CS 525 Homework 4

Junchao Yan
yan114@purdue.edu

March 12, 2014

1 Send & Receive

Average time and standard deviation:

Size	Average time	Standard deviation
1024	3.705502e-04	1.806665e-05
4096	1.084089e-03	5.289455e-05
16384	2.887440e-03	4.115585e-04
65536	9.555793e-03	5.318550e-04
262144	3.633275e-02	5.025303e-04
1048576	1.432897e-01	4.689201e-04

Table 1: Observed time for ping-pong experiment

Estimate t_s and t_w:

$$A = \begin{bmatrix} 1 & 1024 \\ 1 & 4096 \\ 1 & 16384 \\ 1 & 65536 \\ 1 & 262144 \\ 1 & 1048576 \end{bmatrix} b = \begin{bmatrix} 3.705502e-04 \\ 1.084089e-03 \\ 2.887440e-03 \\ 9.555793e-03 \\ 3.633275e-02 \\ 1.432897e-01 \end{bmatrix}$$

$$x = 2\begin{bmatrix} t_s \\ t_w \end{bmatrix} = A^{-1}b = \begin{bmatrix} 5.2887270e-04 \\ 1.3618009e-07 \end{bmatrix}$$

Therefore, $t_s = 2.644364e-04$, $t_w = 6.809005e-08$.

2 Unitest & Bitest & Scantest

(a) **Unitest & Bitest**

Unitest

Size	Average time	Standard deviation
1024	1.614380e-03	3.766816e-04
4096	4.403663e-03	6.963426e-04
16384	1.153657e-02	1.839524e-03
65536	3.828001e-02	2.194081e-03
262144	1.452596e-01	2.018418e-03
1048576	5.731775e-01	2.270290e-03

Table 2: Observed time for uni-directional ring

The time complexity for uni-directional ring is $P * (t_s + M * t_w)$. Therefore, the estimated times are

Size	Estimated time
1024	2.673285e-03
4096	4.346666e-03
16384	1.104019e-02
65536	3.781429e-02
262144	1.449107e-01
1048576	5.732962e-01

Table 3: Estimated time for uni-directional ring

Compared to the observed time, the estimated times are pretty close to the observed ones except the one when size is 1024. The reason is that when size is small, the overheads are more significant.

Bitest

Size	Average time	Standard deviation
1024	8.388281e-04	1.946325e-04
4096	2.279544e-03	2.644261e-04
16384	6.439471e-03	8.186314e-04
65536	2.001550e-02	9.967670e-04
262144	7.276092e-02	1.284534e-03
1048576	2.866852e-01	1.258708e-03

Table 4: Observed time for bi-directional ring

The time complexity for uni-directional ring is $\frac{P}{2} * (t_s + M * t_w)$. The estimated times are shown in Table 5.

Size	Estimated time
1024	1.336642e-03
4096	2.173333e-03
16384	5.520095e-03
65536	1.890714e-02
262144	7.245534e-02
1048576	2.866481e-01

Table 5: Estimated time for bi-directional ring

Similarly, by comparing the estimated time with the observed time, we can find that the estimated times are pretty close to the observed ones except the one when size is 1024. The reason is that when size is small, the overheads are more significant.

(b) **Scantest**

Size	Average time	Standard deviation
1024	1.554966e-03	3.064169e-04
4096	3.941464e-03	6.286839e-04
16384	1.049695e-02	1.401624e-03
65536	3.485785e-02	2.196855e-03
262144	1.359215e-01	2.356497e-03
1048576	5.522718e-01	5.412212e-03

Table 6: Observed time for prefix sum

By comparing the time taken by prefix sum with the time taken by two rings, it seems like the prefix sum computation is implemented using uni-directional ring.

3 Time complexity

1. **Ring**

   There are $\log_2 P$ steps in total. For each step, there are $P/2$, $P/2^2$, ..., $P/2^{\log_2 P}$ hops, respectively. Therefore, the time complexity is

$$
\begin{aligned}
t_{comm} &= \log_2 P(t_s + m * t_w) + (P/2 + P/2^2 + ... + P/2^{\log_2 P}) * t_h \\
&= \log_2 P(t_s + m * t_w) + (1/2 + 1/2^2 + ... + 1/2^{\log_2 P}) * P * t_h \\
&= \log_2 P(t_s + m * t_w) + (1 - \frac{1}{P}) * P * t_h \\
&= \log_2 P(t_s + m * t_w) + (P - 1) * t_h
\end{aligned}
$$

