Matrix Multiplication

CS 625

April, 2014

Outline

Basic Serial Algorithm

```
for (i = 0; i < n; i++)
   for (j = 0; j < n; j++) {
      C[i,j] = 0;
      for (k = 0; k < n; k++)
         C[i,j] += A[i,k]*B[k,j];
   }
```

Reordered Serial Algorithm

```
for (j = 0; j < n; j++)
   for (i = 0; i < n; i++) {
      C[i,j] = 0;
      for (k = 0; k < n; k++)
         C[i,j] += A[i,k]*B[k,j];
   }
```

First Parallel Algorithm

```
// Matrices are distributed by block rows
for (j = 0; j < n; j++) {
   jth_col_B = Allgather column j of B;
   // Work through my rows of A
   for (local_i = 0; local_i < local_n; local_i++) {
      local_C[local_i,j] = 0;
      for (k = 0; k < n; k++)
         local_C[local_i,j] +=
            local_A[local_i,k]*jth_col_B[k];
   }
}
```

- Run-time of serial algorithm: $n^3 t_a$, where t_a is the cost of a floating point add and multiply.
- Run-time of first parallel algorithm:

$$\frac{n^3}{p} t_a + n \log(p) t_s + n^2 t_w$$

Block Submatrix Algorithm

```
Allgather(my_block_rowA, my_blockA, row_comm);
Allgather(my_block_colB, my_blockB, col_comm);
Mat_mult(my_blockC, my_block_rowA, n/p, n,
                    my_block_colB, n, n/p);
```

- Run-time:

$$\frac{n^3}{p} t_a + \log(p) t_s + \frac{2}{\sqrt{p}} n^2 t_w$$

- Big win in terms of communication costs.

Fox's Algorithm

```
// My proc row = i, my proc col = j
dest = (i-1 + q) % q;  // Dest for curr submat of B
source = (i+1) % q;    // Source for next submat of B
C_ij = 0;
B_k_bar_j = B_ij;
for (k = 0; k < q; k++) {
   k_bar = (k+i) % q;
   A_i_k_bar = Broadcast A_i_k_bar across ith row
      from process k_bar;
   C_ij += A_i_k_bar * B_k_bar_j;
   Send old B_k_bar_j to dest;
   Recv new B_k_bar_j from source;
}
```

$$
\begin{aligned}
T_{\text{blk\_row}}(n,p) &= \frac{n^3}{p}t_a + n\log(p)t_s + n^2 t_w \\
T_{\text{blk\_subm}}(n,p) &= \frac{n^3}{p}t_a + \log(p)t_s + \frac{2}{\sqrt{p}}n^2 t_w \\
T_{\text{Fox}}(n,p) &= \frac{n^3}{p}t_a + \left[\frac{1}{2}\log(p) + 1\right]\sqrt{p}t_s \\
&+ \frac{\frac{1}{2}\log(p)+1}{\sqrt{p}}n^2 t_w \\
&\approx \frac{n^3}{p}t_a + \frac{1}{2}\log(p)\sqrt{p}t_s + \frac{\log(p)}{2\sqrt{p}}n^2 t_w.
\end{aligned}
$$

Cannon's Algorithm

$C_{00} += A_{00}B_{00}$	$C_{01} += A_{01}B_{11}$	$C_{02} += A_{02}B_{22}$
$C_{10} += A_{11}B_{10}$	$C_{11} += A_{12}B_{21}$	$C_{12} += A_{10}B_{02}$
$C_{20} += A_{22}B_{20}$	$C_{21} += A_{20}B_{01}$	$C_{22} += A_{21}B_{12}$

$C_{00} += A_{01}B_{10}$	$C_{01} += A_{02}B_{21}$	$C_{02} += A_{00}B_{02}$
$C_{10} += A_{12}B_{20}$	$C_{11} += A_{10}B_{01}$	$C_{12} += A_{11}B_{12}$
$C_{20} += A_{20}B_{00}$	$C_{21} += A_{21}B_{11}$	$C_{22} += A_{22}B_{22}$

$C_{00} += A_{02}B_{20}$	$C_{01} += A_{00}B_{01}$	$C_{02} += A_{01}B_{12}$
$C_{10} += A_{10}B_{00}$	$C_{11} += A_{11}B_{11}$	$C_{12} += A_{12}B_{22}$
$C_{20} += A_{21}B_{10}$	$C_{21} += A_{22}B_{21}$	$C_{22} += A_{20}B_{02}$

```
// I'm in process row my_row and process column my_col
q = sqrt(p);
Left_shift local_A by my_row slots;
Up_shift local_B by my_col solts;
local_C = local_A*local_B;
for (i = 1; i < q; i++) {
   Left_shift(local_A, 1, my_row_comm);
   Up_shift(local_B, 1, my_col_comm);
   local_C += local_A*local_B;
}
```

$$
\begin{aligned}
T_{\text{blk\_row}}(n,p) &= \frac{n^3}{p}t_a + n\log(p)t_s + n^2 t_w \\
T_{\text{blk\_subm}}(n,p) &= \frac{n^3}{p}t_a + \log(p)t_s + \frac{2}{\sqrt{p}}n^2 t_w \\
T_{\text{Fox}}(n,p) &\approx \frac{n^3}{p}t_a + \frac{1}{2}\log(p)\sqrt{p}t_s + \frac{\log(p)}{2\sqrt{p}}n^2 t_w \\
T_{\text{Cannon}}(n,p) &= \frac{n^3}{p}t_a + 2\sqrt{p}t_s + \frac{2}{\sqrt{p}}n^2 t_w
\end{aligned}
$$