Parallel Algorithms

Alan Sussman, Department of Computer Science

UNIVERSITY OF MARYLAND

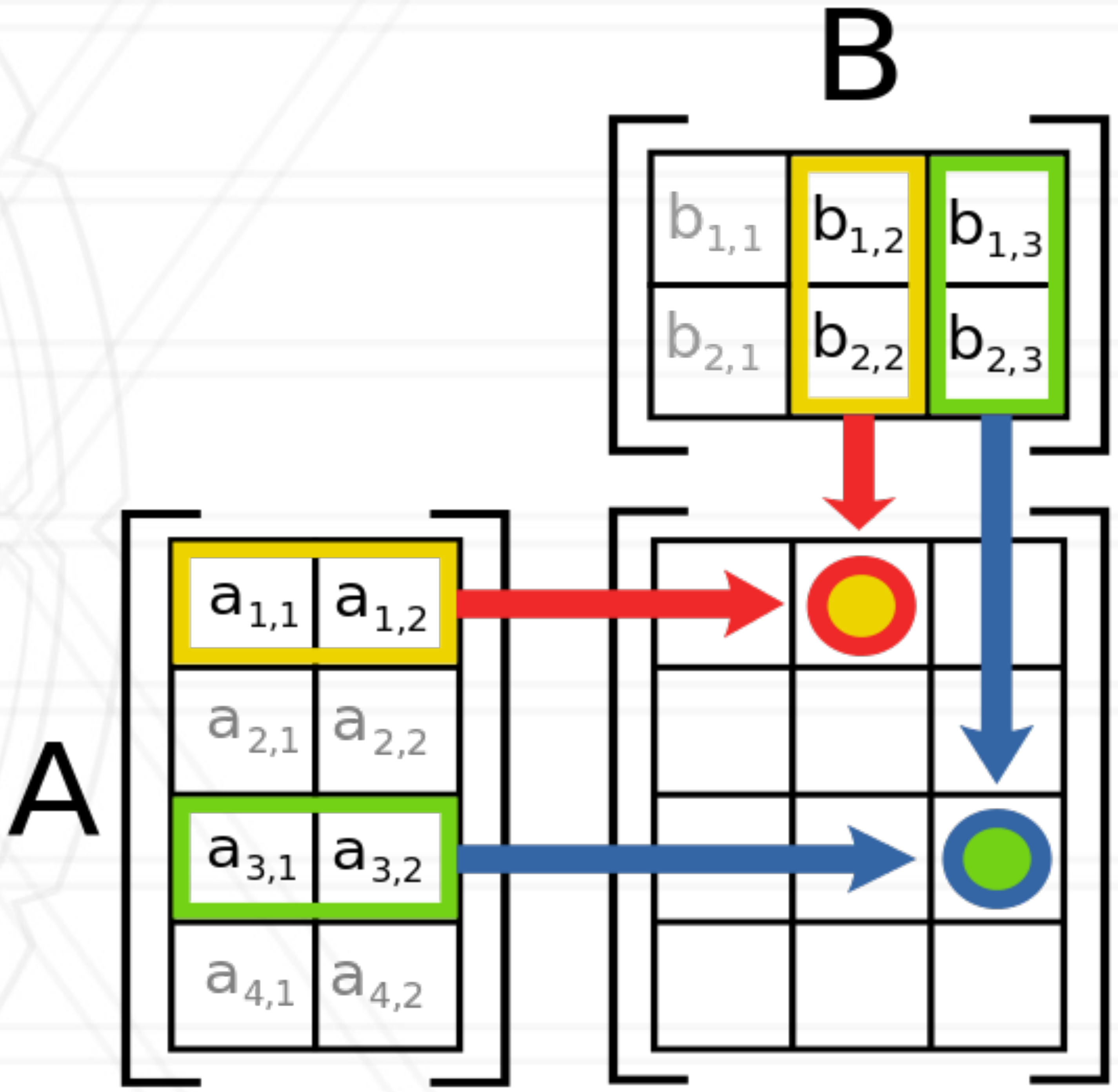
Announcements

- Assignment 1 is due on March 7 11:59 pm
  - Questions?

