Homework 03: Communication Time Complexity

Sunday, March 20, 2022 7:26 PM

(1) Given the problem (whose computational time complexity was solved in class):

DATA: $\vec{x} = (-1, 2, 4, 1, 6, 0, -1, 0)$ → size 8

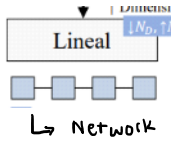
TASK: $\sum_{i=1}^{n=8} x_i = (-1) + 2 + 4 + 1 + 6 + 0 + (-1) + 0$

Calculate the communication time complexity $t_c(np, n)$ considering that the communication between two nodes takes 3 units of time as maximum, using the following communication network:

(a) Commuted - Direct - Linear

Solution

$np = 2^{kp} \rightarrow \log_2 np = \log_2 2^{kp} \rightarrow \log_2 np = kp$



Considering computation time complexity as:

$$t(np, n) = \left[\frac{n}{np} + 1\right] + kp = \left[\frac{n}{np} + 1\right] + \log_2 np$$

Case		Data/Diagram	time	Tree (sums)
(serial)	$kp = 0$ $2^{kp} = 2^0 = 1$	$\vec{x} = (-1\ 2\ 4, 1\ 6, 0, -1, 0)$	$t = 7(2)(3)$	0
(parallel)	$kp = 1$ $2^{kp} = 2^1 = 2$	return partial sum; pass half the data	$t = 6 = (1)(2)(3)$	1
$kp = 2$	$2^{kp} = 2^2 = 4$	0.25, 0.25, 0.25, 0.25	$t = 18 = 3(2)(3)$	2
$kp = 3$	$2^{kp} = 2^3 = 8$ → −1		$t = 42 = 7(2)(3)$	3
$kp = 4$	$2^{kp} = 16$		$15(2)(3)$	4

In general form, we can say

$$t_c = (2^{kp} - 1)(2)(3)$$
$$= (2^{kp} - 1)(6)$$
$$= 6(2^{kp} - 1)$$

and with $kp = \log_2 np$

$$t_c = 6(2^{\log_2(np)} - 1)$$

with logarithm cancelling

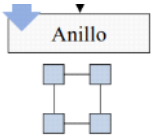
$$t_c(np, n) = \boxed{t_c = 6(np - 1)}$$

for communication time complexity in a commuted - Direct - Linear Ntwk.

Mariana Ávalos Arce
0197495

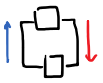
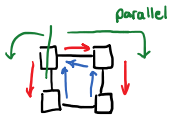
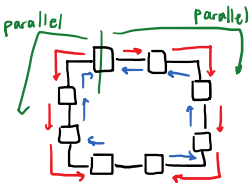
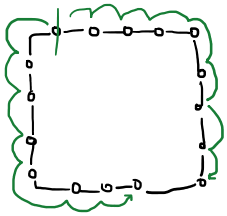
(b) Commuted - Direct - Ring

Solution



Considering computation time complexity as:

$$t(np, n) = \left[\frac{n}{np} + 1\right] + kp = \left[\frac{n}{np} + 1\right] + \log_2 np$$

Case		Data/Diagram	time	Tree (sums)
(serial)	$kp = 0$ $2^{kp} = 2^0 = 1$	P_1	$t = \emptyset$	$\emptyset$ O
(parallel)	$kp = 1$ $2^{kp} = 2^1 = 2$		$t = 6 = 1(2)(3) + \emptyset(2)(3)$, $\emptyset \to \frac{np}{2} - 1$	1
	$kp = 2$ $2^{kp} = 2^2 = 4$		$t = 1(2)(3) + 1(2)(3) = 12$, $1 \to \frac{np}{2} - 1$	2
	$kp = 3$ $2^{kp} = 2^3 = 8$		$t = 1(2)(3) + 3(2)(3) = 24$, $3 \to \frac{np}{2} - 1$	3
	$kp = 4$ $2^{kp} = 2^4 = 16$		$t = 1(2)(3) + 7(2)(3)$, $7 \to \frac{np}{2} - 1$	4

Thus, in a general way, we see that

$$t_c = (2)(3) + \left[\frac{np}{2} - 1\right](2)(3)$$

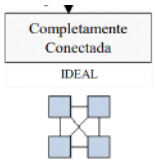
$$t_c = 6 + \left[\frac{np}{2} - 1\right]6$$

$$t_c = 6 + \frac{6np}{2} - 6$$

$$t_c(np, n) = \boxed{t_c = 3np}$$

for communication time complexity in a commuted - direct - ring network.

(c.) Completely connected



Considering computation time complexity as:

$$t(np, n) = \left[\frac{n}{np} + 1\right] + kp = \left[\frac{n}{np} + 1\right] + \log_2 np$$

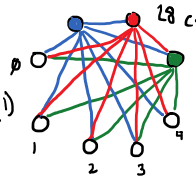
Case		Data/Diagram	time	Tree (sums)
(serial)	$kp = 0$ $2^{kp} = 2^0 = 1$	P_1	$t = \emptyset$	$\emptyset$ 1
(parallel)	$kp = 1$ $2^{kp} = 2^1 = 2$	1 conn $= \frac{2(1)}{2}$	$t = 3 + 3 = 6$ (send, 1 time; $1(3)$; $kp(3)$)	1
	$kp = 2$ $2^{kp} = 2^2 = 4$	6 conn $= \frac{4(3)}{2}$	$t = 3 + 3 + 3 = 9$ (send, 2, 1; $2(3)$; $kp(3)$)	2
	$kp = 3$ $2^{kp} = 2^3 = 8$	28 conn $= \frac{8(7)}{2}$	$t = 3 + 3 + 3 + 3 = 12$ (send, 4, 2, 1; $3(3)$; $kp(3)$)	3

