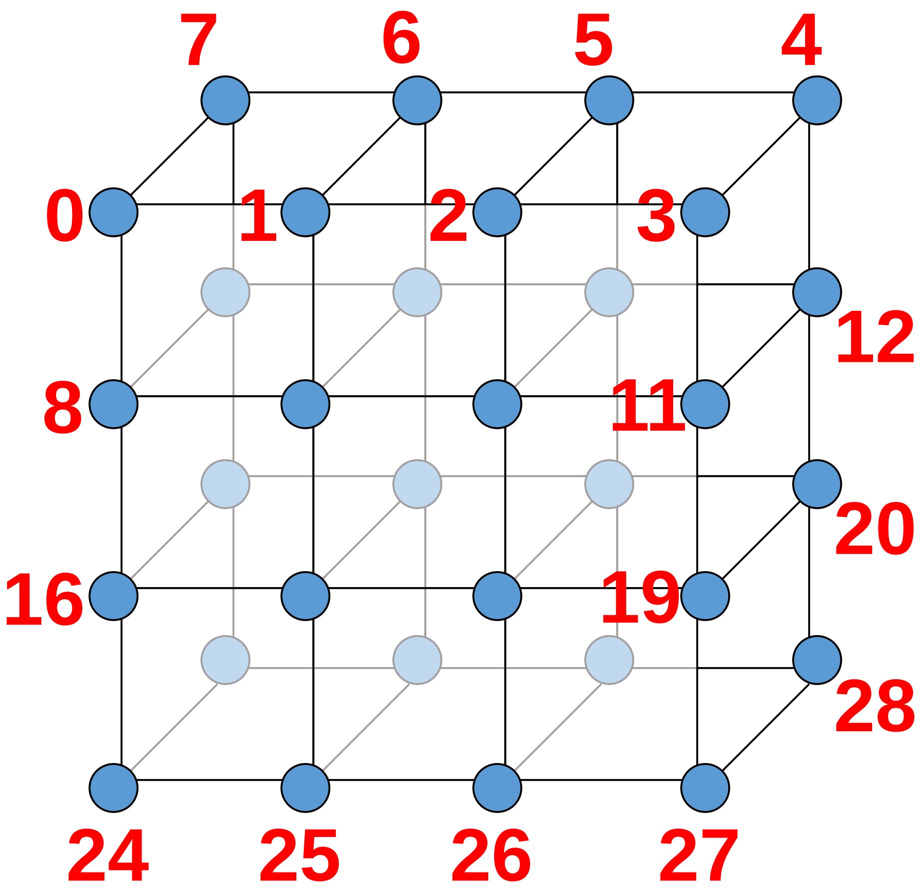


4 x 4 x 2 3D torus

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

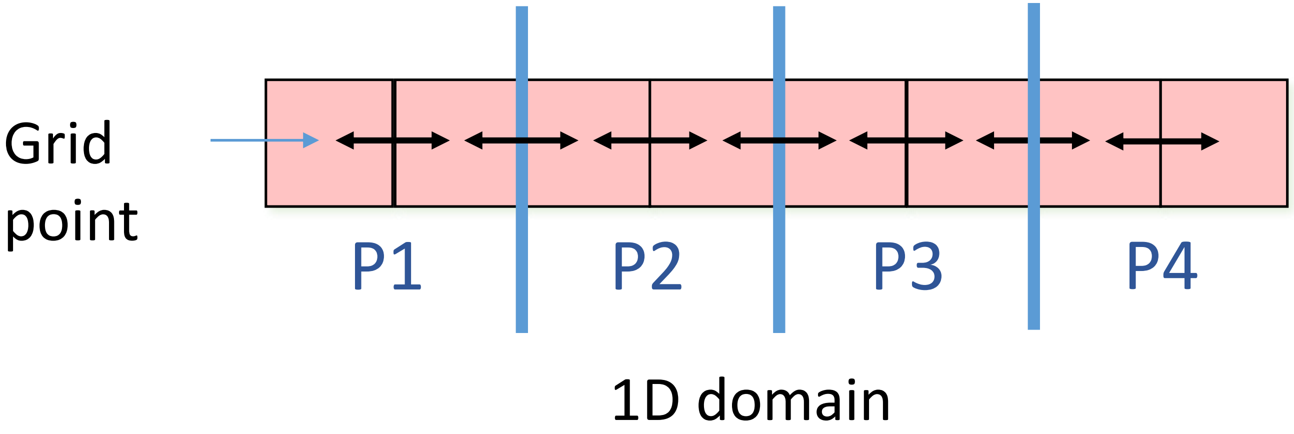


4 x 4 x 2 3D torus

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