Matrix multiplication

```
for (i=0; i<M; i++)
  for (j=0; j<N; j++)
    for (k=0; k<L; k++)
      C[i][j] += A[i][k]*B[k][j];
```

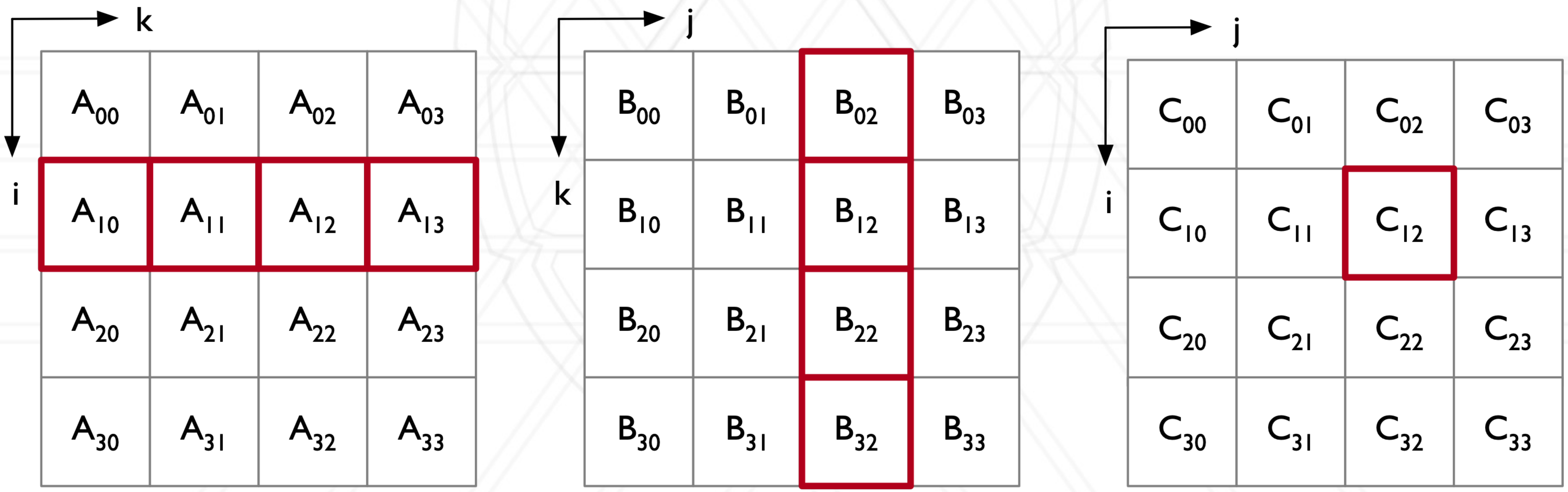
Any performance issues for large arrays?



https://en.wikipedia.org/wiki/Matrix_multiplication

Blocking to improve cache performance

- Create smaller blocks that fit in cache: leads to cache reuse
- $C_{12} = A_{10} * B_{02} + A_{11} * B_{12} + A_{12} * B_{22} + A_{13} * B_{32}$



