CS622

Assignment 1

Group 22

NAME : ANUP ROY

ROLL No. : 20111403

NAME :SHILPA CHATTERJEE

ROLL No. : 20111057

1 Report on Question 1 of Assignment 1

1.1 Inclusive Cache Organization

Trace File Name	**L2 Cache**		**L3 Cache**	
h264ref	Hit Count	1378895	Hit Count	627532
	Miss Count	969678	Miss Count	342146
gromacs	Hit Count	3094660	Hit Count	166320
	Miss Count	336851	Miss Count	170531
hmmer	Hit Count	1766344	Hit Count	1352195
	Miss Count	1743421	Miss Count	391226
bzip	Hit Count	5259461	Hit Count	3951778
	Miss Count	5398166	Miss Count	1446388
sphinx	Hit Count	1933098	Hit Count	612987
	Miss Count	8820349	Miss Count	8207362
gcc	Hit Count	11574350	Hit Count	1663059
	Miss Count	3036461	Miss Count	1373402

Note : For the simulation we have not considered any block to be as dirty, so when a block gets evicted from L2 Inclusive Cache it is not written back to L3 Inclusive Cache

1.2 Exclusive Cache Organization

Trace File Name	**L2 Cache**		**L3 Cache**	
h264ref	Hit Count	1382949	Hit Count	821943
	Miss Count	965624	Miss Count	143681
gromacs	Hit Count	3094787	Hit Count	177422
	Miss Count	336724	Miss Count	159302
hmmer	Hit Count	1774443	Hit Count	1435276
	Miss Count	1735322	Miss Count	300046
bzip	Hit Count	5260051	Hit Count	4508355
	Miss Count	5397576	Miss Count	889221
sphinx	Hit Count	1938317	Hit Count	1594354
	Miss Count	8815130	Miss Count	7220776
gcc	Hit Count	11581002	Hit Count	1786985
	Miss Count	3029809	Miss Count	1242824

1.3 NINE Cache Organization

Trace File Name	L2 Cache		L3 Cache	
h264ref	Hit Count	1382949	Hit Count	632041
	Miss Count	965624	Miss Count	333583
gromacs	Hit Count	3094787	Hit Count	166265
	Miss Count	336724	Miss Count	170459
hmmer	Hit Count	1774443	Hit Count	1358978
	Miss Count	1735322	Miss Count	376344
bzip	Hit Count	5260051	Hit Count	3951730
	Miss Count	5397576	Miss Count	1445846
sphinx	Hit Count	1938317	Hit Count	609986
	Miss Count	8815130	Miss Count	8205144
gcc	Hit Count	11581002	Hit Count	1663561
	Miss Count	3029809	Miss Count	1366248

1.4 Analysis of the results obtained

1.4.1 Observations on L2 Cache miss count and hit count

- L2 inclusive cache miss count > L2 exclusive cache miss count = L2 nine cache miss count
- L2 inclusive cache hit count < L2 exclusive cache hit count = L2 nine cache hit count

This is because of the following reasons :-

1. The number of misses in L2 cache is the highest in case of inclusive cache organization as whenever a block is evicted from L3 cache it gets invalidated in L2 cache irrespective of it being the either least recently used or being mostly used block.

2. The L2 hit count is the lowest in case of inclusive cache organization as L3 cache is unaware of L2 cache hits and may replace a block which is frequently used in L2 , leading to back invalidation of the block in L2 , thereby lowering its hit rate as compared to nine and exclusive cache organizations.

3. The miss count and hit count of L2 exclusive cache and L2 nine cache are identical as when a block is replaced in L3 cache it has no effect on blocks currently residing in L2 cache.

   This can however be proved by the method of induction.
   We know that L2 nine and exclusive cache will start with empty blocks and hence the hit and miss count will be same at the beginning. Hence our base case is proved .
   Now , say at a time **'t'** let us consider that both L2 nine and exclusive caches have same contents . If we can show that at time **'t+1'** also their contents are same then our proof will be done.
   So, let us consider the state of the caches at time **'t+1'**. Since same address trace is given to both the caches(exclusive and nine) , so let us consider the following cases:-

   Case of hit : If it is a hit in L2 exclusive cache it will also be a hit in L2 nine cache as we have considered the cache contents to be same at time **'t'**.

