University of California, Riverside
Department of Computer Science & Engineering

Title: Advanced Computer Architecture Lab 3

Student Name:
Mahbod Afarin

Student ID:
862186340

Winter 2020

I used one of my slip days for this lab and 2 of my slip days for lab2.

Part 1: How to run the program

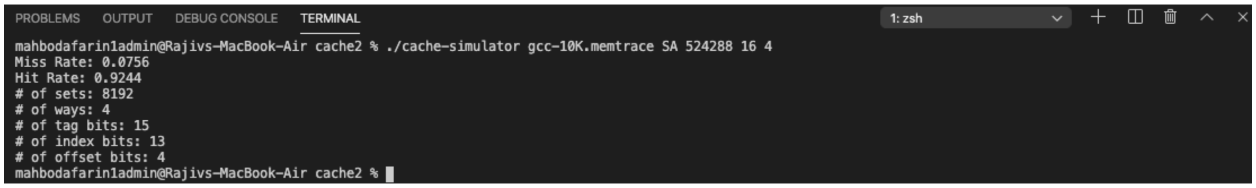
This program simulates direct mapped, set associative, and fully associative caches. For running the program, we can use the following command:

```
./cache-simulator <input> <cache type> <cache size> <block size> <#of ways>
```

Which cache type for direct mapped, set associative, and fully associative is DM, SA, and FA respectively. For example, for simulating a 512KB 4-way set associative cache with 16B block size we should use the below command.

```
./cache-simulator gcc-10K.memtrace SA 524288 16 4
```

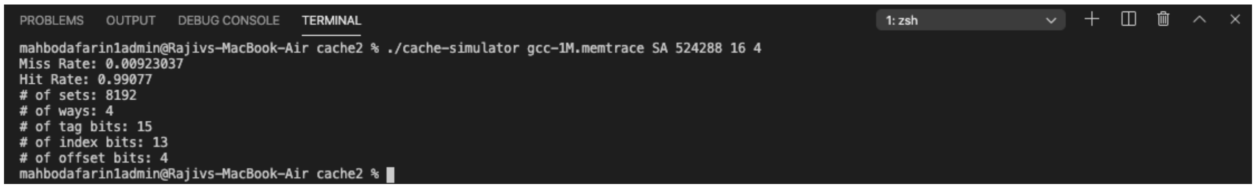
We can see miss rate, hit rate, number of sets, number of ways, number of tag bits, number of index bits, and number of offset bits in the output. For example, the output of the program for 512KB 4-way set associative cache with 16B block size for gcc-10K is in the following figure.



```
mahbodafarin1admin@Rajivs-MacBook-Air cache2 % ./cache-simulator gcc-10K.memtrace SA 524288 16 4
Miss Rate: 0.0756
Hit Rate: 0.9244
# of sets: 8192
# of ways: 4
# of tag bits: 15
# of index bits: 13
# of offset bits: 4
mahbodafarin1admin@Rajivs-MacBook-Air cache2 %
```

Figure 1: Output of the cache program for gcc-10K – Set Associative

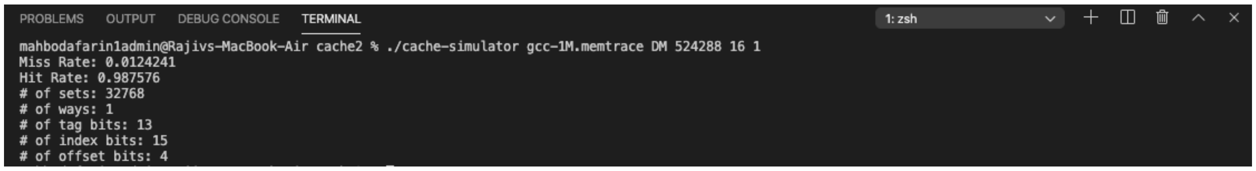
And the output of the same configuration for gcc-1M is in the following figure.



```
mahbodafarin1admin@Rajivs-MacBook-Air cache2 % ./cache-simulator gcc-1M.memtrace SA 524288 16 4
Miss Rate: 0.00923037
Hit Rate: 0.99077
# of sets: 8192
# of ways: 4
# of tag bits: 15
# of index bits: 13
# of offset bits: 4
mahbodafarin1admin@Rajivs-MacBook-Air cache2 %
```

