CS546 “Parallel and Distributed Processing”
Homework 6

Submission:
Due by 11:59pm of 11/12/2019
Total points 100 - Late penalty: 10% penalty for each day late
Please upload your assignment on Blackboard with the following name:
CS546_SectionNumber_LastName_FirstName_HW6.
Please do NOT email your assignment to the instructor and/or TA!

1. **(10 points)** What is MapReduce and what are its characteristics? List its advantages and disadvantages. Compare it with MPI-based approaches to parallelize a certain task.
2. **(20 points)** Provide pseudocode for sorting integers in an out-of-core fashion (i.e., integers cannot fit in memory) both in MapReduce (both mapper and reducer code) and in MPI. Think about the flow of sorting: reading the input, perform the sorting algorithm, merge all intermediate results, write the final sorted output.
3. **(10 points)** Provide a table of features for Distributed File Systems (HDFS) and Parallel File Systems (PVFS). Discuss briefly similarities and differences.
4. **(10 points)** How does data distribution work in HDFS? Who is responsible for distributing data? What would you optimize in the distribution policies to make the system faster and more reliable?
5. **(10 points)** Discuss how the fault tolerance features in HDFS work? Compare it with hardware based (RAID) approaches. What other approaches for fault tolerance are out there (think erasure coding)?
6. **(10 points)** On a ring, all-to-all broadcast can be implemented in two different ways: (a) the standard ring algorithm as shown in Figure 1 (next page) and (b) the hypercube algorithm as shown in Figure 2.
   a. What is the run time for case (a)?
   b. What is the run time for case (b)?

   If *k* messages have to traverse the same link at the same time, then assume that the effective per-word-transfer time for these messages is kt_w. Also assume that $t_s = 100 \times t_w$.
   a. Which of the two methods, (a) or (b), above is better if the message size *m* is very large?
   b. Which method is better if *m* is very small (may be one word)?

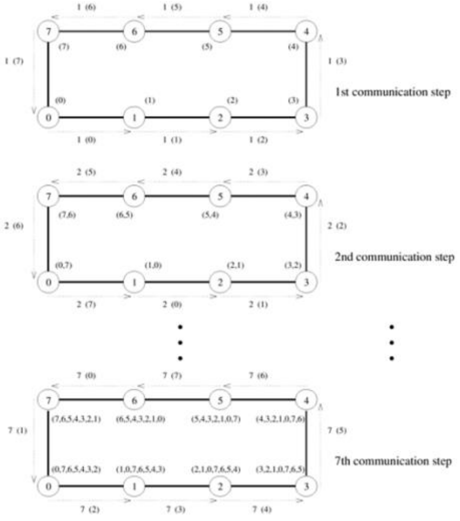
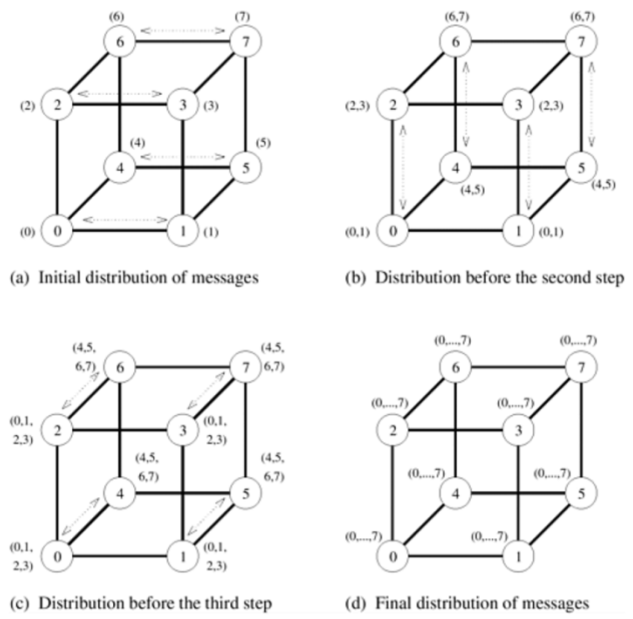


Figure 1

Figure 4.11. All-to-all broadcast on an eight-node hypercube.



