Brian Crafton
High Performance Computing
2/5/16

1.c

After graphing running each program on the three problem sizes:

- 10^7
- 10^8
- 10*9

With thread counts of:

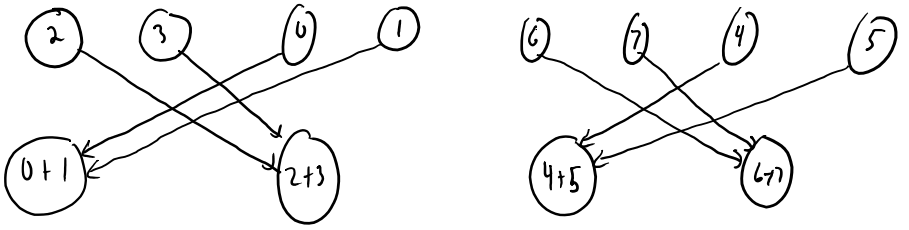
- 1
- 4
- 8
- 10
- 16
- 20
- 30
- 32

the results are shown below

Thread#	Problem Size	Sequential	A	B	min time by program	min time by thread# and program
1	10^7		0.17	0.075904	0.132275	0.075904	0.010087
4	10^7			0.042644	0.05101	0.042644	
8	10^7			0.018817	0.022522	0.018817	
10	10^7			0.012582	0.015107	0.012582	
16	10^7			0.010087	0.019562	0.010087	
20	10^7			0.011382	0.01874	0.011382	
30	10^7			0.012527	0.016266	0.012527	
32	10^7			0.035156	0.036309	0.035156	
1	10^8		0.78	0.750449	0.931806	0.750449	0.112177
4	10^8			0.408221	0.49833	0.408221	
8	10^8			0.168758	0.208453	0.168758	
10	10^8			0.20645	0.257055	0.20645	
16	10^8			0.112177	0.135456	0.112177	
20	10^8			0.154891	0.171097	0.154891	
30	10^8			0.148532	0.174512	0.148532	
32	10^8			0.131161	0.154057	0.131161	
1	10^9		7.94	8.008706	9.879087	7.94	1.197167
4	10^9			2.337339	2.908955	2.337339	
8	10^9			1.242598	1.547214	1.242598	
10	10^9			1.421526	2.053245	1.421526	
16	10^9			1.197167	1.683573	1.197167	
20	10^9			1.313331	2.066982	1.313331	
30	10^9			1.338987	1.7317	1.338987	
32	10^9			1.285785	1.76449	1.285785	