N grid points
P processes
N/P points per process

#Communications?
2

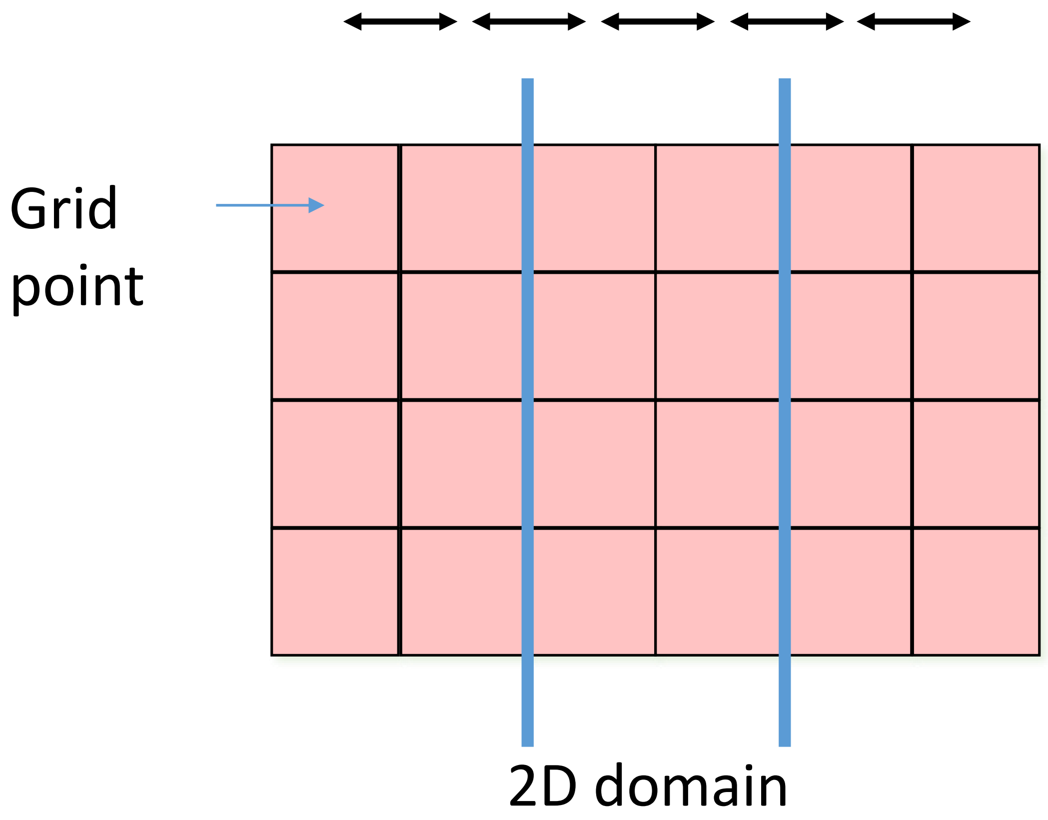
#Computations?
N/P

Communications?

2 sends()
2 recvs()

Communication to computation ratio=?

