High Performance Computing (CSCI-455/522)

Final Exam (April 20, 2020)

NAME: Darwin J. Groskleg STUDENT NUMBER 200903596

10pts ✓

1. What are the programming models of MPI and OpenMP? Which hardware architectures are most suitable for each model? What are the major advantage of MPI over OpenMP and OpenMP over MPI? (10 points)

13 pts ✓

2. A multiprocessor consists of 20 processors. What are the Gustafson's and Amdahl's estimations, respectively when running a program of which 20% is sequential and 80% is parallelizable? Why do they have two different speedup estimations? Why sometime superlinear speedup can be achieved, please explain two possible reasons? (13 points)

10pts ✓

3. Suppose you are programming on a distributed memory computer where each node is connected to every other node and only communication primitives are **Send** and **Receive.** Barrier synchronization of 4 nodes on this machine takes 1ms, estimate the time for synchronization of 16 nodes, assuming an optimal algorithm. Describe your answer (10 points)

32pts

4. Sort the following sequence by hand using the parallel methods described in the book:

   10 6 14 4 9 3 12 15

   a) mergesort method (8 points)
   b) quicksort method (8 points)
   c) quicksort on hypercube method (8 points)
   d) odd-even mergesort method (8 points)

35pts

5. To solve the n by n large and dense linear equation of systems, the GE method is used:
   a) If we have n processors and each processor has one row of the coefficient matrix, suppose we are working on the hypercube architecture, what is the parallel complexity of the GE algorithm? (8 points)
   b) The typical way to improve the parallel performance is to overlap the communication and computation; can you list some MPI routines to support such strategy? (6 points)
   c) If we have p (p < n) processors and suppose we are working on the architecture that the broadcast message can be done in one single step, are they (row-wise strip partitioning, and row-wise cyclic partitioning) cost-optimal? Which one has better load-balancing? (8 points) Hint: *find the definition of "cost-optimal" in the textbook.*
   d) If we have p (p < n) processors with row-wise strip partitioning, we have several choices to reconfigure network topology as binary tree, line and ring, which one you would choose for better parallel performance, bring your arguments? (13 points)

HPC

1. MPI vs Open MP : Prog models of each.
   - → shared memory devices, multiprocessor
     e.g. threaded for loops
   - → Distributed memory devices / multicomputer
     many processes, no shared mem
     Message Passing Interface

Advantages of one over the other

MPI

- API specification

-Suffers network latency

+

- Easier of 2 to scale software up to the hardware, due to limit on existing number of cores/processors on any one computer. No such limit for networked clusters.
- Thus can handle greater sized problems more quickly.

Open MP

- Language Extension, implemented by the compiler
- Easier to Install
- + Faster single machine perf (possibly)
- Code written in C/C++ that uses OpenMP directives may still compile & execute correctly even on machines that don't support Open MP.
- Greater single-node (machine) performance than MPI where they are not equal

2. np = 20 processors $N = 20$

pgm:
- 20% sequential
- 80% parallelizable

Q Gustafson's Estimations

$S_{latency}(s)$

$S = N + (1-N)s$ ← serializable; N = processors

$20 + (-19)\cdot 0.2 = 20 - 3.8 = 16.2$

$= 20 + (-20)\cdot 0.2 = 23.8$

$= s + np$ ← np = # processors × parallelizable portion %; s = serializable %

$= 0.2 + 0.8 \times 20$

$= \frac{s + np}{s + p} = \frac{0.2 + 20 \times 0.8}{1} = 0.2 + 16 = 16.2$

Speedup

$S_{latency} = 1 - p + sp = 1 - 0.8 + 0.16 = 0.2 + 0.16 = 0.36$

Q Amdahl's Estimation

$S_{latency} = \frac{1}{(1-p) + \frac{p}{N}} = \frac{1}{(0.2) + \frac{0.8}{20}}$

$\times 20.00$ / 7.20

$= \frac{1}{0.2 + 0.04} = \frac{1}{0.24} \times \frac{100}{100} = \frac{100}{24} = \frac{100}{24} \times \frac{1/24}{1/24}$ = 4.16