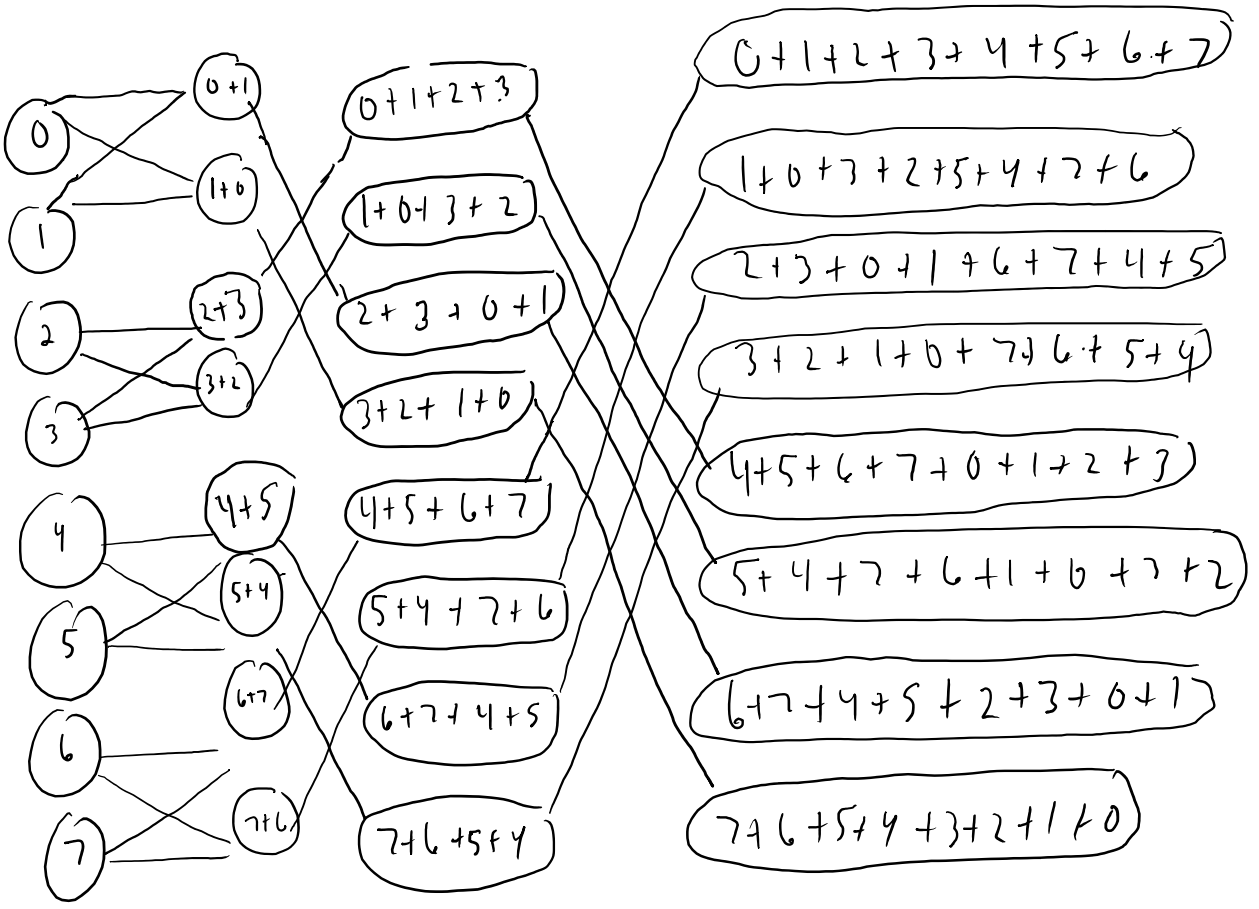
It is clear that the problem was solved best with thread size = 16 and on program 1.A.
Anyone choosing to run the code should run the code on 16 threads.

1. If no processors start with the correct nodes, then the initial super step will require twice the number of communications than it would if each node started on the correct processor.



The most number of communications that will happen is when each processor has to communicate with another like above. The fewest number is when half of the processors communicate with another like in figure 2.3. All / Half = Twice number of communications.

2.

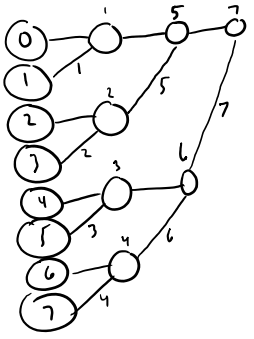


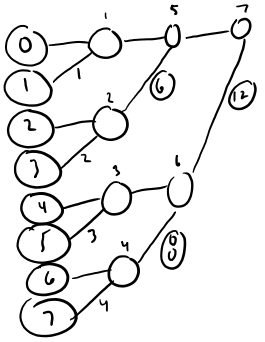
Note: In stages 2 and 3 processors forward data to themselves, but adding the extra lines makes the drawing clunky

3.

Assuming that every processor starts with the correct node, each iteration of the reduction will require logarithmically less communications. This is also true for computations. Execution time of each iteration will be 2*log2(n) where n is the number of computations/communications in the previous iteration of

the reduction
If sending a value from processor p to p+-k requires k time then communication will not be logarithmic it will be linear. Computation time will remain the same and the execution time will be 1/2N + log(n) for each iteration where N is the number of nodes we begin with and n is the number of computations in the previous iteration





Note: each stage requires 1/2N communications in this model

4.
When summing 8 elements on four processors, each processor will take on N/P elements. Each processor will sum these before communicating with any other processors. For this reason in the case of both 8 and 16 elements the number of communications that will be made is 3.
The edges that no longer correspond to communications are the edges that do not cross the "processor lines". These are not communications but just loads in the same processor.