1D Domain decomposition



N grid points
P processes
N/P points per process

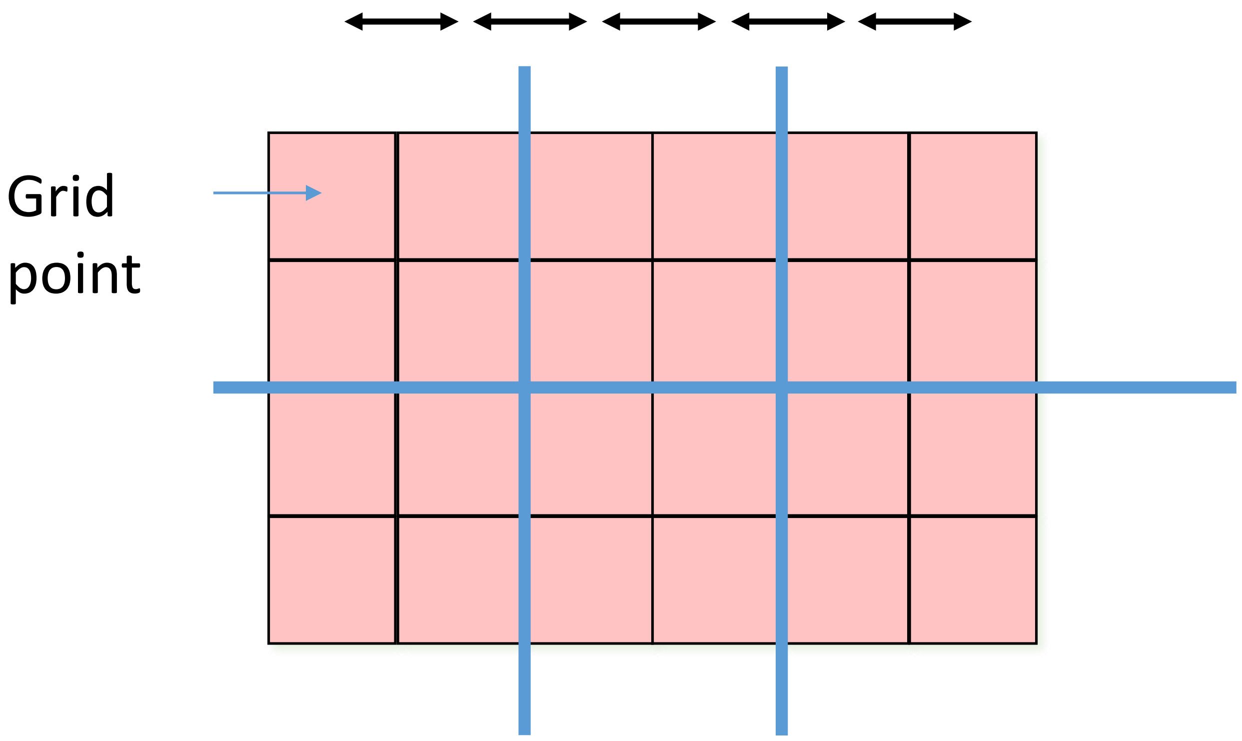
#Communications?
$2\sqrt{N}$ (assuming square grid)

#Computations?
N/P (assuming square grid)

Q: One drawback?

Communication to computation ratio=?

