

Please be sure to show all your work and write down any assumptions you make.

1. One of the earliest “read-modify-write” atomic instructions is **compare-and-swap (CAS)**, introduced by the IBM 370 architecture, and also provided in the 680x0 and SPARC V9 instruction sets. CAS takes three operands: the location to be modified, a value that the location is expected to contain, and a new value to be placed there if (and only if) the expected value is found. The instruction returns an indication of whether it succeeded. Show that CAS is *universal*, in the sense that it can be used to emulate any other read-modify-write instruction that atomically reads a memory location, computes a new value as a function of the old one, and writes the new value back.
2. Consider a snoopy bus-based protocol similar to the MSI protocol we reviewed in class. Suppose you were designing a coherence protocol that favored a migratory access pattern, in the sense that it was designed to minimize bus activity when the access pattern consists of a series of reads and writes by each of several processors in succession. Draw the finite state machine for such a protocol, assuming that you were concerned only with the stable states.
3. The SunFire directory-based protocol uses 3 states per group of processors in the machine. Discuss the scalability issues with this directory scheme. If you were to flatten the machine (i.e., eliminate the snoopy domain and place all processors in the directory-based protocol domain), how would you organize the directory (Describe one directory scheme that would reduce the space requirements (I'd like you to go beyond what is called a *DirN* protocol, where a bit per processor is used to identify the sharers).)? What are the pros and cons of your scheme?
4. Devise an example of competing accesses (preferably something you might expect to see in a real program) that works on one of sync (the SunFire machine) or discovery (the Regatta machine) because of their differing consistency models but not on the other. Test the example on the machines, list the code, and explain why the results are different on the two machines.