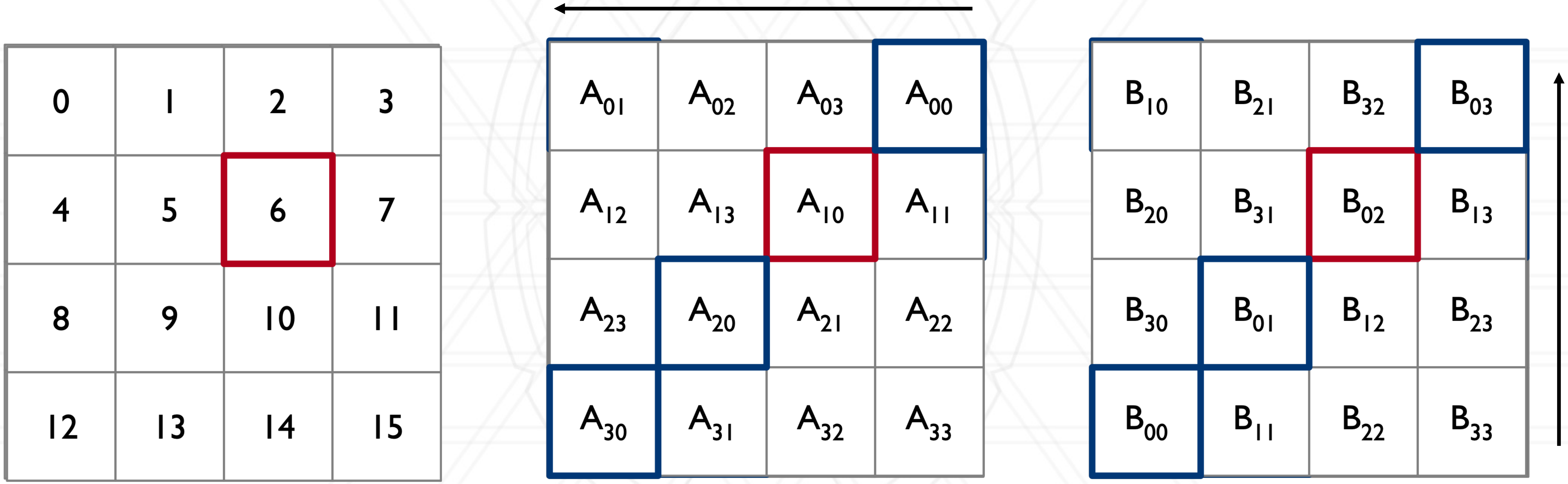
Parallel matrix multiply

- Store A and B in a distributed manner
- Communication between processes to get the right sub-matrices to each process
- Each process computes a portion of C

Cannon's 2D matrix multiply

- Views processors/processes as arranged in a 2D grid
- Storage requirements are constant and independent of number of processes
  - After initial distribution of matrices, only fixed number of intermediate results need to be stored, so each matrix is stored exactly once (no replication)
- Leads to Agarwal's SUMMA (Scalable Universal Matrix Multiplication Algorithm) employed in widely used linear algebra libraries for distributed memory
  - e.g., ScaLAPACK, PLAPack, etc.

