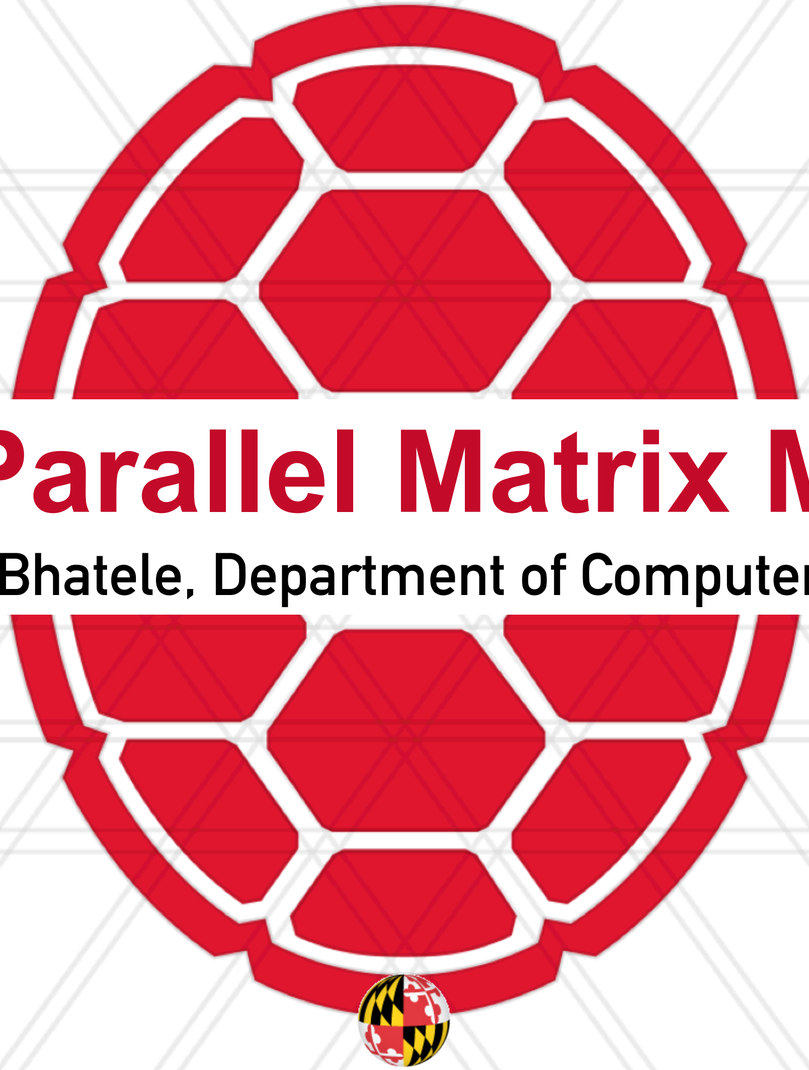
High Performance Computing Systems (CMSC714)



