CSCI 251: Concepts of Parallel and Distributed Systems

Alvin Lin

September 11th, 2017

Topics

- Matrix Multiplication
- Communication/Shared Memory Costs
- Solving Systems of Linear Equations
- Bitonic Sort

Matrix Multiplication

We discussed the case where the number of processors is equal to the number of rows in the matrix. In the case where the number of rows N is greater than the number of processors P, we divide N by P and assign $\frac{N}{P}$ rows to each process.

$$[A] \times [B] = [C]$$

We compute $\frac{N}{P}$ rows of the result matrix C on each processor. This operation is $O(n^3)$ but parallelizing it brings its efficiency close to 1. With P processes, the B matrix must be available to each process in order for the computation to be performed.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix}$$

To compute the first row of the result matrix, we only need the first row of A, but we need the entire B matrix. Row A_i is given to process P_i, but B must be distributed all the processes, which is the cost of this computation.

Communication between Processes

Suppose we have processors P_i and P_j, between which data much be shared. Physically, they can be separate processes on two computers that send messages to each other. This can also be done with shared memory, a region to which both P_i and P_j have read/write access. An important note is that the shared memory access must be synchronized (atomic). There are many types of interprocess communications and these are just a few. Processors can communicate single messages, broadcast one-to-many, or communicate many-to-one.

Message Passing

- Limited by network bandwidth
- Limited by network topology

Shared Memory

- Limited by synchronization
- Limited by memory latency

System of Linear Equations

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3 \\
a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= b_4
\end{aligned}
$$

This can be represented as the following matrix:

$$
\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}
$$

$$
Ax = b
$$

We can reduce this matrix to an upper triangular matrix for parallelization.

$$\begin{bmatrix} 1 & u_{12} & u_{13} & u_{14} \\ 0 & 1 & u_{23} & u_{24} \\ 0 & 0 & 1 & u_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$\begin{aligned} x_1 + u_{12}x_2 + u_{13}x_3 + u_{14}x_4 &= y_1 \\ x_2 + u_{23}x_3 + u_{24}x_4 &= y_2 \\ x_3 + u_{34}x_4 &= y_4 \\ x_4 &= y_4 \end{aligned}$$

From this, we know what x_4 is and we can solve the systems of equations. To compute the triangular matrix we do the following operation:

$$A[i,j] := A[i,j] - A[i,k] \times A[k,j]$$

Example

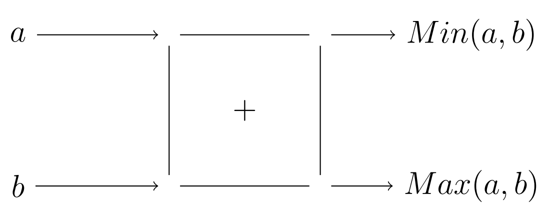
$$\begin{bmatrix} 3 & 4 & 2 & 7 \\ 2 & 5 & 1 & 2 \\ 1 & 2 & 378 & \\ 4 & 1 & 4 & 3 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 20 \\ 14 \\ 10 \\ 8 \end{bmatrix}$$

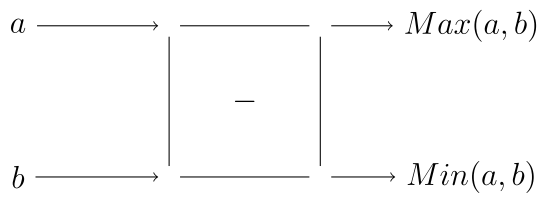
Convert to an upper triangular matrix:

$$\begin{bmatrix} 1 & \frac{4}{3} & \frac{2}{3} & \frac{7}{3} \\ 0 & 1 & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \frac{20}{3} \\ \dots \\ \dots \\ \dots \end{bmatrix}$$

Sorting

The basic part of a sorting algorithm is a **comparator**. A comparator, given two inputs a and b, can give either the minimum or maximum of a and b.





Reminders

Professor Mohan Kumar:
`mjkvcs@rit.edu`
`https://cs.rit.edu/~mjk`

Rahul Dashora (TA):
`rd5476@mail.rit.edu`

We will discuss POSIX next week and the project will be handed out.

You can find all my notes at `http://omgimanerd.tech/notes`. If you have any questions, comments, or concerns, please contact me at alvin@omgimanerd.tech