Cannon's 2D matrix multiply



2D process grid

Initial skew

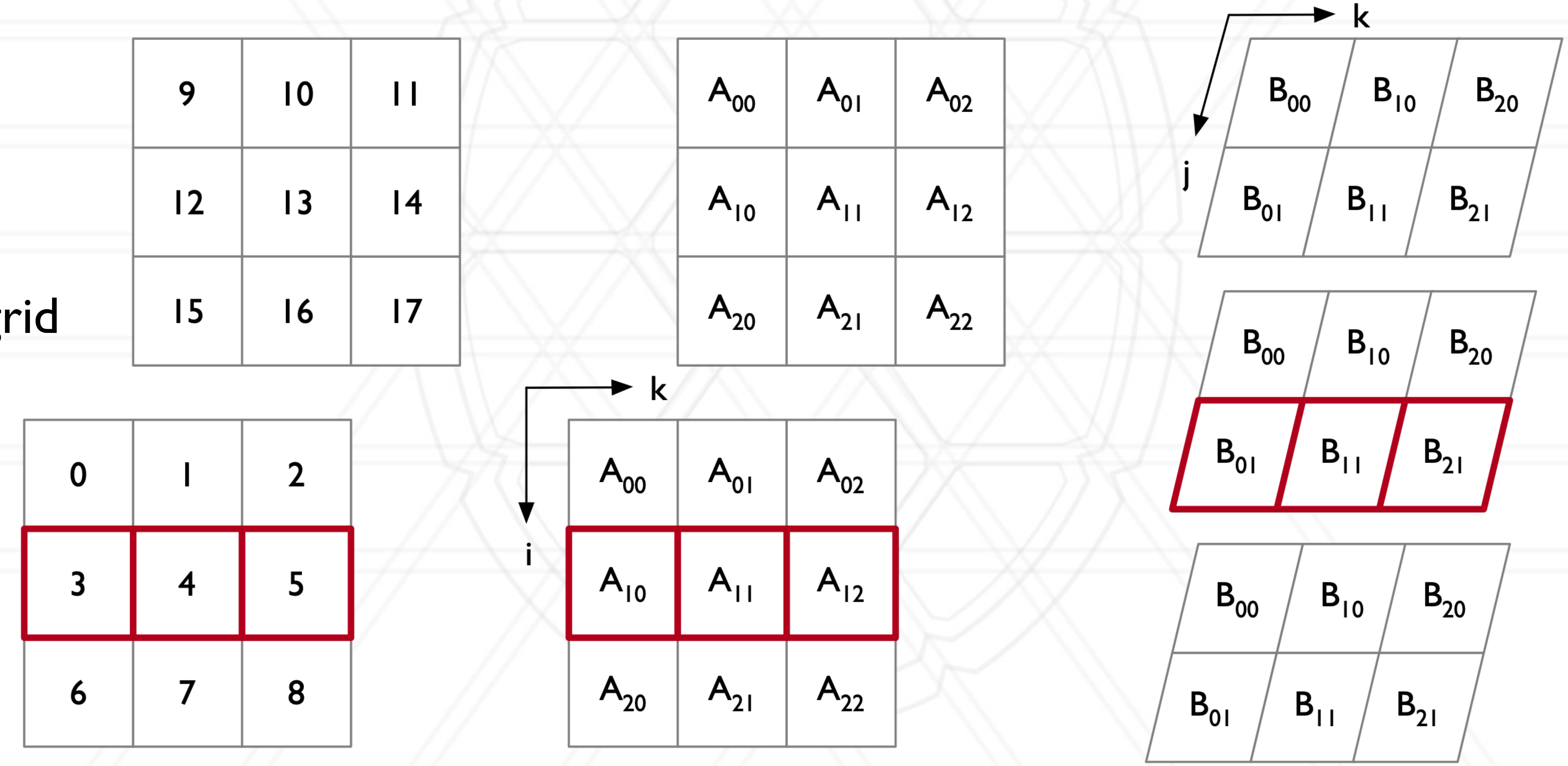
Shift-by-1

by i

Agarwal's 3D matrix multiply - SUMMA

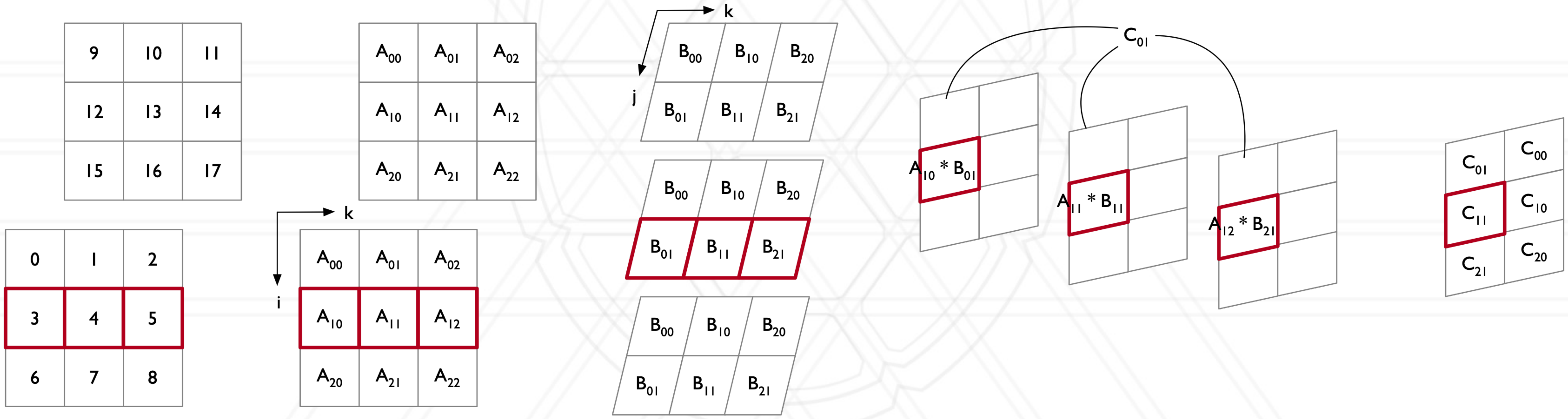
- Copy A to all i-k planes and B to all j-k planes