Lecture 20: Parallel Matrix Multiplication

Abhinav Bhatele, Department of Computer Science

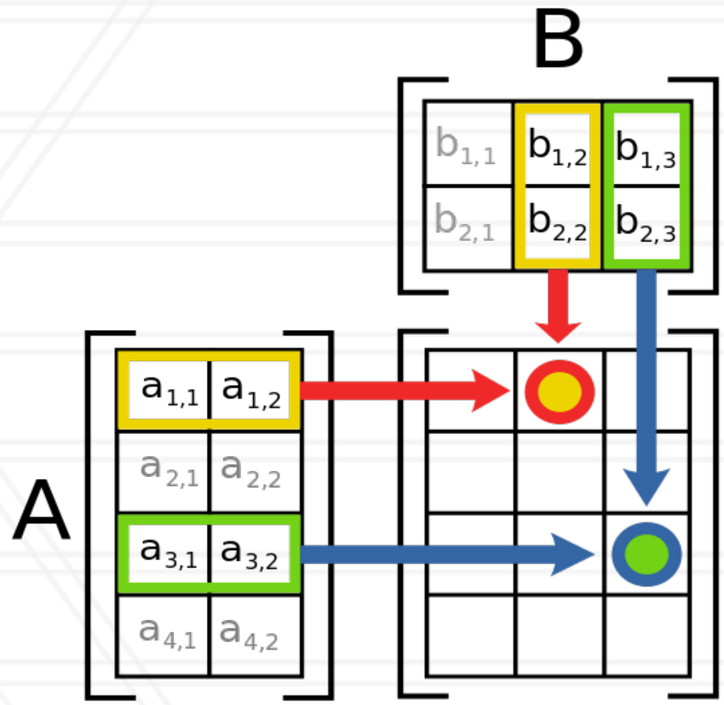
UNIVERSITY OF MARYLAND

Summary of last lecture

- Parallel sorting is used in many HPC applications
- Two categories of parallel sort algorithms: merge-based and splitter-based
- Sample sort: select p-1 splitters
- Radix sort: look at k bits at a time to place keys in 2^k buckets

Matrix Multiplication

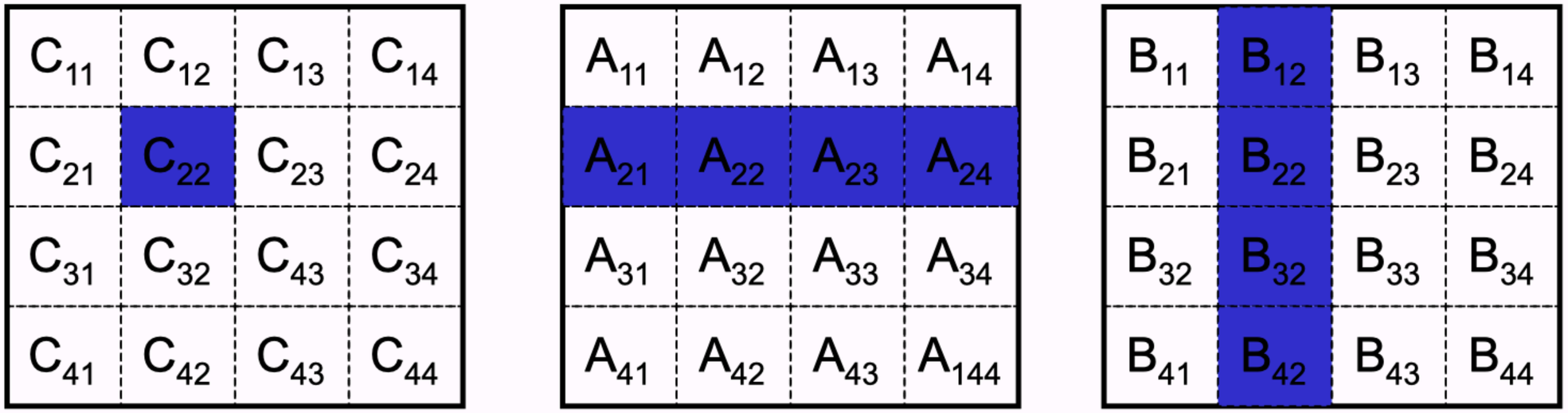
```
for (i=0; i<M; i++)
  for (j=0; j<N; j++)
    for (k=0; k<L; k++)
      C[i][j] += A[i][k]*B[k][j];
```