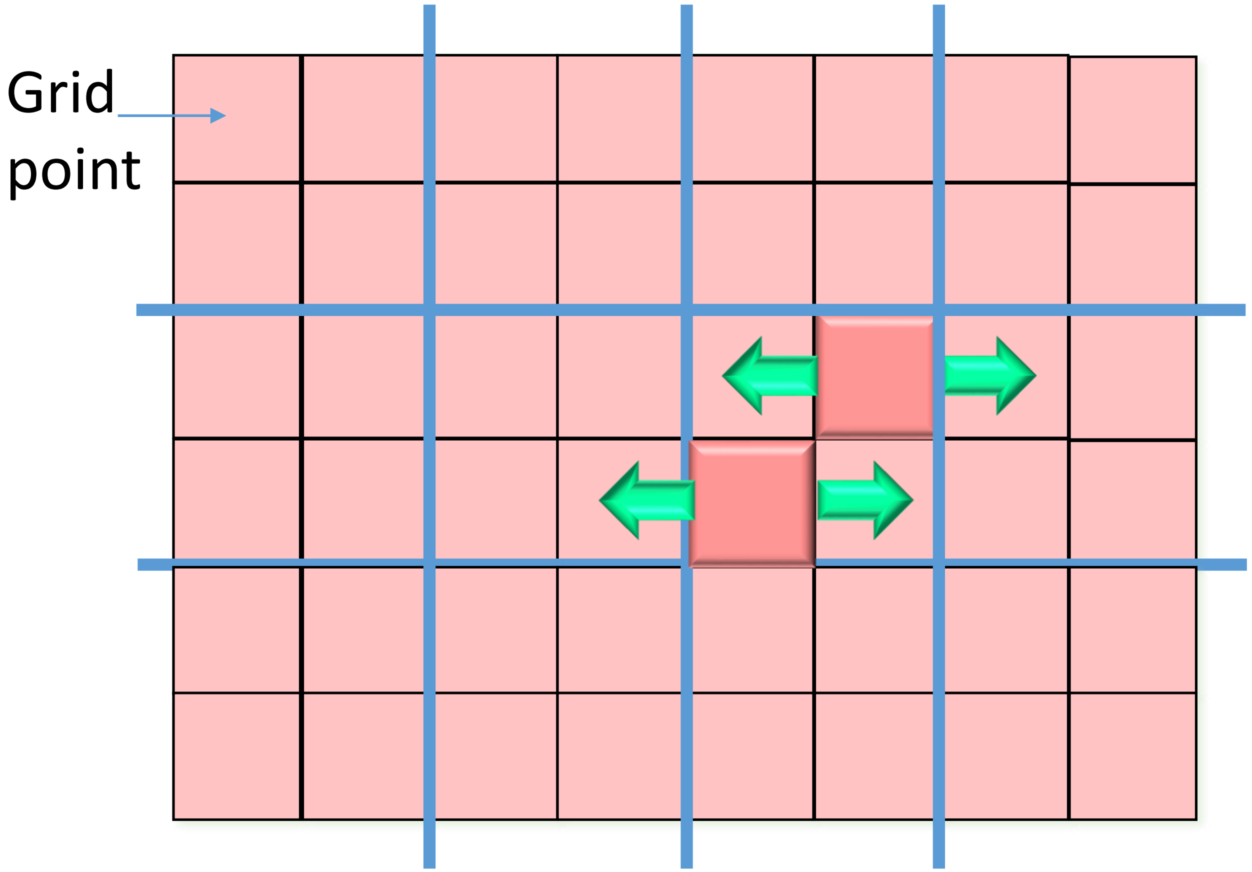
2D Domain decomposition



N grid points
P processes
N/P points per process

+ Several parallel communications
+ Lower communication volume/process

2D Domain decomposition



#Communications?

2 Sends()
2 Recvs()

#Communications?

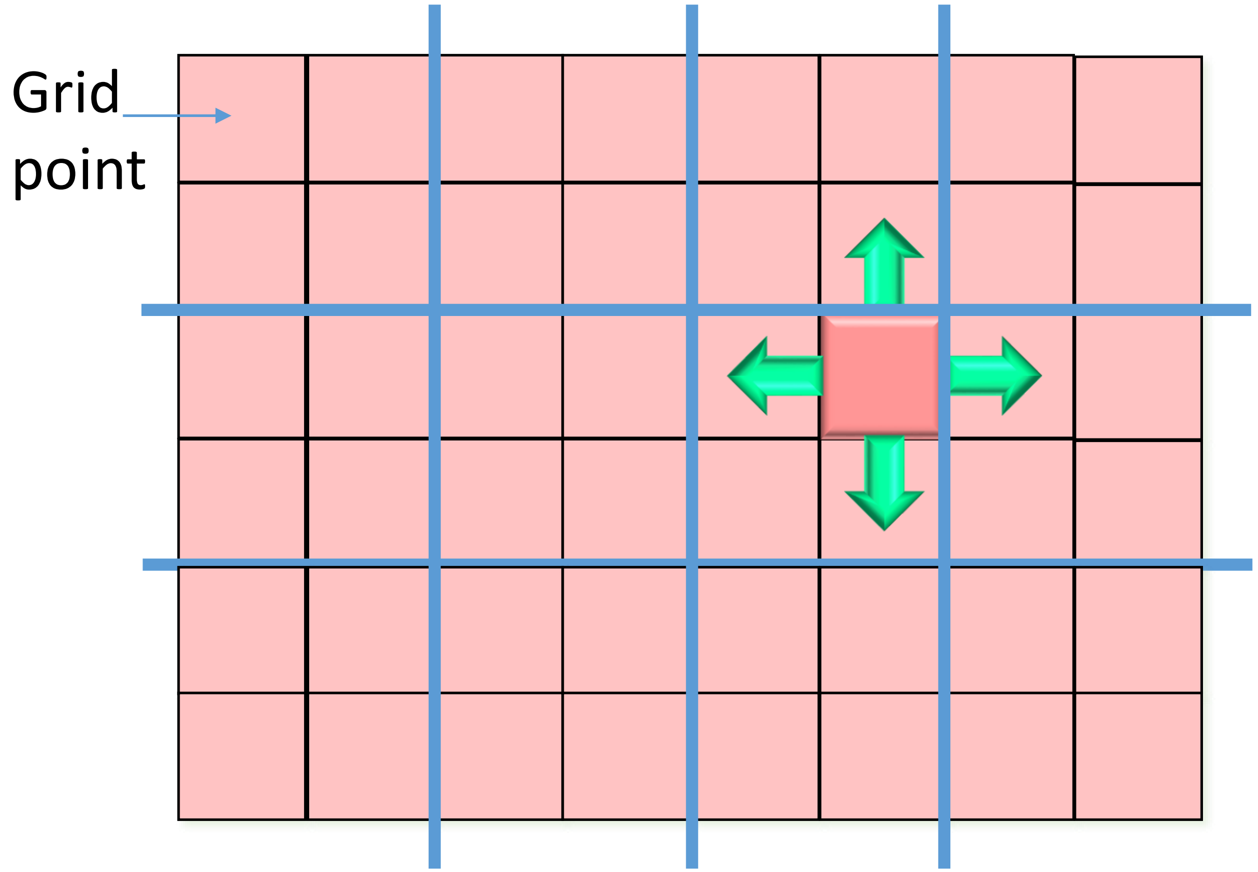
2√N/√P (assuming square grid)