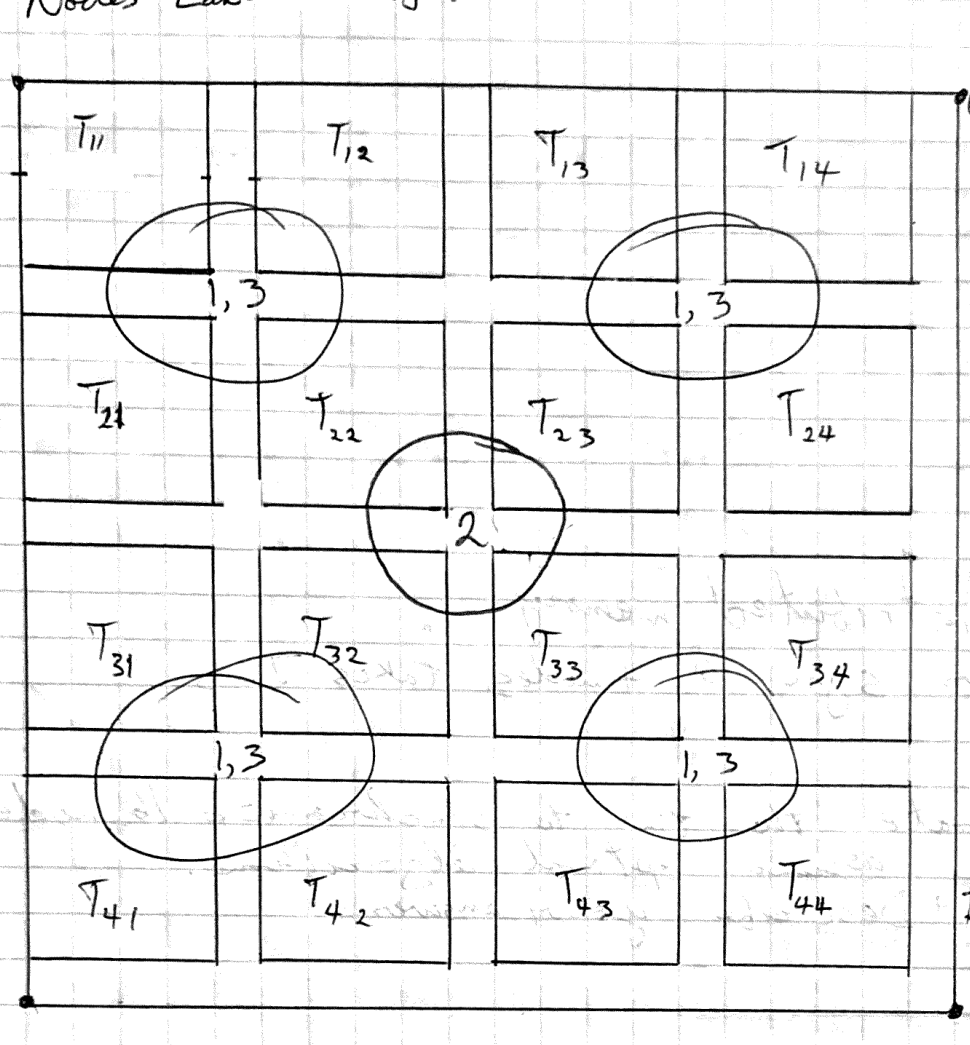
Q. Why different?

1. Caching Threshold (all data fits in cache)
2. Parallelized Tree Traversal.

Q. Why can superlinear speedup happen? (2 reasons)

3. continued

Nodes Labelled T_{ij} ...



By approaching this optimally we can do 4 syncs of 4 nodes concurrently in step 1 & step 3.

∴ Total sync time is 3 ms, one ms for each step.

Much faster than solving for $\frac{4n}{1ms} = \frac{16n}{?ms} \Rightarrow 4ms.$

3. Distributed memory

Barrier sync of 4 nodes takes 1 ms.