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

Blocking to improve cache performance

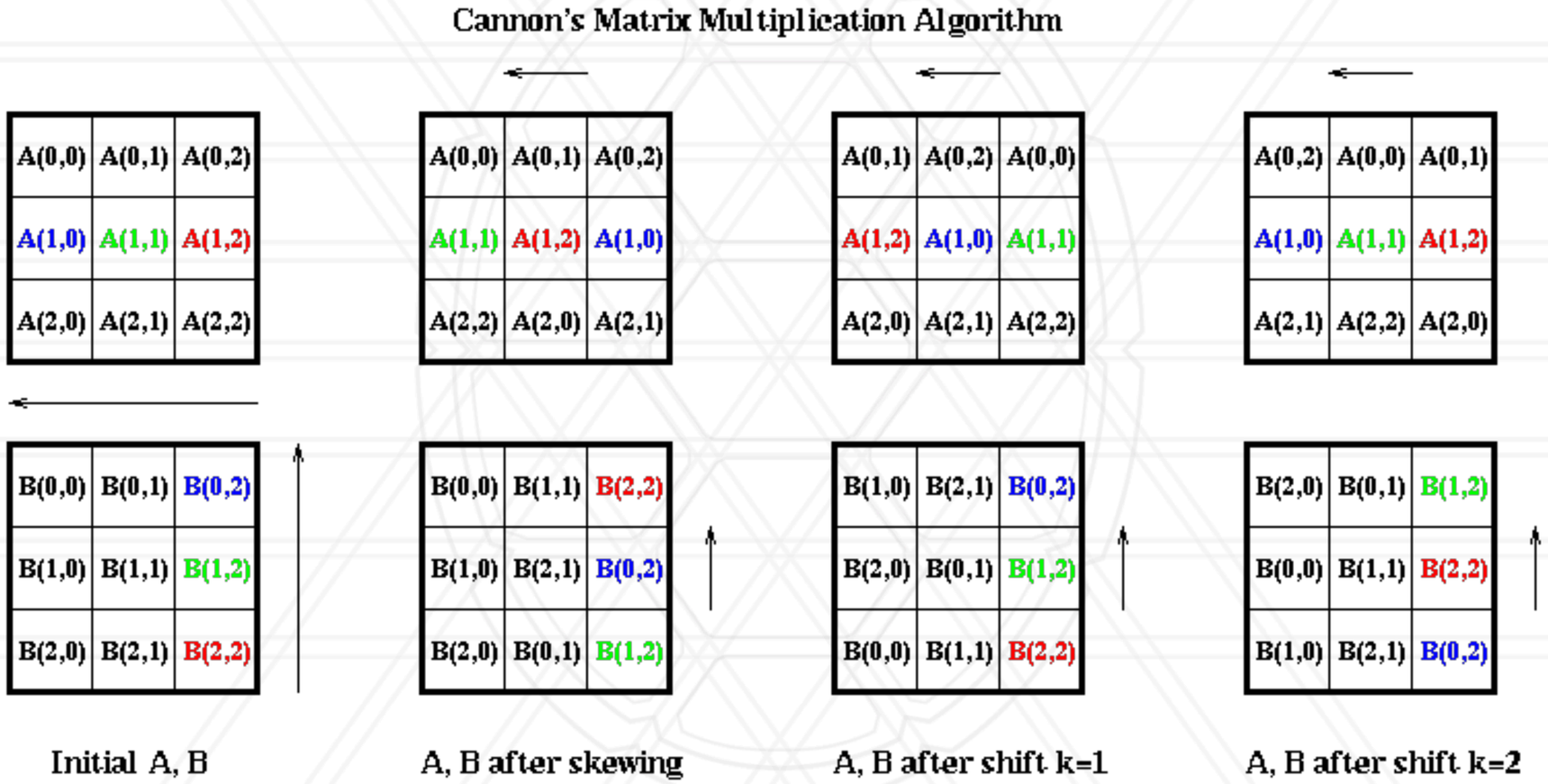
- Create smaller blocks that fit in cache
- $C_{22} = A_{21} * B_{12} + A_{22} * B_{22} + A_{23} * B_{32} + A_{24} * B_{42}$



