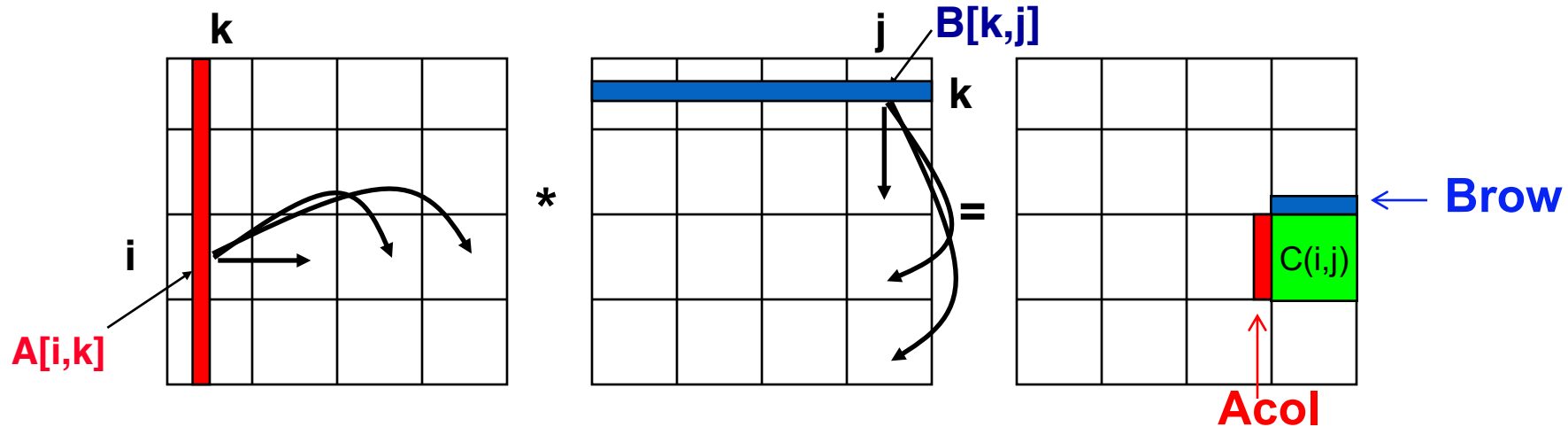


Lab 3

- Assigned on Feb. 16
- Due midnight on March 1 (2 weeks)
- Use MPI to implement the lectured SUMMA algorithm
 - You may assume $N \times N$ matrix.
 - You may assume $r \times c$ processes and $r=c$. (up to 16×16).
 - You may assume N is divisible by b (b is the block size).
 - Local $b \times b$ matrix multiplication should utilize Cray LibSci's `dgemm` function.
 - In your report, show your pseudo code, describe how you implement it, show your performance results (Given different N , different #processes, what is the best performance in Gflops), and discuss the effect of b .
 - Explain the figures for why your results are like this
 - What is the parallel efficiency of your SUMMA program
 - Write down how you tested if your computation result is correct or not.
- I have attached two slides from our previous lectures.

SUMMA— $n \times n$ matmul on $P^{1/2} \times P^{1/2}$ grid



For $k=0$ to $n/b-1$

for all $i = 1$ to $P^{1/2}$

owner of $A[i,k]$ broadcasts it to whole processor row (using binary tree)

for all $j = 1$ to $P^{1/2}$

owner of $B[k,j]$ broadcasts it to whole processor column (using bin. tree)

Receive $A[i,k]$ into $Acol$

Receive $B[k,j]$ into $Brow$

$C_{myproc} = C_{myproc} + Acol * Brow$

SUMMA Costs

```
for k=0 to n/b-1
  for all i = 1 to  $P^{1/2}$ 
    owner of A[i,k] broadcasts it to whole processor row (using binary tree)
    ... #words =  $\log(P^{1/2}) * b * n / P^{1/2}$  ,    #messages =  $\log(P^{1/2})$ 
  for all j = 1 to  $P^{1/2}$ 
    owner of B[k,j] broadcasts it to whole processor column (using bin. tree)
    ... same #words and same #messages
  Receive A[i,k] into Acol
  Receive B[k,j] into Brow
  C_myproc = C_myproc + Acol * Brow    ... #flops =  $2n^2 * b / P$ 
```

- Total #words = $\log P * n^2 / P^{1/2}$ (for n/b iterations)
 - Within factor of $\log P$ of lower bound
 - (more complicated implementation removes $\log P$ factor)
- Total #messages = $\log P * n/b$
 - Choose b close to maximum, $n/P^{1/2}$, to approach lower bound $P^{1/2}$
 - Show the lower bound later
- Total #flops = $2n^3/P$

Reference

- Read the SUMMA paper from van de Geijn and Watts
 - www.netlib.org/lapack/lawns/lawn96.ps
- Simplification: you don't have to call MPI_Bcast to broadcast messages. Just use point2point message passing functions.
- <http://www.cs.berkeley.edu/~knight/cs267/hw1/dgemm-blas.c>
- Fortran implementation of pdgemm from ScaLAPACK:
 - It might be too complicated to be helpful. Just for your interest.
http://www.netlib.org/scalapack/explore-html/d6/da2/pdgemm__8c_source.html