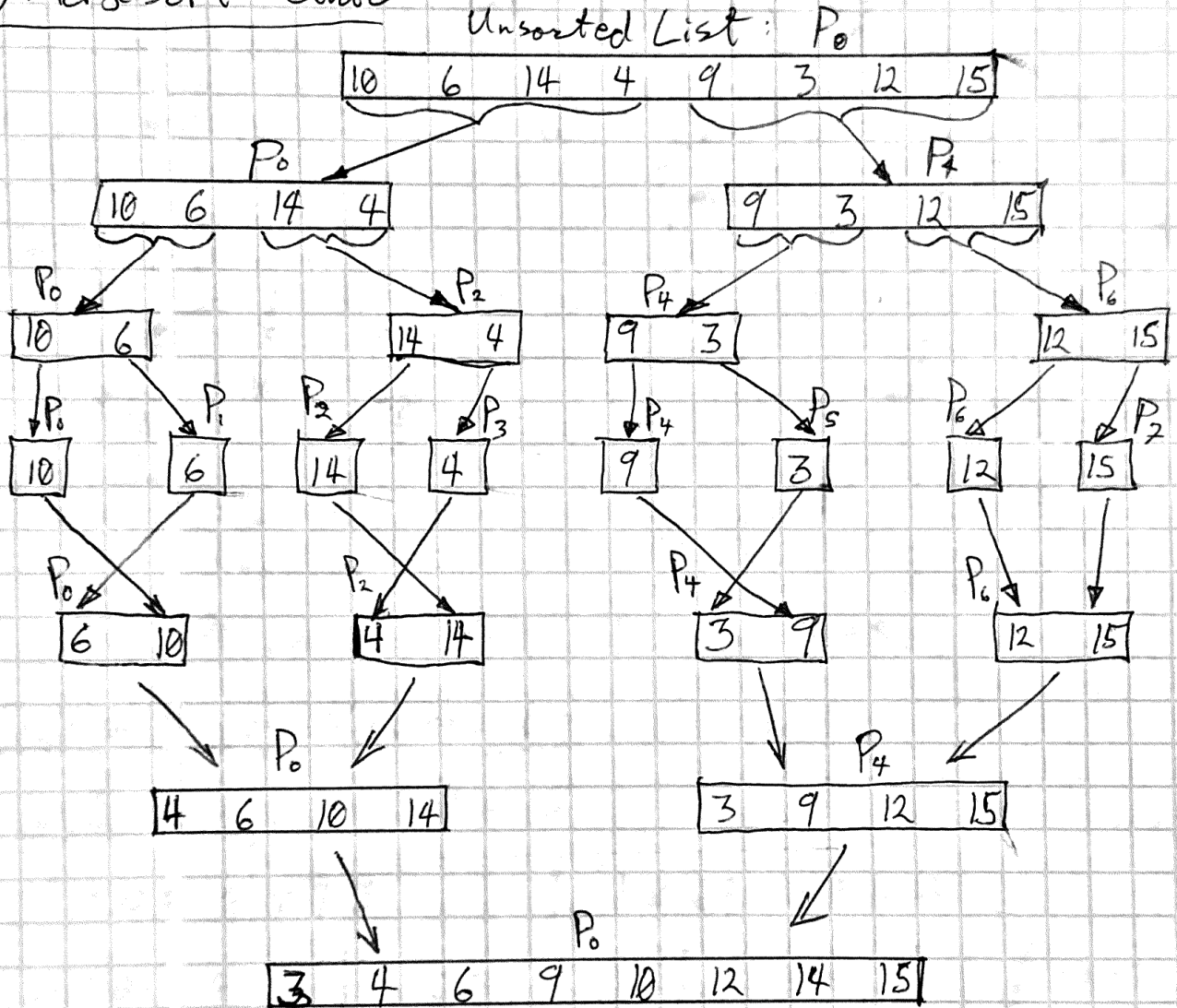
- Estimate the time to synchronize 16 nodes, assume optimal algorithm.
  + Describe your answer

a.

#4.