#Computations?

N/P (assuming square grid)

Communication to computation ratio=?

2D Domain decomposition



#Communications?

4 Sends()
4 Recvs()

#Communications?

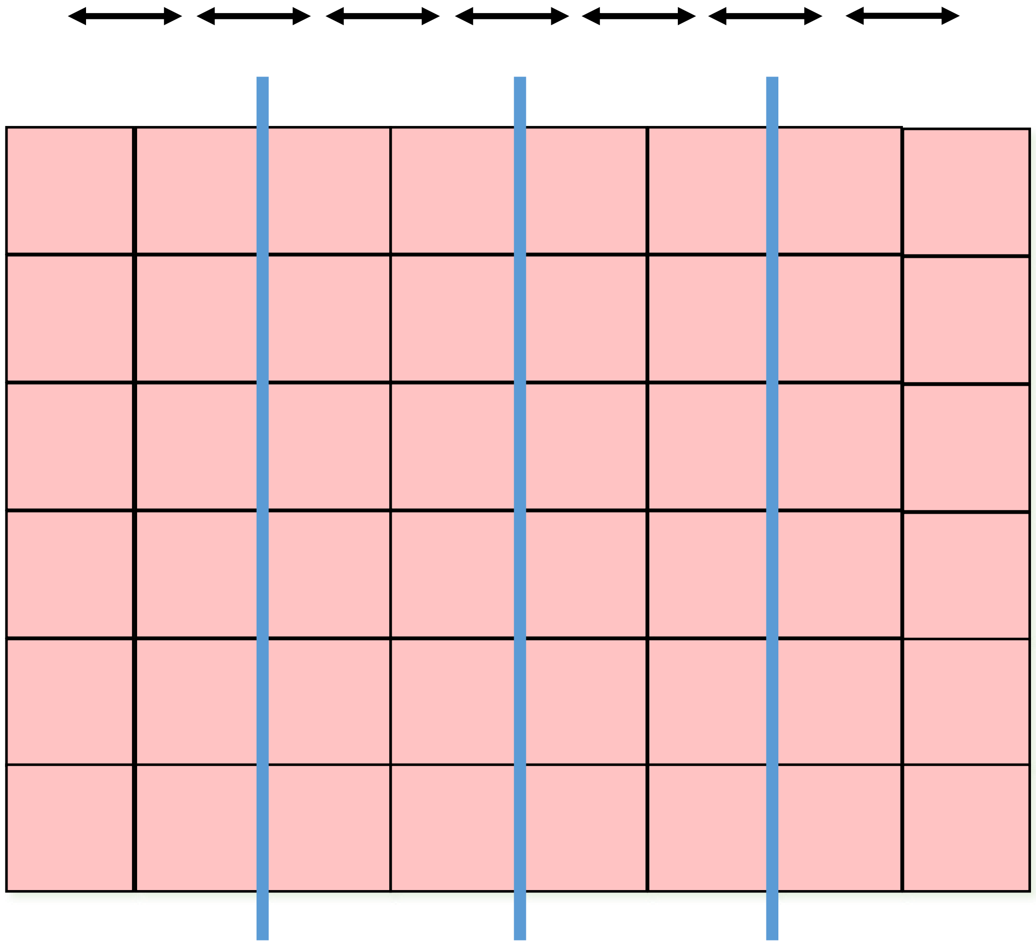
4√N/√P (assuming square grid)

#Computations?