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

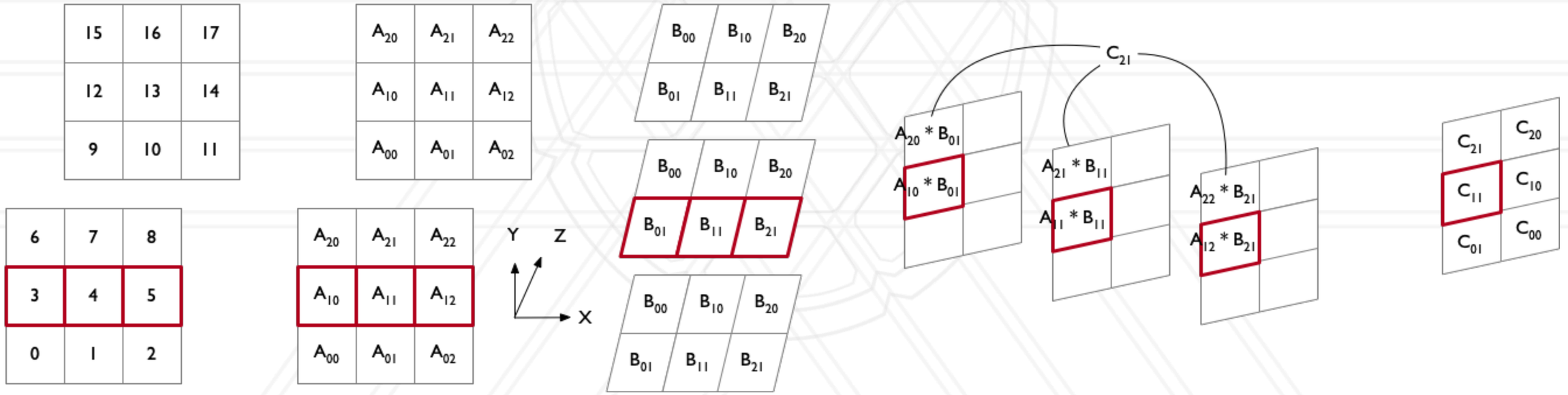
Cannon's 2D Matrix Multiply



http://people.eecs.berkeley.edu/~demmel/cs267/lecture11/lecture11.html

Agarwal's 3D Matrix Multiply

- Copy A to all XY planes and B to all XZ planes
- Perform a single matrix multiply to calculate partial C
- All-to-all along YZ planes to calculate final result



Questions

Online lecture: http://people.eecs.berkeley.edu/~demmel/cs267/lecture11/lecture11.html

- What does gravity on a hypercube mean?
- For 1d blocked layout on a ring, should the copy of A(MYPROC) to T take some time? In this case, will the total time of this algorithm be closer to the total time using 1d blocked layout on a bus with broadcoast?
- I am confused with the notations of the parts of matrices A, B and C: "let B(i) denote the n-by-(n/p) part of matrix B owned by processor i, where i runs from 0 to p-1. A(i) and C(i) are analogous." According to the figure, B is divided into vertical stripes. Is A divided into horizontal stripes? What about C?
- The paper uses synchronous send and receive (p. 2). Is it possible to get even better performance by using asynchronous send/receive and appropriate waits?
- What is the best practice to distribute the work of a 2D task when the number of processors is not a perfect square?
- If we would like to implement matrix multiplication on multiple GPUs installed on a single machine, and the matrices cannot fit into the memory of a single GPU, what kind of interconnection discussed in the paper is the closest to this situation? Or is it totally different?

Questions

A three-dimensional approach to parallel matrix multiplication

- As shown in figure 1, it seems that we need to make a copy of matrix A along the d2 axis. Does it mean that If we are dealing with a large matrix, each processor has to store a large amount of data?
- Under what conditions, we should choose 2d algorithm rather than 3d algorithm?
- How robust in terms of performance is the proposed algorithm under network congestion? It seems that operations such as all-gather and all-to-all might be bottlenecks, but they are performed group by group, not global, so I am not sure.
- It is mentioned that the Winograd variant of Strassen's algorithm is used for local submatrix multiplication. Is it practical to parallelize this algorithm? Will it bring even higher efficiency?
- In Table 1, why do the authors show the performance of cases such as C = C + AB and C = C + A T B? How does transposing the matrices matter? I also do not see the main differences in the performance numbers.
- As the hardware has improved a lot in terms of computation power, do people still distribute matrices of dimension of several thousand across multiple nodes to perform multiplication? Or it is more efficient to multiple such "small" matrices in a single node so that the communication costs are largely reduced?

Questions?

UNIVERSITY OF
MARYLAND

Abhinav Bhatele

5218 Brendan Iribe Center (IRB) / College Park, MD 20742

phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu