**Experimental Report**

1. **L1 Cache size vs. Miss Rate (Without L2)**In this experiment, only L1 cache was used and victim cache and L2 cache are disabled. The test is run for associativity = 1, 2, 4, 8, 16, 32 and each of the associativity were run for all 4 trace files: gcc\_trace.txt, go\_trace.txt, perl\_trace.txt, vortex\_trace.txt.
2. **For Associativity = 1**

**Miss rate table:**

For the above graph, it is known that the miss rate reduces as cache size increases.

**Average Access Time Table:**

From the above graph it can be concluded that AAT reduces as cache size increases but if cache size becomes large it has negative effect on AAT.

1. **For Associativity = 2**

**Miss rate table:**

For the above graph, it is shown that the miss rate reduces as cache size increases and becomes constant for larger cache sizes.

**AAT Table:**

From the above graph it can be concluded that AAT reduces as cache size increases but if cache size becomes large it has negative effect on AAT.

1. **For Associativity = 4**

**Miss rate table:**

For the above graph, it is shown that the miss rate reduces as cache size increases because the cache is now larger and hence less number of capacity misses are there.

**AAT Table:**

For associativity equal to 4, the average access time still increases as the cache size increases. This may be because of cache pollution for a small set associativity.

1. **For Associativity = 8**

**Miss rate table:**

For associativity equal to 8, the miss rate becomes constant as cache size increases. Since as cache becomes larger conflict misses also tend to reduce.

**AAT Table:**

For associativity equal to 8, from the above graph, it is shown that average access time also comes near to constant as cache size increases because all data may be found in cache. But for larger cache size AAT still increase due to cache pollution.

1. **For Associativity = 16**

**Miss rate table:**

As the set associativity increases, cache has more choices to place data and hence less conflict and capacity misses are observed which lead to constant miss rate as cache size increases.

**AAT Table:**

As set associativity increases, from the above graph, it is shown that average access time also comes near to constant as cache size increases because all data may be found in cache. But for larger cache size, AAT still increases due to cache pollution.

1. **For Associativity = 32**

**Miss rate table**

For set associativity such large, the miss rate is constant for very large cache size since most of the data is found in cache.

**AAT Table**

In the end, it can be concluded that as cache size increases the miss rate becomes constant. Initially the miss rate is higher because of compulsory misses, but it gradually decreases due to reduced capacity and conflict misses resulting from higher cache size. The AAT also reduces up to a point where there are less misses but increases for large cache size as cache may have been polluted because of its size.

1. **Associativity vs Miss Rate**

For this experiment, the L1 size and L2 size were kept constant. The victim cache was disabled. Below are the configurations for cache used:

Block size = 32 bytes

L1 size = 32 kB

No victim cache

L2 size = 64 kB

L2 associativity = 8

**Miss rate table**

From the graph it is shown that for associativity greater than 8, there is no effect on miss rate. The cache has stored enough data that the miss rate becomes constant.

**AAT Table**

From the above graph it is shown, for very large associativity the access time increases. This may be the case because larger caches are slower to access as processor has to search for data in set for every block.

1. **L2 cache size vs miss rate**

For this experiment, L1 size and associativity was kept constant. Also, victim cache was disabled too. The following are the configuration of cache used:

Block size = 32 bytes

L1 size = 1024 bytes

L1 associativity = 4

No victim cache

1. **For L2 Associativity = 4**

**Miss rate table**

From the above graph it is shown that the miss rate decreases as the size of L2 cache is increased. But for very large cache size the miss rate becomes constant.

**AAT Table**

Similar to miss rate graph, the AAT also becomes constant for large cache size as most of the data is found in L2 cache.

1. **For Associativity = 8**

**Miss rate table**

For the above graph, the miss rate becomes constant as cache size increase probably because as the capacity of cache increase, there are less number of capacity misses.

**AAT Table**

As presumed, the AAT also becomes constant since capacity and compulsory misses are now become constant. Hence, the access time has also become constant.

1. **For Associativity = 16**

**Miss rate table**

The above graph shows that miss rate further reduces and becomes constant for higher associativity because now the conflict misses are also reduced along with capacity misses. Hence the miss rate becomes constant.

**AAT Table**

Similar to Miss Rate graph, the AAT graph also shows the same trend. The AAT is reduces for large cache size and associativity as number of conflict misses are reduced along with capacity misses. Hence, AAT becomes constant for large cache sizes.

1. **Victim cache size vs miss rate**

In this experiment, the size of the victim cache is varied and compared to miss rate of L1. The L1 and L2 cache is kept constant. Following are the configurations used for this experiment:

Block size = 32 bytes

L1 size = 1024 bytes

L1 associativity = 4

L2 size = 4096 bytes

L2 associativity = 8

**Miss rate table**

From the above graph it can be seen that as victim cache size increase, the miss rate for L1 cache reduces. This may be due to as victim cache size increases, there are less number of L1 misses and data is found in victim cache. So L1 cache doesn’t need to go to L2.

**AAT Table**

Similar to miss rate graph, above graph also shows the similar trend. As victim cache size increases, more data is found in victim cache and less number of misses are there. Hence the time to search for L2 cache reduces and therefore, the average access time also reduces.

**Best Memory Hierarchy Configuration**

From the various configurations tested, it can be seen that an L1 cache with victim cache will perform better and will have lesser AAT than compared to only L1 cache and L2 cache. For very large L1 cache size, the AAT tends to increase exponentially potentially because of cache pollution, but L2 cache have almost constant AAT for large size. Therefore, the cache configuration that will give lowest AAT is:

Block Size = 32 bytes

L1 Cache Size = 16 kB

L1 Associativity = 4

Victim Cache size = 1024 bytes

L2 Cache Size = 32 kB

L2 Associativity = 8

This configuration gives lowest AAT because for this size the capacity misses are constant for all configuration but the conflict misses are lowest. As cache size increases, the AAT increases, as for same set many blocks have to be simultaneously searched.

The unique aspect of this configuration that makes it achieve the lowest AAT is the addition of victim cache to it which also further reduces AAT. In this way data is not evicted to L2 but remains at L1 cache layer.

**Comparing and Contrast Different Benchmarks**

The best memory hierarchy mentioned above has almost similar result for all the benchmark experiments. The results vary very lightly because of different cache sizes benchmarked in every test.

From the results of various benchmarks, it can be concluded that the compulsory misses remains the same for increasing cache size but reduces as we increase the set associativity. The capacity and conflict misses reduces as we increase the cache size and set associativity. Also, inclusion of victim cache reduces the number of misses when compared to configuration with no victim cache. Victim cache helps to reduce capacity and compulsory misses and lowers the AAT too.