   Case of miss : If it is a miss in L2 exclusive cache it surely is going to be a miss in L2 nine cache(again because the cache contents have been assumed to be same at time **'t'**) and now the block will be brought from either L3 or memory with no invalidation to the current residing blocks in L2 nine and L2 exclusive caches , thereby making its contents identical even at time **'t+1'**. Hence we have proved that as the contents of L2 nine and L2 exclusive caches are identical at any given point of time so their miss and hit counts are also identical.

1.4.2 Observations on L3 cache miss count and hit count

- L3 inclusive cache miss count > L3 nine cache miss count > L3 exclusive cache miss count
- L3 inclusive cache hit count < L3 nine cache hit count < L3 exclusive cache hit count

This is observed due to the following reasons:-

1. The number of misses in L3 inclusive cache is the highest because of the need to maintain the inclusive property i.e. L2 cache contents should be a subset of L3 cache contents , therefore the L3 cache capacity is not utilized to its full potential. The blocks that reside in L2 cache will be present in L3 cache as well but will always get a hit in L2 cache itself. Thus , at any point in time the space that is not exploited in L3 inclusive cache is equal to the size of L2 inclusive cache .

Whereas the number of misses in L3 exclusive cache is the least as the L3 cache contents ∩ L2 cache contents =ϕ and hence L3 can be utilized to its full capacity and there is no duplication of blocks between L2 and L3.

2. The number of hits in L3 exclusive cache is the highest whereas that of L3 inclusive cache is the lowest because , the number of distinct addresses present in L3 and L2 cache follows the pattern exclusive cache > nine > inclusive cache and hence the probability of getting a hit in L2 cache or L3 cache will also follow the same pattern.

3. Now if we compare the hit counts of L3 nine cache and L3 inclusive cache , the hit counts in L3 nine cache is more because of the back invalidation policy used in inclusion caches. A block being evicted from L3 cache may later get a hit in L2 cache in case of nine cache organization , which is not possible in cache inclusive caches.

2 Report on Question 2 of Assignment 1

2.1 Inclusive Cache Organization with Belady's Optimal Replacement Policy in L3 Fully Associative Cache

Trace File Name	**Total L3 Set Associative Miss Count**	**L3 Cache Cold Miss**	**L3 Cache Capacity Miss**	**L3 Cache Conflict Miss**
h264ref	342146	63703	47915	230528
Gromacs	170531	107962	35293	27276
Hmmer	391226	75884	77562	237780
Bzip	1446388	119753	417082	909553
Sphinx	8207362	122069	2946509	5138784
Gcc	1373402	773053	166234	434115

2.2 Inclusive Cache Organization with LRU Replacement Policy in L3 Fully Associative Cache

Trace File Name	Total L3 Set Associative Miss Count	L3 Cache Cold Miss	L3 Cache Capacity Miss	L3 Cache Conflict Miss
h264ref	342146	63703	272177	6266
Gromacs	170531	107962	61406	1163
Hmmer	391226	75884	301140	14202
Bzip	1446388	119753	1241648	84987
Sphinx	8207362	122069	8265179	-179886
Gcc	1373402	773053	596871	3478

2.3 Analysis on the results obtained

Total Miss =Cold Miss +Capacity Miss + Conflict Miss ---------- Eq. 1

Now in order to calculate cold, capacity and conflict miss , we made the L3 cache fully associative and in fully associative cache we know that conflict miss is theoretically zero , so our equation of total miss for L3 fully associative cache becomes –

Total Miss of Fully Associative Cache =Cold Miss +Capacity Miss ----------Eq. 2

Now , once we have calculated cold and capacity misses , we can now calculate conflict miss by placing the values of cold and capacity misses in Eq. 1.

