Midterm practice questions
Fall 2000 15-712

The midterm is open book, open notes, and will be held in class. We suggest that you prepare for the exam by answering these review questions, reading your classmates' paper reviews posted to the course webpage, studying the lecture notes and any of your own notes that you have taken, and skimming the assigned papers.

In the actual exam, you should write brief, precise answers. As opposed to what we said earlier, we are not looking for essays. Rambling brain-dump answers may be punished twice — you run out of time writing them and we run out of patience reading them. :–)

1. Soft Updates and Log-structured file systems both try to maximize the value of write-back caching. Give an example of what circumstances would cause each approach to outperform the other (assume soft updates is used with BSD FFS as described in the paper we read).

2. Delaying changes to a set of files is a classic performance improvement scheme for distributed filesystems. Describe the callback-based scheme for ensuring cache consistency in distributed file systems.

3. NASD devices export an "object"-level interface (instead of block-level). What are the advantages and disadvantages to this approach?

4. Explain what NASD and xFS mean by "scalability" and how does each achieve it (in general terms).

5. Consider two types of callback semantics: (1) When a cached block is overwritten, the block's callbacks are broken before the writer is allowed to commit its changes; (2) the writer is allowed to commit its changes immediately and callback breaking can occur asynchronously. For these two alternative semantics, is it more natural to implement callback breaking with message passing or RPCs? For this question, assume machines won't fail and there are no network partitions.

6. River and Abacus both try to spread work across nodes to improve performance. Explain how their foci differ and the resulting effects on their designs.

7. Emerald presents a call-by-reference interface to the programmer, but for performance reasons, calls may actually be translated into call-by-move (where the object and possibly objects that the object references, ..., are moved to the remote node). Discuss scenarios under which the Emerald system should do call-by-reference and call-by-move.

8. Abacus currently assumes that an object's behavior is independent of where it is running. Assuming that correctness is not impacted, what goes wrong if this assumption is false?

9. SPIN and exokernel both promote particular organizations for providing safe and efficient OS extensibility. Explain why this can be useful. Discuss what is needed, beyond such structures, to achieve such benefits in practice.

10. The "virtual file system" (i.e., vnodes) abstraction defines a specific interface for new FSs to be added. In most systems, this is done by staticly or dynamically linking object code into the OS kernel. Explain why this is dangerous and one low-overhead alternative.