Cmpe344 Fall 2021 FF67
Experiment #6: Analysis of Different Cache Configurations

We will again be using the SMPCache simulator in this lab. Answer the questions according to the following configurations, unless stated otherwise in the question:

Multiprocessor

Leave all with their defaults.

Main Memory

Word Size: 8 bytes **Block Size:** 512 bytes **Memory Size:** 1 MB

Cache

Cache Levels: 1 **Cache Size:** 8 KB **Mapping:** Set Associative
Number of Cache Sets: Varies across questions
Replacement Policy: Varies across questions

Use the following memory and cache access times whenever necessary:

Memory Access Time (t_m) = 40 ns

Number of Cache Sets	2	4	16
Cache Access Time (t_c)	0.5 ns	1 ns	4 ns

Reminder on how to calculate…

$$(\text{average access time}) = (\text{hit ratio}) \times t_c + (\text{miss ratio}) \times (t_c + t_m)$$

Questions

▶ *Use the trace "/SAMPLES/TRACES/NASA7.prg" and set the number of cache sets to 4 to answer the questions 1–4.*

1) Calculate the following hit ratios:

   a) ___92.507___% for FIFO replacement policy.

   b) __92.776____% for LRU replacement policy.

   c) ___92.237___% for LFU replacement policy.

2) Which replacement policy is the best in terms of **hit ratios** given this trace? Briefly explain why.

   We get the highest hit ratio with LRU. It shows that, in general least recently used blocks can be preferred to be replaced when there are conflict misses. It can be concluded that this trace doesn't frequently go back to use the blocks that were frequently used a while ago. However, if LFU had better hit ratio that would indicate that frequently used blocks (once upon a time) may be referred again by the program although they are not being used recently for a while.

3) Would the ordering between the replacement policies be different if we used another trace? Briefly explain why.

   Yes, as stated above for different traces ordering between the replacement policies may differ. For example, for programs that refers to a specific block more than once with an amount of time spent in between. In this case, although a block is least recently used, it may be frequently used once upon a time and program may refer to that in the future again.

4) What is the **average access time** for each of the replacement policies in ns?

   Average access time for FIFO : (92.507 * 1 ns + 7.493 * 41)/100 = 3.9972 ns

   Average access time for LRU: (92.776 * 1 ns + 7.224 * 41)/100 = 3.8896 ns

Average access time for LFU: (92.237 * 1 ns + 7.763 * 41)/100 = 4.1052 ns

▶ *Use the trace "/SAMPLES/TRACES/NASA7.prg" and set the replacement policy to FIFO to answer the questions 5–8.*

5) Calculate the following hit ratios:

   a) ___92.507___% with 2 sets in the cache.

   b) ___92.507___% with 4 sets in the cache.

6) Which cache configuration is better in terms of **hit ratios** given this trace? Briefly explain why.

They are the same. Cache size might be bigger than the required size, but this is not the case when we tried the fully associative one. So probably the access pattern for these two cases are similar.

7) Would the ordering of the cache configurations be different if we used another trace? Briefly explain why.

   Ordering of the cache configurations may differ from trace to trace. Results of the ordering depicts the characteristics of the traces. For a trace with different range of instruction and data, efficiency of 2-set and 4-set cases may differ.

8) Which cache configuration is better in terms of the **average access time** in ns? Show your work.

   Average access time for 2 sets : (92.507 * 0.5 ns + 7.493 * 40.5)/100 = 3.4972 ns
   Average access time for 4 sets : (92.507 * 1 ns + 7.493 * 41)/100 = 3.9972 ns

9) We want to analyze the effect of the block size on the hit ratio performance (keeping the overall cache size constant), using the following configuration:

   Trace: HYDRO.prg **Word Size:** 16 bits **Memory Size:** 1 MB
   Cache Size: 8 KB **Mapping:** Fully Associative **Replacement Policy:** LRU