- **Cold Miss** is the number of unique accesses made to the L3 cache and hence it is same for both the replacement policy (LRU and Belady) applications on L3 fully associative cache.
- **Capacity Miss** is the miss that occurs due to lack of space in a cache for accommodation of an incoming block. It can be observed the capacity miss count is more when LRU is used than when with Belady's Optimal replacement policy is used and this actually should be the case because Belady's Optimal Replacement policy provides the optimal solution and no other solution can go beyond the solution provided by it.
- **Conflict Miss** is the miss that occurs when several blocks are mapped to the same set or block frame because in case of set associative mapping a block can be placed only within the set inspite of empty blocks being present in other sets of the cache. These

misses can also occur in direct mapped block placement strategies. Since the capacity miss is more when LRU replacement policy is used , the conflict miss becomes less in case of LRU replacement policy implemented L3 cache than that of Belay replacement policy implemented L3 cache.

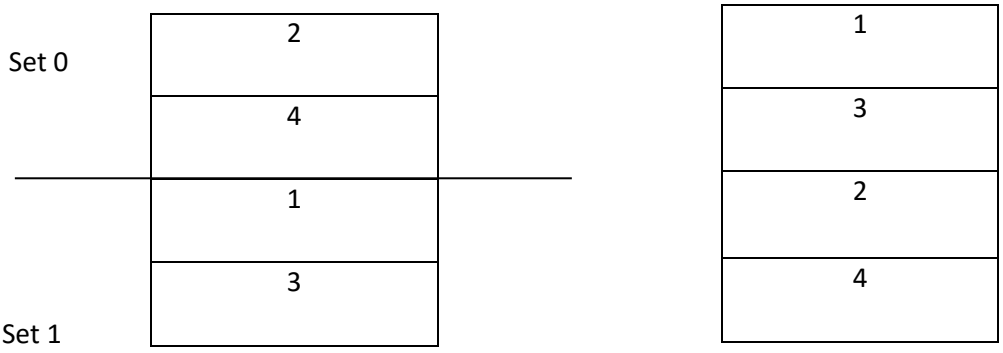
2.4 Analysis of Negative Conflict Miss Count in Sphinx

We think the following is one of the situation that may have caused negative conflict miss i.e. higher total miss count in fully associative cache than total miss count in set associative cache like the result obtained in case of Sphinx.

Let us consider a address trace 1,3 , 2, 4, 6, 8, 3

Let us consider a L2 cache with cache lines =4, associativity=2 and an L3 fully associative cache with cache lines=4

Now the contents of L2 and L3 cache after the address trace 1,3,2,4 have arrived are as follows

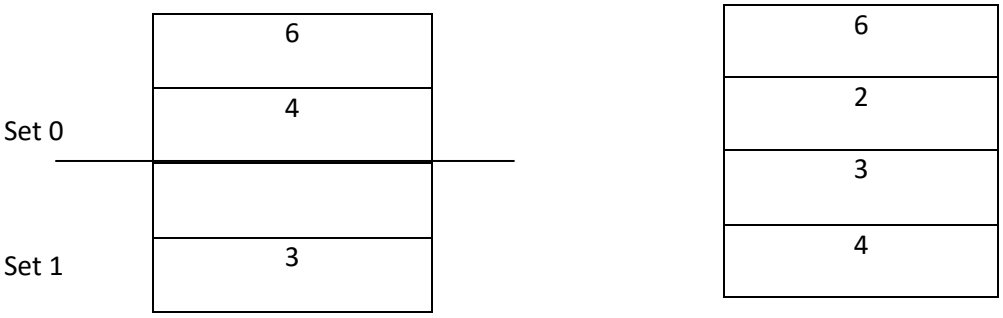


L2 2-way set associative Cache L3 Fully Associative Cache

(Assuming xth block is placed in L2 cache with Hashing Function x modulo y where y is the no. of sets)

Now when address 6 comes it misses in L2 and now starts searching in L3 and misses there as well. So the block of the address needs to be brought from main memory. Now in L3 Fully Associative Cache , since LRU is the replacement policy used , 1 will be evicted from L3 and since it is inclusive , so 1 must be invalidated from L2 as well and 6 will be placed in both L3 and L2 . But since L2 follows a hashing function x modulo y so 6 modulo 2 will place it in set 0 and thereby evicting the block containing address 2 and back invalidating block containing address 1 in L2 cache for placing one block.

