Stanford University
Computer Science Department
CS 240 Final

May 31, 2012

!!!!!! SKIP 20 POINTS WORTH OF QUESTIONS. !!!!!

This is an open book exam. You have 75 minutes. Cross out the questions you skip. Write all of your answers directly on the paper. Make your answers as concise as possible. Sentence fragments ok.

NOTE: We will take off points if a correct answer also includes incorrect or irrelevant information. (I.e., don't put in everything you know in hopes of saying the correct buzzword.)

Question	Score
1-6 (30 points)	
7-12 (30 points)	
13-17 (30 points)	
total (max: 70 points):	

Stanford University Honor Code

In accordance with both the letter and the spirit of the Honor Code, I did not cheat on this exam nor will I assist someone else cheating.

Name:

Signature:

I Short answer

1. [5 points]: You create a patch, there are no incoming edges on the block the patch applies to: what does featherstitch do? What intuition is it exploiting?

2. [5 points]: Draw the shortest multi-block patch dependency that featherstich will reject. Explain how to write this data out correctly, despite featherstitch's rejection (be very concrete!).

3. [5 points]: The livelock paper states that you can solve livelock by doing almost everything at high IPL or do almost nothing at high IPL. What is the intuition behind this statement? For which of the first four livelock experiments is this not true?

4. [5 points]: Your system has two kernel threads A and B with a message queue Q between them. There is also a user level process C. Assume the system is under heavy load and does not use the techniques from the livelock paper: give two bad things you would expect to see. In *at most 40 words* give the give the **complete** livelock solution for this system (ignore quotas and grammar).

5. [5 points]: Your old 140 partner JimBo says that if you run an NFS client and server on the same machine that you can eliminate the biggest requirement in the NFS protocol that limits performance. What the hell is he talking about? Is he correct?

6. [5 points]: Give two concrete ways you could modify the NFS paper's implementation to use leases to *clearly* improve consistency or performance.

7. [5 points]: Sketch how to modify an NFS client to make data/metadata writes faster using ideas from xsyncfs. (Assume the original NFS client waits for each NFS server response before continuing.) If you reran the experiments in Figure 4 and Figure 5 in the xsyncfs paper using the original NFS client and your modified version, how would you expect things to play out?

8. [5 points]: You compare the system in the Lease's paper to two others: one with infinitely fast CPUs and one with an infinitely fast network. Rank these three systems in terms of the relative benefit of leases for each and give the intuition behind your ordering.

9. [5 points]: LBFS re-checks its hashes on each lookup. Assuming that no local file system can access an LBFS file system: explain how to change the LBFS client and server so that they can trust the hashes in their database. Be very specific about how to write data during normal operation and what to do after crashes (if anything).

10. [5 points]: You run the following code on LBFS:

```
fp = open("foo");
             <----- Receive invalidation message.
read(fp, buf, ...);
close(fp);

...  some other code ...
```

Assume you receive a lease invalidation after the `open` but before the `read`. At what point would the client fetch the more recent copy and how would it know to do so? (Hint: think about the type of consistency LBFS uses.)

11. [5 points]: If the Map/Reduce system was modified so that map tasks wrote their output to the Google GFS, instead of the local file system. Would this change ever improve performance? Would the lines in figure 2 and Figure 3(a) go up, down, or remain the same? Please justify your answer.

12. [5 points]: Map/reduce: "We rely on atomic commits of map and reduce task outputs..." What do they mean by atomic? How do they make map and reduce output commits atomic?

13. [5 points]: Cycles, cells and platters: Give three *actionable* results from the paper (e.g., results that you or google could act on).

14. [5 points]: Cycles, cells and platters: in their experiments the authors significantly prune the population of reports they consider in an attempt to eliminate known biases. Give two specific examples where they do so, and the problems that could have occured if they had not.

15. [5 points]: Rocket science benchmarking: give a concrete *useful* experiment you could add to one of the storage papers covered that would address the concerns in the rocket science paper. Please justify your answer.

16. [5 points]: Disks melt snowflakes, Figure 2: What would it mean if you could only see one color band in (c), but it was thick? What would it mean if there was three bands, but they were as thin as (a)?

II Real is better than virtual

17. [10 points]: To improve their Linux benchmark numbers VMware hires you to build xsyncfs into ESX. Sketch how this would work. You can assume that ESX has control over all devices, but doesn't have much visibility into the guest OS.

Explain any tricky problems that arise for tracking process dependencies that might force you to do things slightly differently than in the xsyncfs paper.