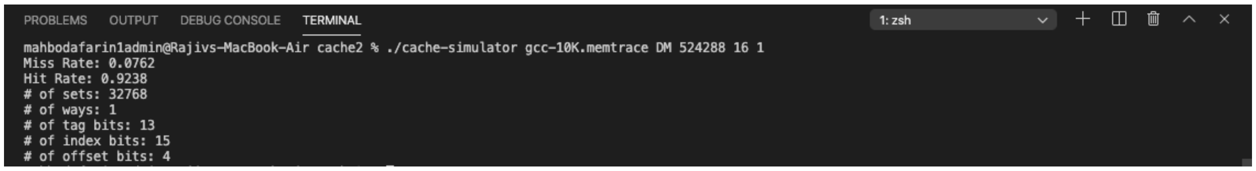
Figure 2: Output of the cache program for gcc-1M - Set Associative

In the following figurers we can see the results for the same configuration for direct mapped cache.



```
mahbodafarin1admin@Rajivs-MacBook-Air cache2 % ./cache-simulator gcc-1M.memtrace DM 524288 16 1
Miss Rate: 0.0124241
Hit Rate: 0.987576
# of sets: 32768
# of ways: 1
# of tag bits: 13
# of index bits: 15
# of offset bits: 4
```

Figure 3: Output of the cache program for gcc-10K – Direct Mapped



```
mahbodafarin1admin@Rajivs-MacBook-Air cache2 % ./cache-simulator gcc-10K.memtrace DM 524288 16 1
Miss Rate: 0.0762
Hit Rate: 0.9238
# of sets: 32768
# of ways: 1
# of tag bits: 13
# of index bits: 15
# of offset bits: 4
```

Figure 4: Output of the cache program for gcc-1M – Direct Mapped

Part 2: Questions

1- Assuming a 512KB 4-way set associative cache with 16B block size, how many bits does the tag have? What is the total size, in bytes, of the cache including tag bits?

Answer:

$b\ (block\ size\ in\ byte) = 16B$
$n\ (number\ of\ ways) = 4$
$m\ (size\ of\ the\ cache\ in\ blocks) = \frac{512KB}{16B} = 32K$
$s\ (number\ of\ sets) = \frac{m}{n} = \frac{32K}{4} = 8K$
$offset = \log_2 b = \log_2 16 = 4$
$Index = \log_2 s = \log_2 2^{13} = 13$
$Tag = 32 - offset - index = 32 - 4 - 13 = 15$
$Cache\ Size = s \times n \times b = 8K \times 4 \times (16 \times 8 + 15 + 1) = 4718592 bits = 576KBytes$

2- What is the cache miss rate of the given traces and cache configuration? Assume we have a 512KB cache and 16B block size.

Answer:

Trace	Direct	2-way	4-way	Fully Associative
gcc-10K	0.0762	0.0756	0.0756	0.0756
gcc-1M	0.012424078	0.0093852447	0.0092303748	0.0092316762

3- For the following configurations, how many bits are for tag, index, and offset fields? Assume we have a 256KB cache and 16B block size.

Answer:

Trace	Direct	2-way	4-way	Fully Associative
gcc-10K	$offset = 4\ bits$ $index = 14\ bits$ $tag = 14\ bits$	$offset = 4\ bits$ $index = 13\ bits$ $tag = 15\ bits$	$offset = 4\ bits$ $index = 12\ bits$ $tag = 16\ bits$	$offset = 4\ bits$ $index = 0\ bits$ $tag = 28\ bits$
gcc-1M	$offset = 4\ bits$ $index = 14\ bits$ $tag = 14\ bits$	$offset = 4\ bits$ $index = 13\ bits$ $tag = 15\ bits$	$offset = 4\ bits$ $index = 12\ bits$ $tag = 16\ bits$	$offset = 4\ bits$ $index = 0\ bits$ $tag = 28\ bits$

Calculations for Direct:

$$b\ (block\ size\ in\ byte) = 16B$$
$$n\ (number\ of\ ways) = 1$$
$$m\ (size\ of\ the\ cache\ in\ blocks) = \frac{256KB}{16B} = 16K$$
$$s\ (number\ of\ sets) = \frac{m}{n} = \frac{16K}{1} = 16K$$
$$offset = \log_2 b = \log_2 16 = 4\ bits$$
$$index = \log_2 s = \log_2 2^{14} = 14\ bits$$
$$tag = 32 - 14 - 4 = 14\ bits$$

Calculations for 2-way:

$$b\ (block\ size\ in\ byte) = 16B$$
$$n\ (number\ of\ ways) = 2$$
$$m\ (size\ of\ the\ cache\ in\ blocks) = \frac{256KB}{16B} = 16K$$
$$s\ (number\ of\ sets) = \frac{m}{n} = \frac{16K}{2} = 8K$$
$$offset = \log_2 b = \log_2 16 = 4\ bits$$
$$index = \log_2 s = \log_2 2^{13} = 13\ bits$$
$$tag = 32 - 14 - 4 = 15\ bits$$

