Introduction to Parallel Program Analysis

Ned Nedialkov

McMaster University
Canada

CS/SE 4F03
February 2016

Outline

Parallel matrix-vector product

1D distribution

Analysis

2D distribution

Parallel matrix-vector product

- Consider parallel matrix-vector multiplication
- Let $A \in \mathbb{R}^{n \times n}$ and $x \in \mathbb{R}^n$
- We wish to compute $y = Ax$ in parallel and analyze scalability
- We consider 1D and 2D distribution schemes

1D distribution

- Assume that process i stores n_i rows of A and n_i rows of x
- Then i computes $y_i = A_i x_i$

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{p-1} \end{bmatrix} = \begin{bmatrix} A_0 \\ A_1 \\ \vdots \\ A_{p-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{p-1} \end{bmatrix}$$

$A_i \in \mathbb{R}^{n_i \times n}$, $x, y \in \mathbb{R}^{n_i}$

- Algorithm
  1. Process i gathers x
  2. Process i computes $y_i = A_i x_i$

Analysis

- Step 1 can be done using all-to-all broadcast
  - Assume that all-to-all broadcast of m words takes time

$$t_s \log_2 p + t_w m(p-1) \tag{1}$$

  t_s is start-up time, t_w is pre-word transfer time
  - Assume $n_i = n/p$. Then $m = n/p$, and (1) becomes

$$\begin{aligned} t_s \log_2 p + t_w m(p-1) &= t_s \log_2 p + t_w \frac{n}{p}(p-1) \\ &\approx t_s \log_2 p + t_w n \end{aligned} \tag{2}$$

- Step 2 can be done in $\approx (n/p)n$ operations:

$$\frac{n^2}{p} \tag{3}$$

  For simplicity, we omit constants. For example, this computation is more like $2n^2/p\gamma$, where γ is the time per arithmetic operation

From (2, 3), the parallel time is

$$T_p = \frac{n^2}{p} + t_s \log_2 p + t_w n$$

Since the serial time is $T_s = n^2$, the speed up is

$$S = \frac{T_s}{T_p} = \frac{n^2}{\frac{n^2}{p} + t_s \log_2 p + t_w n} = \frac{1}{\frac{1}{p} + t_s \frac{\log_2 p}{n^2} + t_w \frac{1}{n}}$$

The efficiency is

$$E_{1\mathrm{D}} := \frac{S}{p} = \frac{1}{1 + t_s \frac{p \log_2 p}{n^2} + t_w \frac{p}{n}} \tag{4}$$

Scalability

- For fixed n, $E_{1\mathrm{D}}$ decreases as p increases
  Not strongly scalable
- Denote the work for problem of size n by $W(n) = n^2$
  Assume we double n. Then

  $$W(2n) = 4n^2 = 4W(n)$$

- Since the work quadruples, we increase the number of processes to $4p$
  Then

$$
\begin{aligned}
E'_{1\mathrm{D}} &= \frac{1}{1 + t_s \frac{4p \log_2(4p)}{(2n)^2} + t_w \frac{4p}{2n}} = \frac{1}{1 + t_s \frac{p(2+\log_2 p)}{n^2} + t_w \frac{2p}{n}} \\
&= \frac{1}{\underbrace{(1 + t_s \frac{p \log_2 p}{n^2} + t_w \frac{p}{n})}_{\text{same as in } E_{1\mathrm{D}}} + (t_s \frac{2p}{n^2} + t_w \frac{p}{n})}
\end{aligned}
$$

- Obviously $E'_{1D} < E_{1D}$ due to

$$t_s \frac{2p}{n^2} + t_w \frac{p}{n}$$

- Let M be the number of words that can be stored per node
- On p nodes, we can store pM words
- Let N be the size of the largest problem we can store on p nodes
- Assume we store N^2 items, that is A. Then $N^2 = pM$ and $N = \sqrt{pM}$
  We ignore the storage for the x_i
- Using $n = N$ in (4), we obtain

$$E''_{\mathrm{1D}} = \frac{1}{1 + t_s \frac{\log_2 p}{M} + t_w \frac{\sqrt{p}}{\sqrt{M}}} \tag{5}$$

