1. Soft Updates and Journaling [9 points]

(a) There are certain invariants that must be true to ensure that a file system will always be in a consistent state.
   * Name at least 2 invariants that FFS respects
   * Provide at least one thing that a reasonable person might like but is not guaranteed by FFS

(b) Louis Reasoner has a workload that consists of creating a bunch of files in the same directory. Remembering that Journaling systems may perform additional I/O to maintain ordering information, Louis believes that FFS will perform fewer meta-data writes than Journaling. Is Louis Reasoner correct? Explain

(c) When evaluating Soft Updates, the authors discovered that its performance sometimes decreased when the size of the benchmark increased. That is, when the benchmarks operated on either larger data sets or for longer periods of time, the relative throughput of Soft Updates (as compared to FFS) often decreased considerably when compared to smaller data sets over shorter periods of time.
   * List two causes of this effect.
   * Soft Updates sometimes out-performs FFS-Async. Why is this surprising? Why is it happening?

2. Rio [9 points]

(a) For reliability purposes, FFS writes metadata in a certain order. Does RIO also require metadata writes to be ordered for reliability? Explain.

(b) The authors of RIO injected their own bugs instead of using real ones. Why? What do they gain from this? Lose?

(c) Despite the fact that Rio marks the buffer cache as read only, the authors observe that it is still possible for the kernel to make wild writes to the buffer cache. Explain how

3. Caching and Prefetching [9 points]

(a) Describe a scenario (time access fetch cache contents like figure 1) where the kernel can tell if an application is acting foolish (one that uses a policy worse than the kernel's default policy, in this case LRU). Explain how it can tell.

(b) Does aggressive prefetching policy from Fig 1 of the paper adhere to the controlled-aggressive prefetching policy described in the paper?

(c) The LRU-list of a kernel is depicted below, where the frames Ai belong to process A, and the frames Bi belong to process B. The left end of the list corresponds to the item that has been the least recently used.

   \[ A1 \rightarrow B1 \rightarrow A2 \rightarrow B2 \]

   Suppose process A chooses to use its own page replacement policy of most recently used (MRU) while B uses the system's default policy of LRU.
Illustrate the state of the kernel's LRU-list with placeholders using the LRU-SP policy mentioned in the paper at the end of each of the following page references for the page reference string A2, B3, A3, A1, A4, B3.

4. GMS [9 points]

(a) Louis Reasoner has a machine that runs an important application that requires most of the CPU at all times. His application rarely touches memory tough so he believes that he can help out the community by inserting his machine into a cluster running GMS without hurting the application running on his machine. Is Louis Reasoner right? Explain

(b) Ben Bitdiddle is concerned for the well-being of his application's pages. He fears that since his pages can be anywhere in the cluster, he might lose the page's data if a node carrying it crashes or leaves the cluster. Should Ben Bitdiddle be worried?

(c) The authors of GMS provide many microbenchmarks to show the ease and benefits of using GMS in a cluster.
   - In table 3, explain why there is a large difference in GMS and no GMS for random access.
   - In table 4 why is GMS duplicate lower than GMS single?