Calculations for 4-way:

$$b\ (block\ size\ in\ byte) = 16B$$
$$n\ (number\ of\ ways) = 4$$
$$m\ (size\ of\ the\ cache\ in\ blocks) = \frac{256KB}{16B} = 16K$$
$$s\ (number\ of\ sets) = \frac{m}{n} = \frac{16K}{4} = 4K$$
$$offset = \log_2 b = \log_2 16 = 4\ bits$$
$$index = \log_2 s = \log_2 2^{12} = 12\ bits$$

$tag = 32 - 14 - 4 = 16\ bits$

Calculations for Fully-Associative:

$b\ (block\ size\ in\ byte) = 16B$
$offset = \log_2 b = \log_2 16 = 4\ bits$
$index = 0\ bit$
$tag = 32 - 4 = 28\ bits$

4- Assuming a 256KB cache and 32B block size. How does increasing the number of ways affect cache miss rate? Plot the number of ways (1,2,4,8,16) vs miss rate for the two traces. What do you observe and why?

Answer: As we see in the following chart, the miss rate decrease when we increase the number of ways; because in this way, we increased the associativity of the cache and it led to **reducing conflict misses**. So, we improved the miss rate by reducing the conflict misses. For the 10K trace it remains constant after 2-way because our benchmark is small and we have compulsory misses which increasing the number of ways cannot eliminate that.

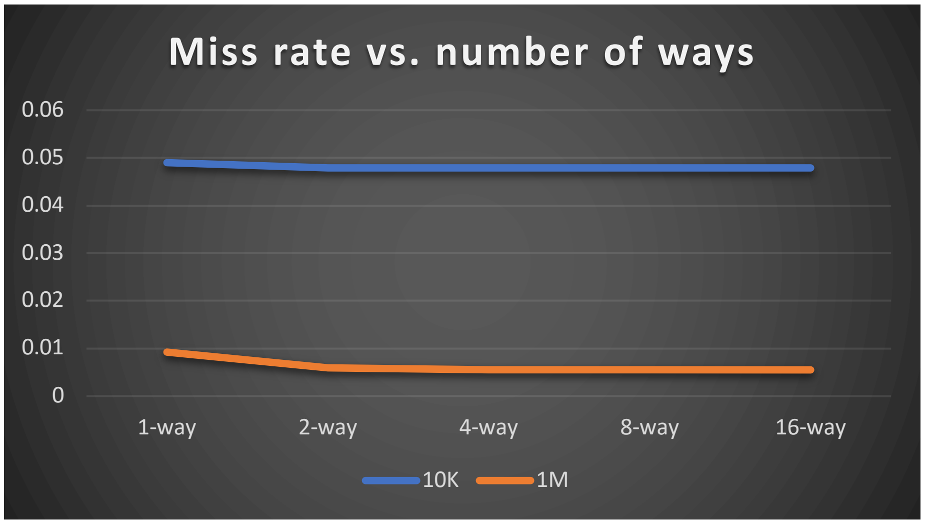


Figure 5: Miss rate vs number of ways

Trace	1-way	2-way	4-way	8-way	16-way
10K	0.049	0.0479	0.0479	0.0479	0.0479
1M	0.00923363	0.00599568	0.00552131	0.00550894	0.00550309

5- Assuming a 256KB 2-way set associative cache. How does varying the size of the block affect cache miss rate? Plot the block size (2B, 4B, 8B, 16B, 32B, 64B) vs miss rate for the two traces. What do you observe and why?

Answer: As we see in the following chart, the miss rate decrease when we increase block size because of the **special locality**. Special locality says that if something is accessed, something nearby will probability be accessed. We will increase the locality when we increase the block size because we transfer larger blocks to the cache and in this way, our miss rate will decrease because of increasing the special locality. But larger block size can increase miss penalty because generally it reduces the number of the total blocks in cache.