- $\lim_{p\to\infty} E''_{\mathrm{1D}} = 0$
- This algorithm does not scale well

2D distribution

- Consider a 2D grid of processs
- For simplicity, assume $q \times q = p$ processs
- Process (i, j) stores submatrix $A_{ij} \in \mathbb{R}^{n_i \times m_i}$
- Process (j, j) stores subvector $x_j \in \mathbb{R}^{n_j}$
- This distribution can be visualized as

$$
\begin{array}{llll}
A_{0,0}, x_0 & A_{0,1} & \dots & A_{0,q-1} \\
A_{1,0} & A_{1,1}, x_1 & \dots & A_{1,q-1} \\
\vdots & \vdots & \dots & \vdots \\
A_{q-1,0} & A_{q-1,1} & \dots & A_{q-1,q-1}, x_{q-1}
\end{array}
$$

Algorithm

1. Process (j, j) broadcasts x_j along column j

$$
\begin{array}{cccc}
A_{0,0}, x_0 & A_{0,1}, x_1 & \dots & A_{0,q-1}, x_{q-1} \\
A_{1,0}, x_0 & A_{1,1}, x_1 & \dots & A_{1,q-1}, x_{q-1} \\
\vdots & \vdots & & \\
A_{q-1,0}, x_0 & A_{q-1,1}, x_1 & \dots & A_{q-1,q-1}, x_{q-1}
\end{array}
$$

2. Process (i, j) computes $A_{ij}x_j$
3. Process (i, i) does “sum" reduction across row i. Then (i, i) contains y_i:

$$y_i = A_{i,0}x_0 + A_{i,1}x_1 + \cdots + A_{i,q-1}x_{q-1}$$

Analysis

Assume $n_i, m_i = n/q$

- Assume one-to-all broadcast of m words takes

$$(t_s + t_w m)\log_2 p \tag{6}$$

Here $m = n/q = n/\sqrt{p}$ and we broadcast along $q = \sqrt{p}$ nodes
Then (6) becomes

$$\begin{aligned}(t_s + t_w m)\log_2 p &= \left(t_s + t_w \frac{n}{q}\right)\log_2 q \\ &= \left(t_s + t_w \frac{n}{\sqrt{p}}\right)\log_2 \sqrt{p} \\ &= \frac{1}{2}\left(t_s + t_w \frac{n}{\sqrt{p}}\right)\log_2 p\end{aligned} \tag{7}$$

- $A_{ij}x_j$ takes

$$\left(\frac{n}{q}\right)^2 = \frac{n^2}{p} \tag{8}$$

- All-to-one reduction is like one-to-all broadcast:

$$\frac{1}{2}\left(t_s + t_w \frac{n}{\sqrt{p}}\right)\log_2 p \tag{9}$$

We ignore the time for sumations

The parallel time is (7) + (8) + (9):

$$T_p = \frac{n^2}{p} + \left(t_s + t_w \frac{n}{\sqrt{p}}\right)\log_2 p$$

Speedup is

$$S = \frac{T_s}{T_p} = \frac{n^2}{\frac{n^2}{p} + \left(t_s + t_w \frac{n}{\sqrt{p}}\right) \log_2 p} = \frac{1}{\frac{1}{p} + \left(t_s + t_w \frac{n}{\sqrt{p}}\right) \frac{\log_2 p}{n^2}}$$

Efficiency is

$$E_{2D} = \frac{1}{1 + t_s \frac{p \log_2 p}{n^2} + t_w \frac{\sqrt{p} \log_2 p}{n}} \tag{10}$$

What happens if we increase n to $2n$ and p to $4p$ compared to increasing in

$$E_{1D} = \frac{1}{1 + t_s \frac{p \log_2 p}{n^2} + t_w \frac{p}{n}} \; ?$$

- As before, assume $n = \sqrt{pM}$
  Then

$$E''_{\text{2D}} = \frac{1}{1 + t_s \frac{p \log_2 p}{pM} + t_w \frac{\sqrt{p} \log_2 p}{\sqrt{pM}}}$$
$$= \frac{1}{1 + t_s \frac{\log_2 p}{M} + t_w \frac{\log_2 p}{\sqrt{M}}}$$

- $\log_2$ is a very slowly growing function, and can be considered as a constant here
- As p increases, the efficiency decreases very slowly and much slower than E'_{1D} in (5)
- For practical purposes, this algorithm scales well