N/P (assuming square grid)

Communication to computation ratio=?

Communication to Computation Ratio (1D)

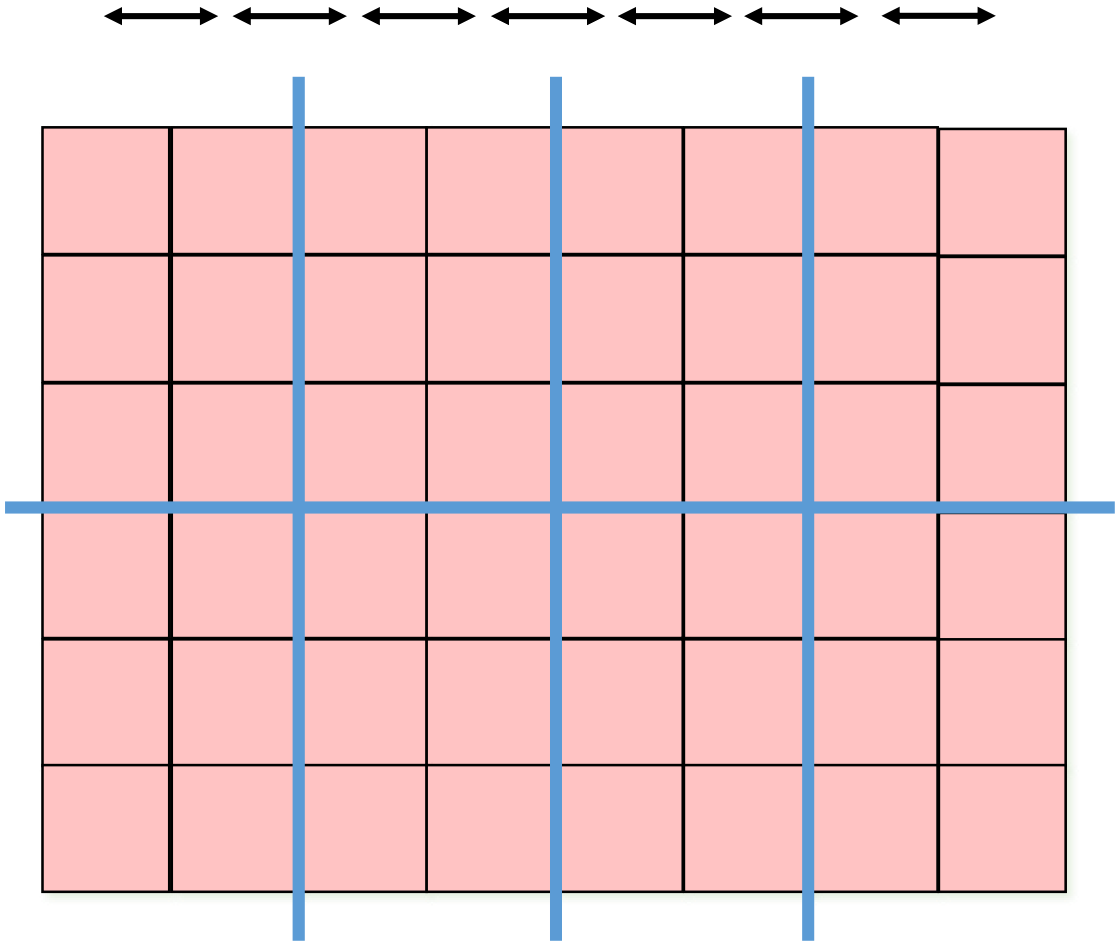


#Communications = 12

#Computations = 12

Communication to computation ratio = 1

Communication to Computation Ratio (2D)



#Communications = 6

#Computations = 6

Communication to computation ratio = 1