3D process grid



Agarwal's 3D matrix multiply

- Perform a single matrix multiply to calculate partial C
- Allreduce along i-j planes to calculate final result



Communication algorithms

- Reduction
- All-to-all

Types of reduction

- Scalar reduction: every process contributes one number
  - Perform some commutative and associative operation
- Vector reduction: every process contributes an array of numbers

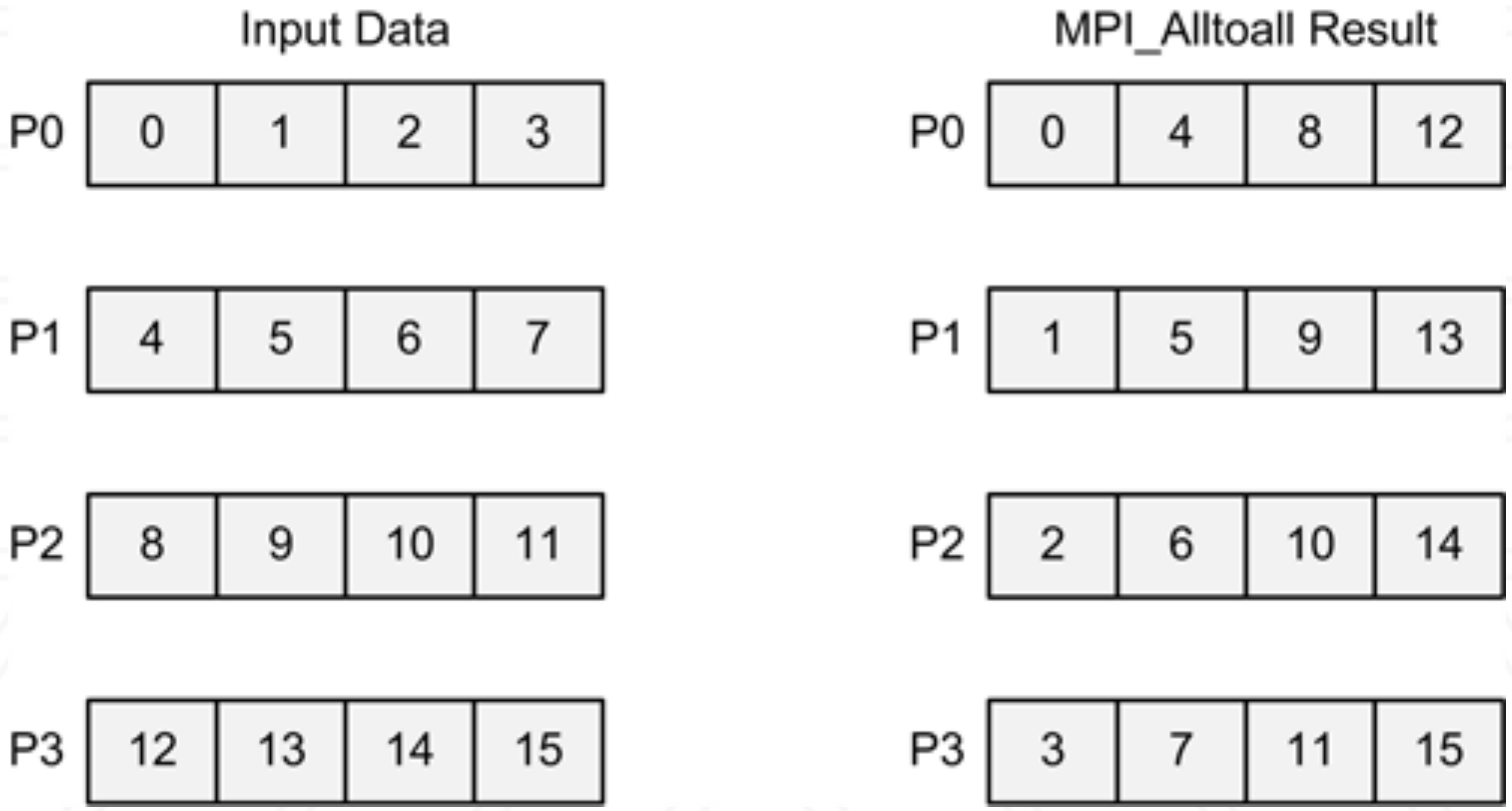
Parallelizing reduction