Therefore the contents of L2 set associative cache and L3 fully associative cache are now as follows

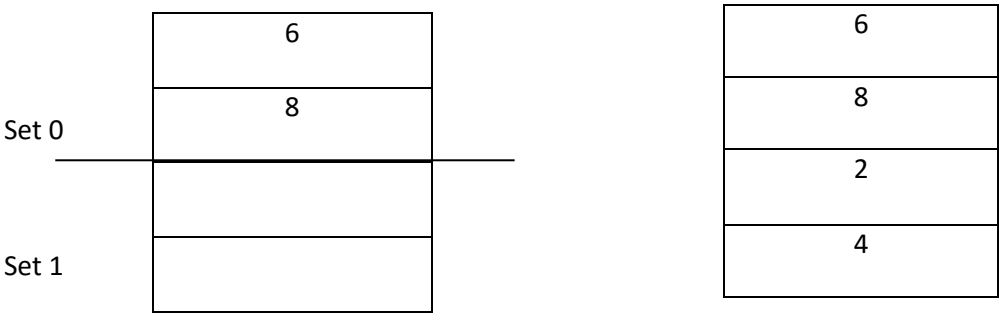


L2 2-way set associative Cache L3 Fully Associative Cache

(Assuming xth block is placed in L2 cache with Hashing Function x modulo y where y is the no. of sets)

Now if address 8 comes it is a miss in L2 and L3 hence we need to bring the block to L2 and L3. This block in L2 needs to be placed in set 0 thereby leading to an eviction in L2 cache block . In L3 it replaces block containing address 3 while in L2 it replaces block containing address 4 and invalidates block containing address to maintain inclusion property.

Therefore the contents of L2 set associative cache and L3 fully associative ache are now as follows



L2 2-way set associative Cache L3 Fully Associative Cache

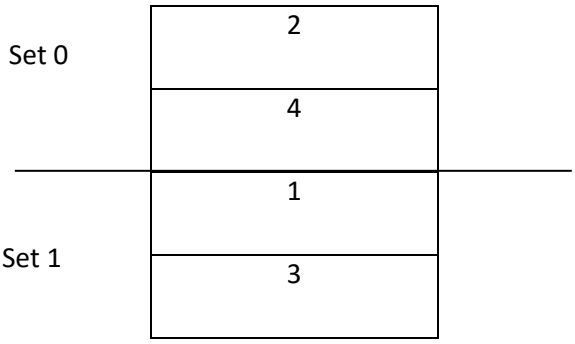
(Assuming xth block is placed in L2 cache with Hashing Function x modulo y where y is the no. of sets)

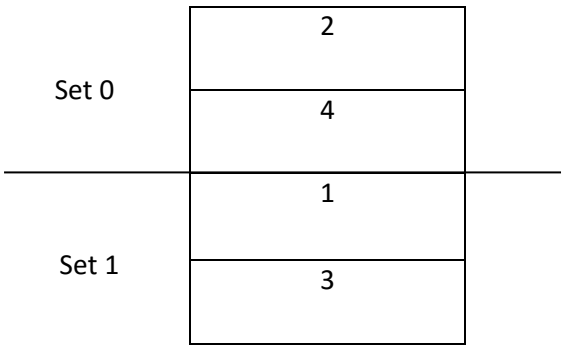
Now if address 3 comes it is going to miss both L2 and L3 and needs to be brought from main memory. In L3 cache it will be replacing block 2 and in L2 cache it will be placed in Set 1

Therefore the misses in L2 cache are 1,3,2,4,6,8,3(all misses for the address trace considered) [Miss Count = 7] and the misses in L3 cache are 1,3,2,4,6,8,3 [Miss Count = 7].

