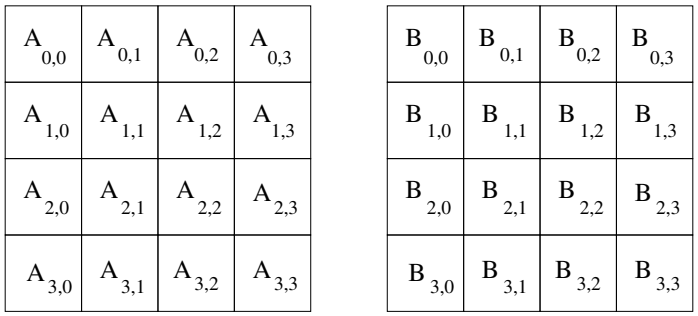
CSC346/646 Analysis of Cannon's Algorithm Spring 2015

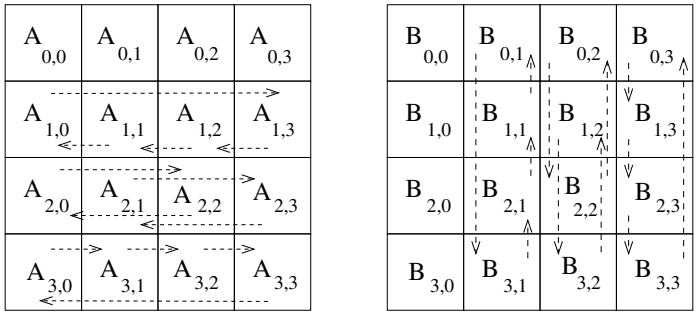
Cannon's algorithm is a parallel message-passing algorithm for matrix-matrix multiplication $C = AB$. Cannon's algorithm uses a block-distribution of the input matrices A and B, and the output C. For simplicity let us assume that the number of processors $p = 4^k$ for some $k \geq 0$; i.e., the number of processors is both perfect square and a power of two. Assume the underlying topology is a $2k$-dimensional hypercube.

Also assume the two matrices A and B are $n \times n$ square matrices and n is a multiple of $\sqrt{p}$. These assumptions allow a uniform block decomposition.

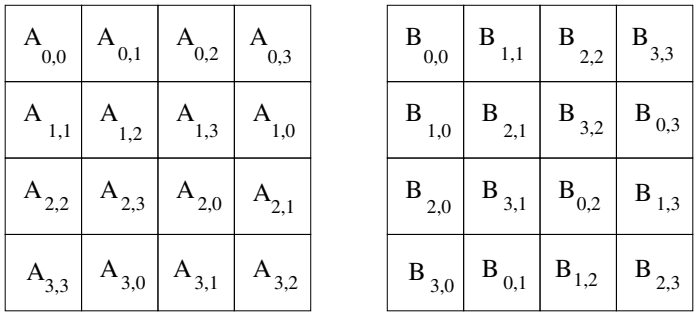
Step 0 Initial block-distribution of A and B.



Step 1 Shift the blocks to get the initial alignment.

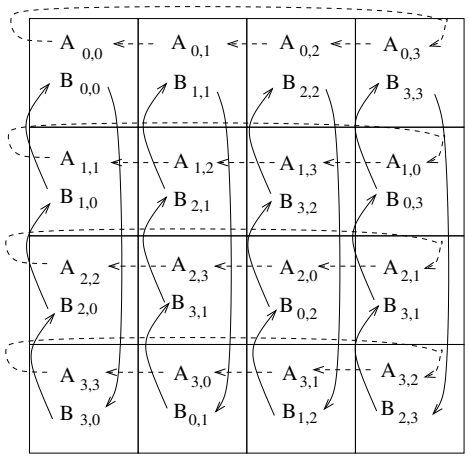


After Step 1 Initial alignment of A and B.



$A_{0,0}$ $B_{0,0}$	$A_{0,1}$ $B_{1,1}$	$A_{0,2}$ $B_{2,2}$	$A_{0,3}$ $B_{3,3}$
$A_{1,1}$ $B_{1,0}$	$A_{1,2}$ $B_{2,1}$	$A_{1,3}$ $B_{3,2}$	$A_{1,0}$ $B_{0,3}$
$A_{2,2}$ $B_{2,0}$	$A_{2,3}$ $B_{3,1}$	$A_{2,0}$ $B_{0,2}$	$A_{2,1}$ $B_{3,1}$
$A_{3,3}$ $B_{3,0}$	$A_{3,0}$ $B_{0,1}$	$A_{3,1}$ $B_{1,2}$	$A_{3,2}$ $B_{2,3}$

Compute Local Matrix Products and Accumulate



Shift: A rotates left, B rotates up.

$A_{0,1}$ $B_{1,0}$	$A_{0,2}$ $B_{2,1}$	$A_{0,3}$ $B_{3,2}$	$A_{0,0}$ $B_{0,3}$
$A_{1,2}$ $B_{2,0}$	$A_{1,3}$ $B_{3,1}$	$A_{1,0}$ $B_{0,2}$	$A_{1,1}$ $B_{3,1}$
$A_{2,3}$ $B_{3,0}$	$A_{2,0}$ $B_{0,1}$	$A_{2,1}$ $B_{1,2}$	$A_{2,2}$ $B_{2,3}$
$A_{3,0}$ $B_{0,0}$	$A_{3,1}$ $B_{1,1}$	$A_{3,2}$ $B_{2,2}$	$A_{3,3}$ $B_{3,3}$

After Shift, Compute Local Matrix Products and Accumulate

The algorithm continues to shift, compute local matrix products, and accumulate. After four multiply-accumulate steps, the distributed matrix-matrix product is complete.

Your Task:

Analyze the asymptotic time complexity of this algorithm in terms of problem size n and number of processors p. Use the communication cost models for a mesh embedded in a hypercube. Use our analysis of matrix-vector multiplication as an example for deriving the required analytical quantities. Include:

1. Give an expression for:
   (a) Sequential time
   (b) Parallel time
   (c) Speedup
   (d) Parallel cost
   (e) Overhead
   (f) Efficiency
2. Write the iso-efficiency equation for this algorithm
3. What can you conclude about the relationship between problem size n and number of processors p to ensure that efficiency converges to a constant which is greater than zero?