(a) Initial distribution of messages

(b) Distribution before the second step

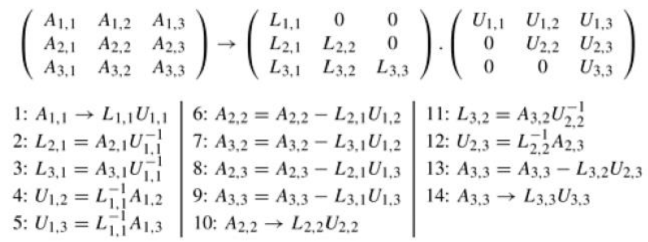
(c) Distribution before the third step

(d) Final distribution of messages

Figure 2

7. **(10 points)** Figure 3 shows the decomposition into 14 tasks of LU factorization of a matrix split into blocks using a 3 x 3 two-dimensional partitioning. If an *m* x *m* partitioning is used, derive an expression for the number of tasks *t*(*m*) as a function of *m* in a decomposition along similar lines.

Figure 3.27. A decomposition of LU factorization into 14 tasks.



$$\begin{pmatrix} A_{1,1} & A_{1,2} & A_{1,3} \\ A_{2,1} & A_{2,2} & A_{2,3} \\ A_{3,1} & A_{3,2} & A_{3,3} \end{pmatrix} \rightarrow \begin{pmatrix} L_{1,1} & 0 & 0 \\ L_{2,1} & L_{2,2} & 0 \\ L_{3,1} & L_{3,2} & L_{3,3} \end{pmatrix} \cdot \begin{pmatrix} U_{1,1} & U_{1,2} & U_{1,3} \\ 0 & U_{2,2} & U_{2,3} \\ 0 & 0 & U_{3,3} \end{pmatrix}$$

1: $A_{1,1} \rightarrow L_{1,1}U_{1,1}$
2: $L_{2,1} = A_{2,1}U_{1,1}^{-1}$
3: $L_{3,1} = A_{3,1}U_{1,1}^{-1}$
4: $U_{1,2} = L_{1,1}^{-1}A_{1,2}$
5: $U_{1,3} = L_{1,1}^{-1}A_{1,3}$
6: $A_{2,2} = A_{2,2} - L_{2,1}U_{1,2}$
7: $A_{3,2} = A_{3,2} - L_{3,1}U_{1,2}$
8: $A_{2,3} = A_{2,3} - L_{2,1}U_{1,3}$
9: $A_{3,3} = A_{3,3} - L_{3,1}U_{1,3}$
10: $A_{2,2} \rightarrow L_{2,2}U_{2,2}$
11: $L_{3,2} = A_{3,2}U_{2,2}^{-1}$
12: $U_{2,3} = L_{2,2}^{-1}A_{2,3}$
13: $A_{3,3} = A_{3,3} - L_{3,2}U_{2,3}$
14: $A_{3,3} \rightarrow L_{3,3}U_{3,3}$

Figure 3

8. **(10 points)** What is the Log(P) model? Why do we need the Log(P) model? Can we use it in memory hierarchy performance modeling?

9. **(10 points)** What is the Memory-Log(P) model? Why do we need the Memory-Log(P) model? Can we use C-AMAT to model the memory part of the Memory-Log(P) model? Provide your explanation or result.

Bonus Questions

1. **(10 points)** If a problem of size *W* has a serial component *Ws*, prove that *W*/*Ws* is an upper bound on its speedup, no matter how many processing elements are used

2. **(10 points)** Why do we use a probability model to descript the performance of a non-dedicated shared environment? Can you list other two methods we can use to descript the performance of a non-dedicated environment? Provide an example to demonstrate/illustrate one of your method.

Note: We encourage collaboration between you and your classmates. Discuss various approaches and techniques to better understand the questions. However, we do NOT allow copying solutions or code. This is considered as cheating and falls under IIT code of honor. Penalties will be enforced. Please make sure you write your own solutions.
GOOD LUCK!