Now let us see what difference it would have been in miss count if L3 cache was set associative with cache lines=4 and associativity=2-way

Now the contents of L2 and L3 cache after the address trace 1,3,2,4 have arrived are as follows

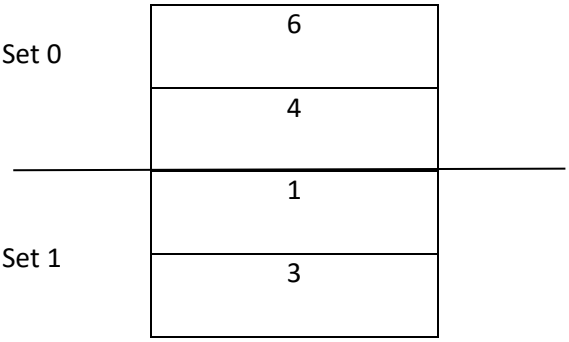


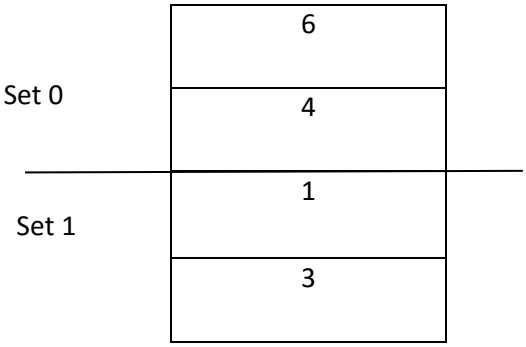


L2 2-way set associative Cache L3 Set Associative Cache

(Assuming xth block is placed in L2 cache with Hashing Function x modulo y where y is the no. of sets)

Now , when address 6 comes it is going to miss both L2 and L3 caches and hence it will be copied in both of them in set 0 by replacing address 2 in both L2 and L3 as it is the least recently used .

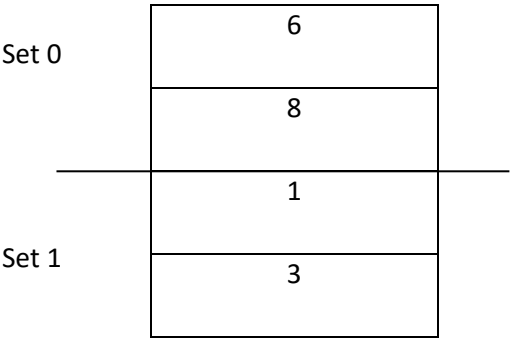


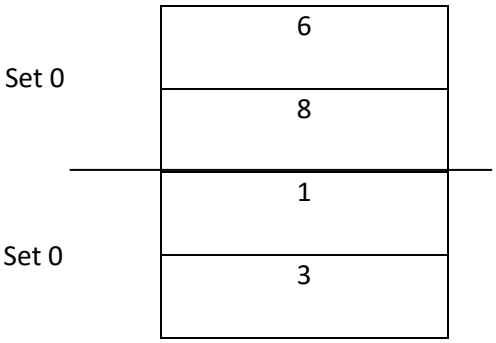


L2 2-way set associative Cache L3 Set Associative Cache

(Assuming xth block is placed in L2 cache with Hashing Function x modulo y where y is the no. of sets)

Now when address 8 comes in it evicts block containing address 4 from both L2 and L3 . Next when address 3 comes it hits in L2 cache.





Now the misses for L2 cache will be 1,3,2,4,6,8(entire address trace considered for this example) [Miss Count = 6]and L3 set associative cache will be 1,3,2,4,6,8 [Miss Count =6].

So, for this small example we could see that it might so happen that the L3 miss count for fully associative cache can be more than the L3 miss count for set associative cache and hence when we try to calculate Conflict miss of L3 set associative cache by subtracting cold and capacity miss of L3 fully associative cache from L3 set associative cache total miss it comes out a negative value.