- Naive algorithm: every process sends to the root
- Spanning tree: organize processes in a k-ary tree
- Start at leaves and send to parents
- Intermediate nodes wait to receive data from all their children
- Number of phases: $\log_k p$

MPI Reduction Algorithms: https://hcl.ucd.ie/system/files/TJS-Hasanov-2016.pdf

All-to-all

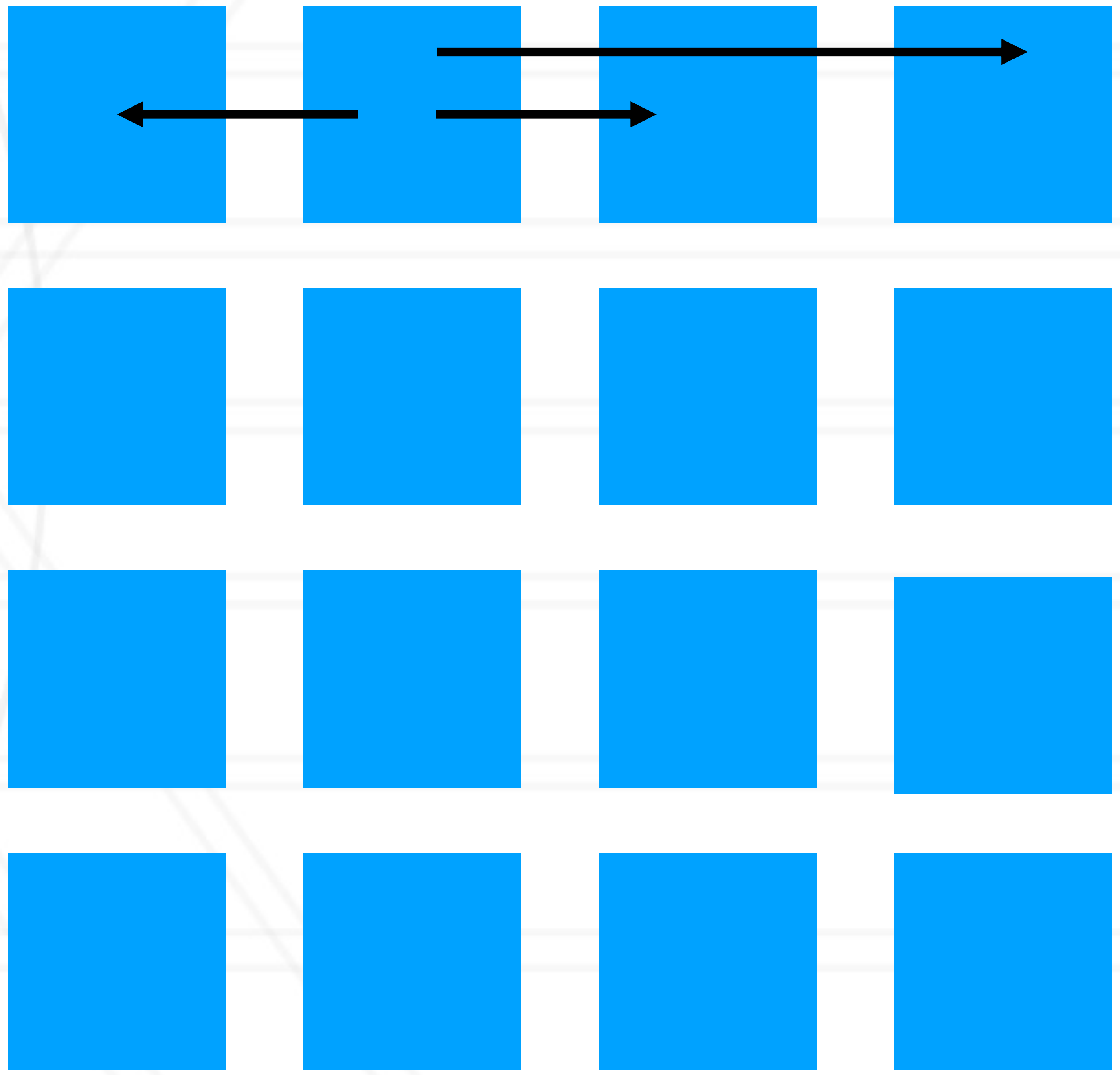
- Each process sends a distinct message to every other process
- Naive algorithm: every process sends the data pair-wise to all other processes