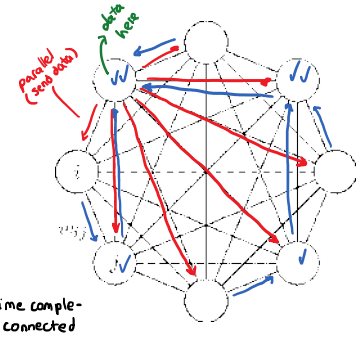
Gauss Sum: $\frac{n(n+1)}{2} \rightarrow \frac{(np-1)((np-1)+1)}{2} = \frac{(np-1)(np)}{2}$

Connections $= \frac{np(np-1)}{2}$

$7 + 6 + 5 + 4 + 3 + 2 + 1$

$\frac{8(7)}{2} = 28$





Therefore we can say that

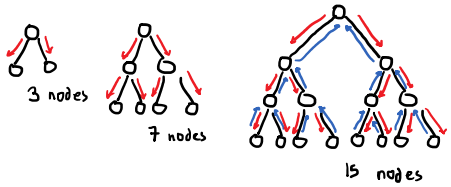
$t_c = 3 + kp(3)$ with $kp = \log_2 np$

$t_c = 3 + (\log_2 np)(3)$

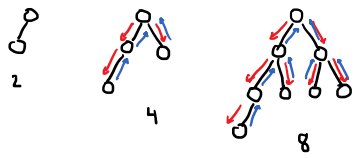
$t_c(np, n) = \boxed{t_c = 3(1 + \log_2 np)}$ for communication time complexity in a completely connected network.

Monday, March 21, 2022 6:54 PM

(d.) What's the problem when trying to use a communication network of type Commuted - Direct - Binary Tree?



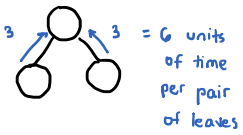
The first problem is that the number of elements (n size) must grow as $n = 2^{kn}$, but this will generate an unbalanced number of nodes in the tree, as shown below: the binary tree is not completely balanced if n grows as $n = 2^{kn}$.



The second, and probably more troubling, is that even if we have a balanced tree, the connections are done in such a way that, whenever a leaf node wants to send its partial sum to another processor, it's faster to send it to its parent node, but that means that the other child needs to wait until the sum of that child and its parent is done so that the partial sum of the other child can be sent to its parent too, which increases the synchronization among the leaves, which are the highest proportion of a tree's nodes:

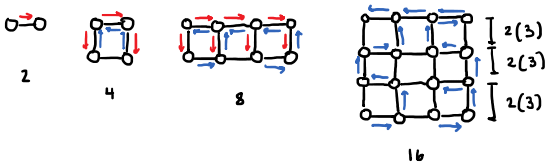
$$\text{number of leaves} = \text{int}\left(\frac{\text{nodes}}{2}\right) + 1$$

Which would double the communication time, instead of allowing the other child to send it to another available node.



And this would need to happen for any interior node that has two children. But since leaves are more than 50% of a tree, the communication time is very affected by this shape.

(e.) What's the problem when trying to use a communication network of type Commuted - Direct - 2D?



The problem is that, due to the rows of connections in this network, the only parallel communication to transfer partial results to other processors can be the communications per row. Thus, the next row of processors needs to wait for the communication of the below row to be done (3 seconds) until it can receive their partial sums, and so on. The synchronization time depends and grows with the amount of rows the grid has, which is a problem in big networks.

(f) Plot the complexities.

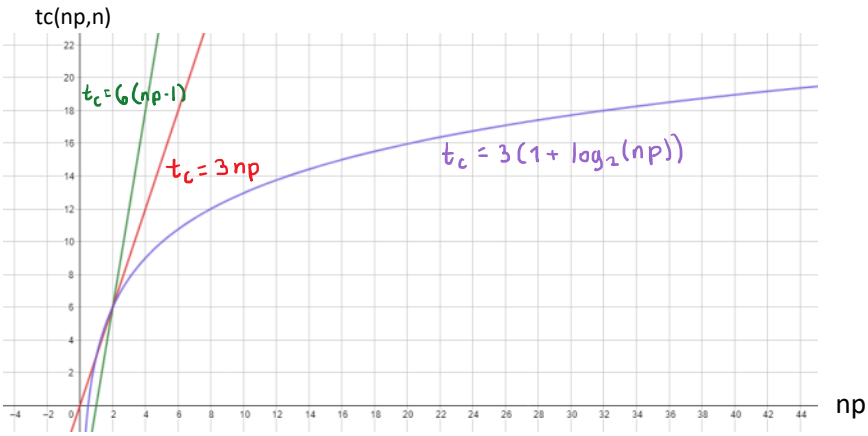
The time complexities for communication are:

$$t_c^a = 6(np - 1)$$

$$t_c^b = 3np$$

$$t_c^c = 3(1 + \log_2 np)$$

Which plotted together looks as below.



(g) Compare the results obtained in a, b and c:
Which network is the best? Which one the worst?

For the three networks, we got the following complexities:

a. Linear $t_c^a = 6(np - 1)$

b. Ring $t_c^b = 3np$

c. Completely Connected $t_c^c = 3(1 + \log_2 np)$

By looking at their models, we see that both the Linear and the Ring involve the np term raised to the power of 1, which means both are linear functions. Out of the two, the Linear function has slope of 6, while the Ring function has a slope of 3, meaning that the Ring function grows slower in time when np increases, suggesting that the Ring function is more recommended than the Linear. Finally, the Completely Connected function has a logarithmic behaviour, which grows slower in time as np increases than any of the two linear functions. Therefore, the best network is the Completely Connected, and the worst is the Linear Network, in terms of time complexity for communication.