Name (Last, First): _____

This exam consists of 5 questions on 8 pages; be sure you have the entire exam before starting. The point value of each question is indicated at its beginning; the entire exam is worth 100 points. Individual parts of a multi-part question are generally assigned approximately the same point value: exceptions are noted. This exam is open text and notes. However, you may NOT share material with another student during the exam.

Be concise and clearly indicate your answer. Presentation and simplicity of your answers may affect your grade. **If I cannot read your answer easily you will not get credit**. Answer each question in the space following the question. If you find it necessary to continue an answer elsewhere, clearly indicate the location of its continuation and label its continuation with the question number and subpart if appropriate.

You should read through all the questions first, then pace yourself.

The questions begin on the next page.

Problem	Possible	Score
1	20	
2	20	
3	20	
4	20	
5	20	
Total	100	

1. (_____/20 points)

Distributed Sorting

Assume that you have 10 machines connected by a Gigabit Ethernet switch. Any two nodes can communicate at an effective rate of 50 Mb/sec (Megabytes per second). Each node has a large local disk. A node can write to the disk at a rate of 50 Mb/sec and it can read at a rate of 100 Mb/sec. Each node has 10 GB of RAM. Assume that you can sort records in memory at a rate of 50 Mb/sec. Now assume that you have the 100 Gb file on node 1. The 100 Gb file consists of 100 byte records. Note that 1 Gb = 1000 Mb.

For the questions below assume that you can use the entire 10 GB of RAM on each node for sorting. Also assume that you are not overlapping disk I/O with computation or disk I/O with network communication. You can, however, have pairs of nodes communicate with each other simultaneously. You can assume that you can merge two sorted files at the rate of reading from disk as long as the first file is in memory. State any additional assumptions you make.

The questions continue on the next page.

(a) Come up with a scheme for sorting the 100 Gb file on a single node (node 1). Using the parameters given above determine how long it will take to sort the 100 Gb file using your scheme?

(b) Come up with a scheme for sorting the 100 Gb file using all 10 nodes. The 100 Gb file only exists on node 1 initially. Using the parameters above how long will it take to sort the 100 Gb file using your scheme?

2. (_____/20 points)

The Google File System

Recall the paper *The Google File System* by Ghemawat, Gobioff, and Leung. GFS was designed to handle a relatively small number of very large files. Why are small files a problem for GFS? Give specific architectural issues.

More specifically, what does a large number of small files mean for the GFS master node? Proposal a change to the architecture that would allow the GFS master to handle a large number of files. Justify your answer.

3. (_____/20 points)

MapReduce

Recall the paper *MapReduce: Simplified Data Processing on Large Clusters* by Dean and Ghemat. Is it possible for Reduce tasks to begin before all the Map tasks have completed? Explain and justify your answer.

4. (_____/20 points)

Scalable Storage

Recall the Big Table paper *Bigtable: A Distributed Storage System for Structured Data* and the Dynamo paper *Dynamo: Amazon's Highly Available Key-value Store*. Explain at least 2 significant similarities between these two systems. Also explain at least 2 significant differences between these two systems.

5. (_____/20 points)

Hadoop MapReduce

Recall the Map task for WordCount:

```
public static class TokenCounterMapper
extends Mapper<Object, Text, Text, IntWritable> {

private final static IntWritable one = new IntWritable(1);
private Text word = new Text();

public void map(Object key, Text value, Context context)
    throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
    while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
    }
}
```

Explain how to optimize this code by detecting repeated words on the same line. You can provide pseudo code, but explain the modifications to the above code snippet.

Continue your answers here if necessary.