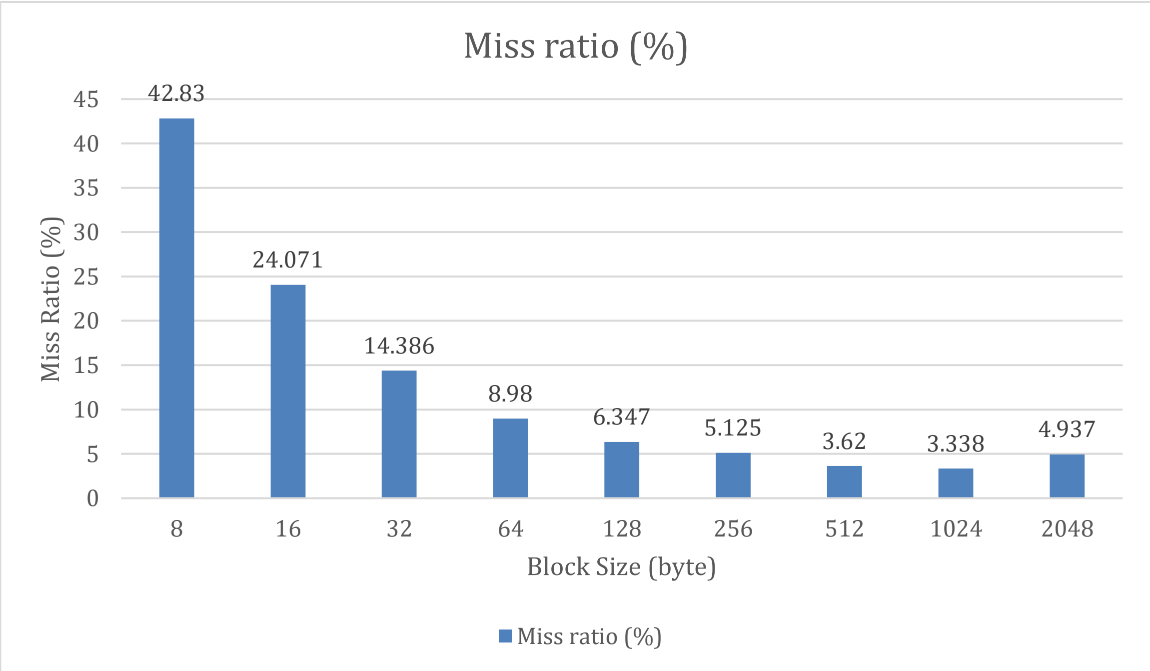
   Report the hit ratios for each possible block size from 8 bytes to 2048 bytes numerically in the table below. Then, plot those results in a graph with block sizes on the x axis and the miss ratios on the y axis. Let the x axis be in logarithmic scale (so that every adjacent setting is equidistant).

WARNING: You will have to change the number of blocks parameter every time you change the block size in order to keep the overall cache size constant.

Numeric Report

Block size	Hit rate
8 bytes	57.170
16 bytes	75.929
32 bytes	85.614
64 bytes	91.020
128 bytes	93.653
256 bytes	94.875
512 bytes	96.380
1024 bytes	96.662
2048 bytes	95.063

Plot Report



10) Explain the trend in the graph you drew in question (9).

As the block size increases, utilization of spatial locality (retrieving the contiguous words in the blocks to the cache) is maximized. Therefore, increasing the blocks results in decrease in the miss rates. However, after a point number of blocks that can be stored in the cache becomes extremely smaller and it leads to a race between the blocks to be stored. Consequently, a block will be extracted from the cache before it is used again. This leads to a decrease in the hit ratio as shown in the table for the case of 2048-byte block size.

11) State one change in the overall configuration that is likely to change the trend in the graph you drew in question (9). You may not change the axes of the graph, i.e. the graph should be about the hit rates against the block sizes 8 bytes thru 2048 bytes. Explain why the change you propose is likely to work.

Decreasing the cache size may lead the trend change its direction earlier than this version. Because competition between the blocks will start from an early point, since number of blocks stored in the cache is lower compared to this version.