https://www.codeproject.com/Articles/896437/A-Gentle-Introduction-to-the-Message-Passing-Inter

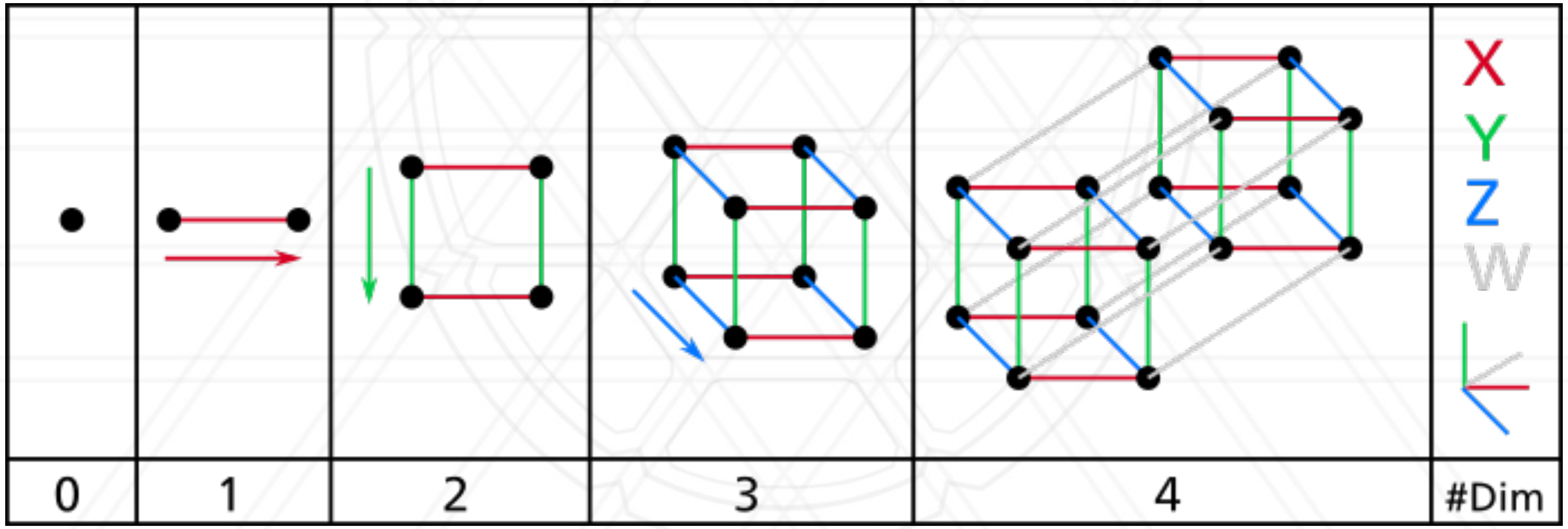
Virtual topology: 2D mesh

- Phase 1: every process sends to its row neighbors
- Phase 2: every process sends to column neighbors



Virtual topology: hypercube

- Hypercube is an n-dimensional analog of a square (n=2) and cube (n=3)
- Special case of k-ary d-dimensional mesh



https://en.wikipedia.org/wiki/Hypercube

UNIVERSITY OF
MARYLAND