$A = \{10, 6, 14, 4, 9, 3, 12, 15\}$
8 elements

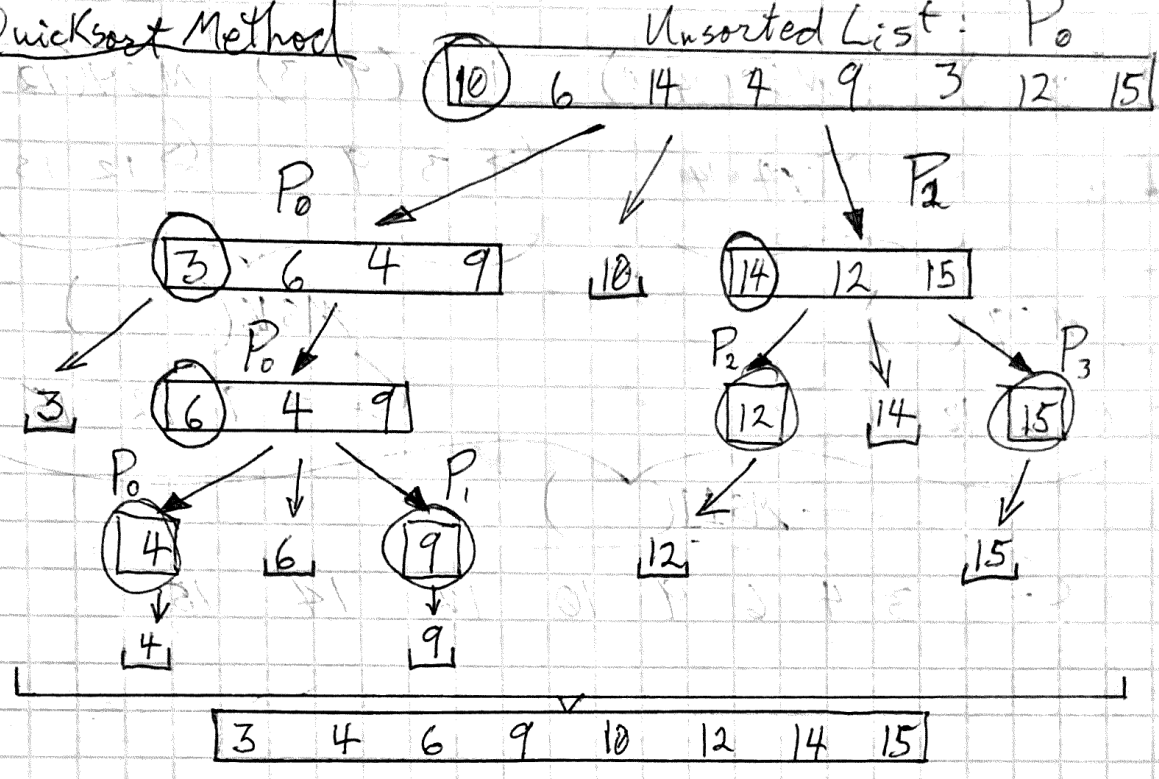
a) Mergesort Method



#4

Pivot is Blue

b) Quicksort Method



Pivots are withheld in processes.

c) Quicksort on Hypercube

000: 10 6 14 4 9 3 12 15
(10 = piv)

