Problem 1 (8.3):

512 processors / 8 processors/node = 64 total nodes
64 nodes * 1GB or memory/node = 64GB total memory
64 GB / 64 B cache line size = 1073741824 (1G) total cache blocks

a) Each cache block sized piece of memory needs 64 bits for a bit vector, one bit for each of the 64 nodes.
   1G * 64 bits = 64G bits = 8 GB of overhead. That's 12.5% overhead.

b) Each cache block sized piece of memory needs 3 pointers of log 64 = 6 bits each. So each block needs 18 bits of overhead.
   1G * 18 bits = 18G bits = 2.25 GB overhead. That's around 3.5% overhead.

Problem 2 (8.4):

States shown are the state that the block was found in when the directory was checked originally. Also, I assume that the block is owned, either in E or M, by a remote node, just as the authors do in Figure 8.12.
Problem 3 (8.8):

The optimizations that I would consider are:
1. Having the nodes in the sharing list pass the invalidation request down the list to the last node and then for each node to invalidate its copy of the block and acknowledge the previous node. Then it would only take one trip down and up the sharing list to do all of the invalidations.
2. Do a global invalidate if the sharing list passed a certain threshold in length.

Problem 4 (4.18):

a) Page X: Place in both M0 and M1.
   Page Y: Place in M1.
   Page Z: Place in M0.

b) Page X: Replicate to M1.
   Page Z: Do Nothing.

c) Page X: Replicate to M0. Cost without replication = 58. Cost with replication = 35.
   Page Z: Do Nothing. Cost = 51

   Page Z: Do Nothing. Cost = 51