1) **Replication and Consistency:**

a) A design issue in content replication is whether updates to data items are pulled or pushed by or to the clients from servers. As discussed in the class, the former strategy is commonly known as *pull-based* content replication, while the latter one is denoted as *push-based* content replication.

1) Discuss one main advantage of pull-based versus push-based replications?
2) Discuss one main disadvantage of pull-based versus push-based replications?
3) Devise a scenario where neither push-based nor pull-based replication will work well by itself.
4) Suggest an algorithm that combines both strategies (or something totally different) so as to effectively tackle your presented scenario.
5) When using your algorithm, is it necessary that the clocks of a client and the server are tightly synchronized? Explain.
6) Would your algorithm necessarily lead to better performance? Explain.

b) Explain which strategy scales better, totally-ordered multicasting using Lamport’s logical clocks or the sequencer approach discussed in the class under active replication.

c) An operation $OP1$ upon an object $B1$ causes $B1$ to invoke an operation $OP2$ upon another object $B2$. If $B1$ is replicated but not $B2$, what problem(s) will emerge? Suggest a solution for such (a) problem(s).

d) Assume two applications $A1$ and $A2$ and two clusters $C1$ and $C2$. Suppose that $A1$ and $A2$ have high and low read-write ratios, respectively. Suppose that $C1$ is characterized with scarce network bandwidth while $C2$ exhibits exactly the
opposite. For two types of consistency, sequential and eventual, which application will you deploy at which cluster? Justify your answer.

II) Amdahl’s Law:

Use Amdahl’s law to solve the following questions:

1) Suppose a parallel program $P'$ includes a method $M$ that cannot be parallelized. Besides, assume that $M$ accounts for 10% of the original sequential program $P$. What are the maximum speed-ups of $P'$ on 10 processors and 1000 processors?

2) Suppose that you want to achieve a speedup of 20 with 32 processors. What fraction of the original computation in $P$ should be sequential?

3) Suppose that $M$ accounts for 0.5% of $P$ rather than 10%.
   - What will be the maximum speed-up ratio on an unlimited number of processors?
   - What observation can you attain regarding obtaining a high degree of scalability in distributed systems? Explain whether this is true in reality.

III) Analytics Engines:

a) Read the paper titled “MapReduce: Simplified Data Processing on Large Clusters” by Jeffrey Dean and Sanjay Ghemawat and answer the following questions:
   1) Summarize the paper in 2 paragraphs.
   2) What kind of constraints does MapReduce place on its problem domain? Said in another way, what applications do you think would not work (well) using MapReduce?
   3) Given the following characteristics of a distributed system: (i) fault-tolerance, (ii) scalability, and (iii) effective scheduling. Write few sentences describing how MapReduce addresses each of these characteristics.

b) Write critiques for the research papers “Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud” by Y. Low et al. and “Pregel: A System for Large-Scale Graph Processing” by G. Malewicz et al. Refer to the course webpage on how to write a critique for a research article.