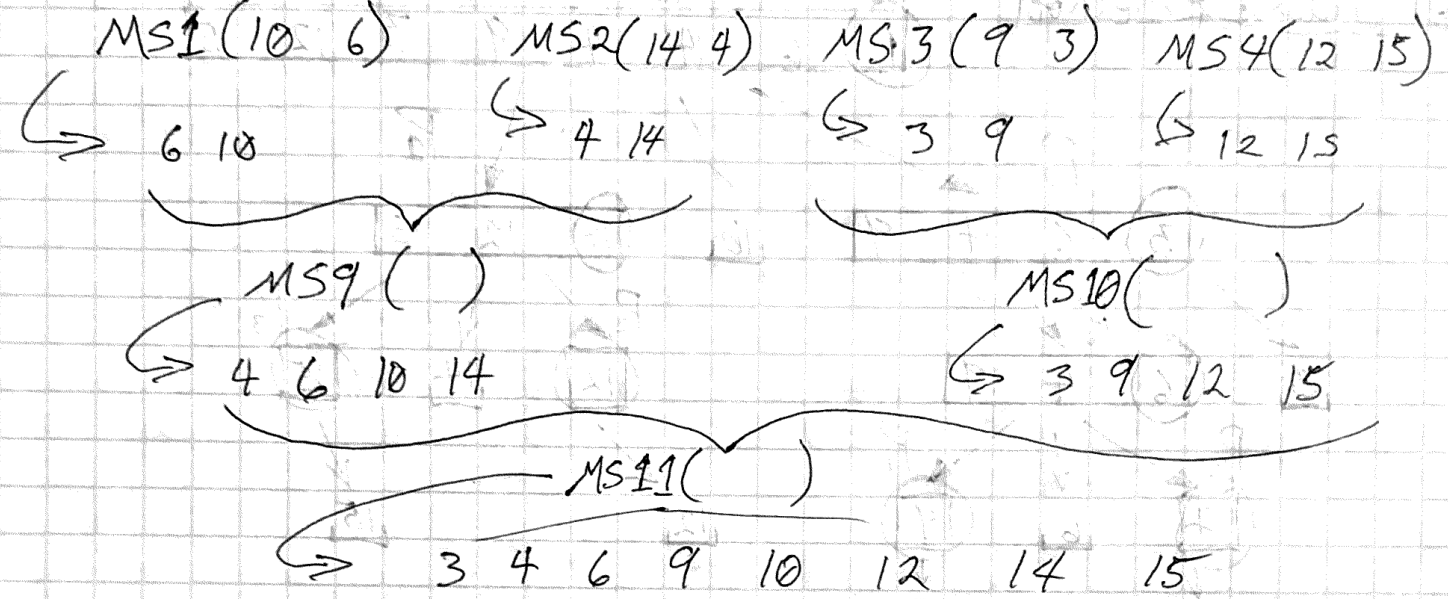
000: 6 4 9 3 | 100: 10 14 12 15

000: 4 3 | 001: 6 9 | 100: 10, 12 | 110: 14 15

000: 3 4 001: 6 9 100: 10, 12 110: 14 15

3 4 6 9 10 12 14 15

4. d) Odd-Even merge Sort



#5. a) n procs, 1 row per proc, n×n matrix

sequential GE: $O(n^3)$

parallel in Hypercube: each message must travel at most $\log_2(n)$ nodes to reach it's destination.

t_{comm} dominates but hypercube allows an improvement to

$= O(n^2 \log n)$

This may be less if bcast requires only 1 steps for all.

#5.

b) improvement: overlap communication & computation.
List MPI routines to ~~do so~~. support a "Latency Hiding" strategy.

- Non-Blocking Send & Receive:
  - MPI_Irecv, MPI_Isend, MPI_Wait

1. A Receiving Task (worker) may request a piece of data before it is needed by calling `MPI_Irecv`.

2. The same task may proceed to some computations that do not depend on the data being received. This makes efficient use of the time until the data is ~~sent~~ then received.

3. When step 2 computations are complete and the data to be received is needed, The receiving Task will call `MPI_Wait`, passing it the request handle that was returned by MPI_Irecv earlier. This guarantees that the data is completely received before proceeding by blocking.

This latency hiding approach ~~changes~~ reduces the execution time from $t_{comm} + t_{comp}$ to $\max(t_{comm}, t_{comp})$

#5. c) · p ($p<n$) procs.

· Bcast is one step

Then $t_{comm} \Rightarrow O(n^2)$

Then $O\left(\frac{n^2}{p}\right)$ for t_{comp} too

$$p \cdot O\left(\frac{n^2}{p}\right) = \cancel{p} \cdot \frac{1}{\cancel{p}} O(n^2)$$

So yes, this is cost-optimal since the p number of processes scales w the problem size.

Cyclic partitioning will have better load balancing.

d) i Row-wise strip, use BinTree, line or Ring

Looking a diameters:

line	n
Ring	$n/2$
BinaryTree	$2(\log_2 n - 1)$

BinaryTree w in strip would be best as internode communication time will not exceed the diameter distance.