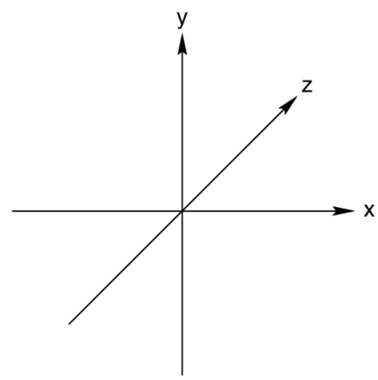

Figure 1: Coordinate for 3D torus

2. **3-dimensional torus**

   Suppose the coordinate (x, y, z) is denoted as above (Figure 1). The root is at the origin. There are three phases for the broadcasting. In the first phase, the root sends messages to all the processes along the x-axis. Then, processes (i, 0, 0) send the messages to all processes (i, j, 0). At last, processes (i, j, 0) send the messages to processes (i, j, k), where $0 \le i, j, k \le \sqrt[3]{P}$. Each phase applies the ring broadcasting.

   - Phase 1: $\frac{1}{3}\log_2 P(t_s + m * t_w) + (\sqrt[3]{P} - 1) * t_h$
   - Phase 2: $\frac{1}{3}\log_2 P(t_s + m * t_w) + (\sqrt[3]{P} - 1) * t_h$
   - Phase 3: $\frac{1}{3}\log_2 P(t_s + m * t_w) + (\sqrt[3]{P} - 1) * t_h$

   Therefore, the time complexity is $t_{comm} = \log_2 P(t_s + m * t_w) + 3(\sqrt[3]{P} - 1) * t_h$.

3. **Binary tree**

   There are $\log_2 P$ steps in total. For each step, there are $2\log_2 P$, $2\log_2 P/2$,$\cdots$,$2\log_2(P/2^{\log_2 P-1})$ hops, respectively. Therefore, the time complexity is

$$
\begin{aligned}
t_{comm} &= log_2 P(t_s + m * t_w) + 2(\log_2 P + \log_2 P/2+, ..., +1) * t_h \\
&= log_2 P(t_s + m * t_w) + 2(\log_2 P * \log_2 P - (1 + 2 + 3+, ..., +(\log_2 P - 1))) * t_h \\
&= log_2 P(t_s + m * t_w) + \log_2 P(\log_2 P + 1) * t_h
\end{aligned}
$$

4 Long message broadcasting

1. **Ring**

   The time complexity for each algorithm is:

- One to all broadcasting in a ring: $\log_2 P(t_s + M * t_w)$
- New algorithm:

  1. Scatter: $\log_2 P * t_s + \frac{M}{P} * (P-1) * t_w$
  2. All to all broadcasting: $(P-1)(t_s + \frac{M}{P} * t_w)$

Therefore, to ensure the new algorithm is faster than the original algorithm,

$$\log_2 P * t_s + \frac{M}{P} * (P-1) * t_w + (P-1)(t_s + \frac{M}{P} * t_w) < \log_2 P(t_s + M * t_w)$$

$$M > \frac{P-1}{\log_2 P + \frac{2}{P} - 2} * \frac{t_s}{t_w}$$

2. **2-dimensional torus**

Similarly, the time complexity for each algorithm is:

- One to all broadcasting in a 2D torus: $\log_2 P(t_s + M * t_w)$
- New algorithm:

  1. Scatter: In Phase 1, the root scatters the message to one direction (say, to the nodes in row 0). Then in Phase 2, these nodes scatter the message to the other nodes in the column.

     - Phase 1: $\frac{1}{2} \log_2 P * t_s + \frac{M}{\sqrt{P}} * (\sqrt{P} - 1) * t_w$
     - Phase 2: $\frac{1}{2} \log_2 P * t_s + \frac{M}{P} * (\sqrt{P} - 1) * t_w$
     - Total: $\log_2 P * t_s + \frac{P-1}{P} * M * t_w$

  2. Broadcasting:
     - Phase 1: all to all broadcast in every row like the ring algorithm $(\sqrt{P} - 1)(t_s + \frac{M}{P} * t_w)$
     - Phase 2: similarly, all to all broadcast in every column $(\sqrt{P} - 1)(t_s + \frac{M}{\sqrt{P}} * t_w)$
     - Total: $2(\sqrt{P} - 1) * t_s + \frac{P-1}{P} * M * t_w$

Therefore, to ensure the new algorithm is faster than the original algorithm,

$$\log_2 P * t_s + \frac{P-1}{P} * M * t_w + 2(\sqrt{P} - 1) * t_s + \frac{P-1}{P} * M * t_w < \log_2 P(t_s + M * t_w)$$

$$M > \frac{2(\sqrt{P}-1)}{\log_2 P + \frac{2}{P} - 2} * \frac{t_s}{t_w}$$