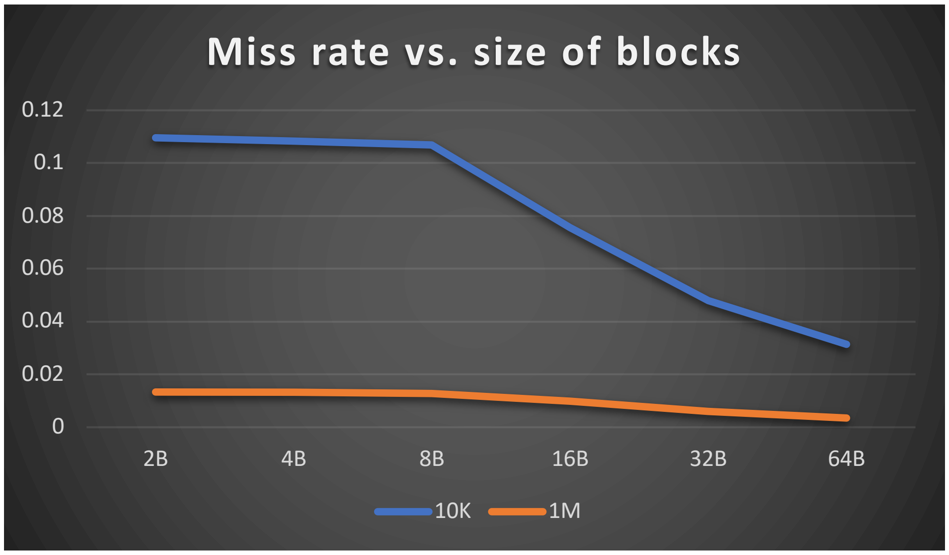


Figure 6: Miss rate vs. number of blocks

Trace	2B	4B	8B	16B	32B	64B
10K	0.1096	0.1084	0.107	0.0756	0.0479	0.0314
1M	0.01334028	0.01327586	0.01270584	0.00992729	0.00599568	0.00349694

6- Assuming a 2-way set associative cache and 32B block size. How does varying the size of the cache affect cache miss rate? Plot the cache size (64K, 128K, 256K, 512K, 1M, 2M) vs miss rate for the two traces. What do you observe and why?

Answer: As we see in the following chart, the miss rate decrease when we increase cache size because we **increase the number of blocks** in the cache. For the 10K trace it remains constant after 128K because our benchmark is small and we have compulsory misses which increasing the cache size cannot eliminate that.

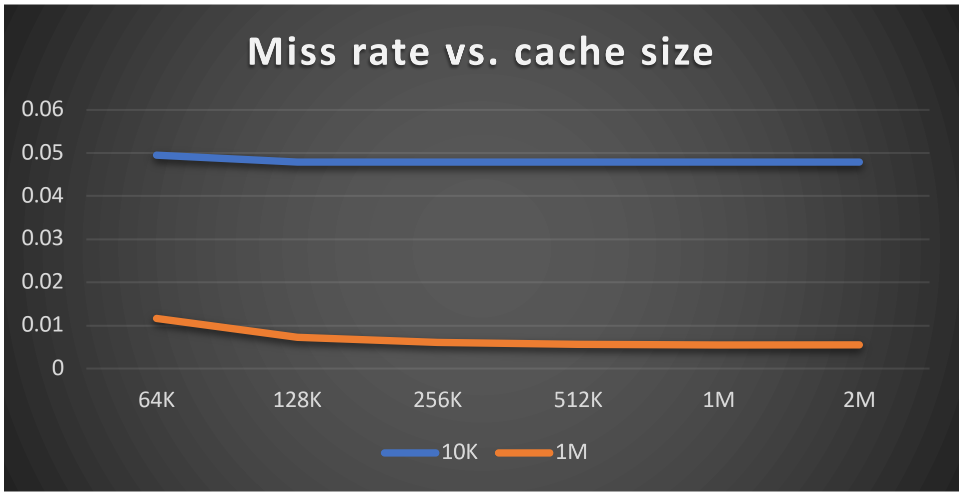


Figure 7: Miss rate vs. cache size

Trace	64K	128K	256K	512K	1M	2M
10K	0.0495	0.0479	0.0479	0.0479	0.0479	0.0479
1M	0.0479	0.00730621	0.00599568	0.00561696	0.0055187	0.00550113

7- Measuring cold, capacity, and conflict misses. (See "Measuring/Classifying Misses" in slides). In this problem, we will identify the types of misses for an 8KB, 4-way set associative cache with block size of 32B. For the gcc-1M trace, provide a breakdown of the type of cache misses. *You can provide the breakdown in terms of the miss rate.* For example, if the 8KB, 4-way cache has a 20% miss rate, an infinite size cache have a 1% miss rate, and a fully associative cache have a 10% miss rate, then 1% is due to cold misses, 9% is due to capacity misses, and 10% is due to conflict misses.

Answer:

$$Miss\ rate\ for\ 8KB, 4\ way\ set\ associative\ cache\ =\ 0.04985053104\ =\ 4.985053104\%$$

$$Miss\ rate\ for\ infinite\ size\ cache\ =\ 0.00550113419\ =\ 0.550113419\%$$

$$Miss\ rate\ for\ fully\ associative\ =\ 0.0445368\ =\ 4.45368\%$$

So, we have:

$$Compulsory\ misses\ =\ 0.550113419\%$$

$$Capacity\ misses\ =\ 4.45368 - 0.550113419\ =\ 3.903566581\%$$

$$Conflict\ misses\ =\ 4.985053104 - 3.903566581 - 0.550113419\ =\ 0.531373104\%$$