10 Cache [50 points]

Consider a processor using a 4-block LRU-based L1 data cache with a block size of 1 byte. Starting with an empty cache, an application accesses three cache blocks with the following addresses in the order given below:

$$0 \rightarrow 2 \rightarrow 4$$

A malicious programmer tries to reverse-engineer the number of sets and ways in the L1 data cache by issuing *only* two more accesses and observing the cache hit rate across these two accesses. Assume that the programmer can insert the malicious accesses only after the above three accesses of the application.

(a) [20 points] What are the addresses of the next two cache blocks that should be accessed to successfully reverse-engineer the number of sets and ways in the cache? There may be multiple solutions; ***please give the lowest possible addresses that can enable the identification of the number of sets and ways***. Please explain every step in detail to get full points.

0 → 2

Explanation. There are two possible answers:

- **[0 → 2]**
- **[0 → 4]**

There are three possible set/way configurations, shown below labeled by their respective sets/ways. Each configuration shows a drawing of the cache state after the three initial accesses. Rows and columns represent sets and ways, respectively, and the LRU address is shown for each occupied set:

(a) **(4 sets, 1 way)**

4
-
2
-

(b) **(2 sets, 2 ways)**

4	2
-	-

(c) **(1 set, 4 ways)**

0	2	4	-

At this point, all three configurations have a 100% miss rate since they started cold. In order to differentiate between the three configurations with *just two* more accesses, we need to induce *different* hit/miss counts in each of them. The only way this is possible is if one configuration experiences two hits, another two misses, and the last one hit and one miss.

Only two solutions exist to produce this case:

- **[0 → 2]**
  (a) 0 miss, 2 hit = 50% miss rate
  (b) 0 miss, 2 miss = 100% miss rate
  (c) 0 hit, 2 hit = 0% miss rate
- **[0 → 4]**
  (a) 0 miss, 4 miss = 100% miss rate
  (b) 0 miss, 4 hit = 50% miss rate
  (c) 0 hit, 4 hit = 0% miss rate

Choosing the lowest possible addresses, the correct solution is **0 → 2**

(b) [15 points] What is the number of sets and ways if the cache hit rate observed over the two extra addresses accessed in Part (1) were:

L1 hit rate	# sets	# ways
100%		
50%		
0%		

Explain your reasoning:

Based on the solution to Part (1), these are the number of sets and ways corresponding to different hit rates.

Solution:

L1 hit rate	# sets	# ways
100%	1	4
50%	4	1
0%	2	2

(c) [15 points] Is it possible to reverse-engineer the number of sets and ways of the cache using two accesses (after the application's first three accesses) if the Most Recently Used (MRU) block is replaced first? Explain your reasoning.

No. There is no solution for just two more accesses because with an MRU policy, no permutation of two more accesses is able to assign a unique L1 hit rate to each of the three cache configurations.