Assignment Set Number:

Problem Bank 28

Group Name:

108

Contribution Table:

Sl. No.	**Name (as appears in Canvas)**	**ID NO**	**Contribution**
1.	**MOHD SAIF ALI**	**2021fC04026**	**Equal (100%)**
2.	**REDDYVARI SASI KUMAR REDDY**	**2021FC04030**	**Equal (100%)**
3.	**DHRUBA ADHIKARY**	**2021FC04034**	**Equal (100%)**

Part I: Direct Mapped Cache

a) **Execute the above program by setting block size to 2, 4, 8, 16 and 32 for cache size = 8, 16 and 32. Record the observation in the following table**.

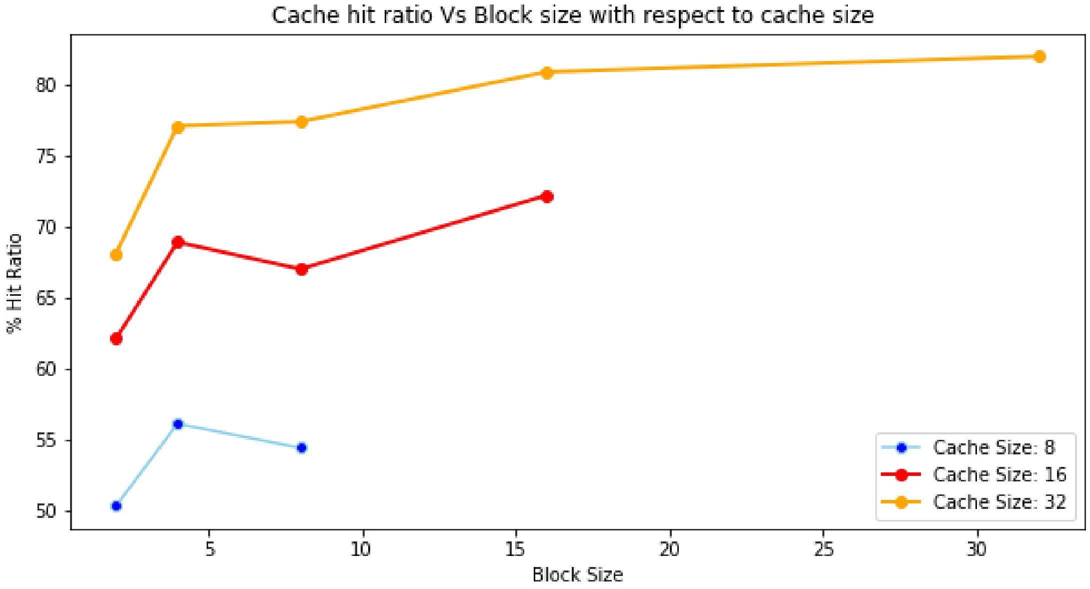
The following are the Observations from **"Data Cache"**

Block Size	Cache size	# Hits	# Misses	% Miss Ratio	%Hit Ratio
2	8	184	182	49.7	50.3
4		205	161	43.9	56.1
8		199	167	45.6	54.4
2	16	227	139	37.9	62.1
4		252	114	31.2	68.8
8		245	121	33.1	66.9
16		264	102	27.8	72.2
2	32	249	117	31.9	68.1
4		282	84	22.9	77.1
8		283	83	22.7	77.3
16		296	70	19.2	80.9
32		300	66	18.1	81.9

The following are the Observations from **“Instruction Cache”**

Block Size	Cache size	# Hits	# Misses	% Miss Ratio	%Hit Ratio
2	8	171	199	53.7	46.3
4		258	112	30.2	69.8
8		301	69	18.6	81.4
2	16	251	119	32.2	67.8
4		292	78	21.1	78.9
8		312	58	15.7	84.3
16		328	42	11.4	88.6
2	32	288	82	22.2	77.8
4		325	45	12.2	87.8
8		336	34	9.2	90.8
16		352	18	4.8	95.2
32		360	10	2.7	97.3

b) **Plot a single graph of Cache hit ratio Vs Block size with respect to cache size = 8, 16 and 32. Comment on the graph that is obtained**.



c) Fill the below table and write a small note on your observation from the data cache.

- Block Size = 2
- Cache Size = 32
- Cache Type = Direct Mapped

Addresses	Data	Miss (%)
0006	**02**	**31.9**
0007	**00**	**31.9**
0008	**0C**	**31.9**
0009	**02**	**31.9**
0052	**02**	**31.9**
0053	**00**	**31.9**
0054	**07**	**31.9**
0055	**02**	**31.9**
0056	**00**	**31.9**
0057	**07**	**31.9**
0058	**02**	**31.9**
0059	**00**	**31.9**
0060	**07**	**31.9**
0061	**02**	**31.9**
0062	**00**	**31.9**
0063	**07**	**31.9**
0074	**20**	**31.9**
0075	**62**	**31.9**
0076	**65**	**31.9**
0077	**20**	**31.9**
0078	**73**	**31.9**
0079	**65**	**31.9**
0080	**61**	**31.9**
0081	**72**	**31.9**
0082	**63**	**31.9**

0083	**68**	**31.9**
0096	**46**	**31.9**
0097	**6F**	**31.9**
0098	**75**	**31.9**
0099	**6E**	**31.9**
0100	**64**	**31.9**
0101	**00**	**31.9**

Binary Search is implemented. The key to be searched is traversed in the loop. Due to the Direct mapping technique, a block is placed only in one cache line. 16 blocks exist. As the block referenced gets increased i.e cache size is larger, hit ratio % is more here. Addresses gets trashed in the bottom up execution. Misses are more in the middle of the execution to traverse back since 2 blocks reference in the same line is more. Optimal values or parameters need to be considered for a good performance.

CPI = 6.63

Part II: Associative Mapped Cache

a) **Execute the above program by setting block size to 2, 4, 8, 16 and 32 for cache size = 8, 16 and 32. Record the observation in the following table**.

The following are the Observations from **"Data Cache"**

LRU Replacement Algorithm					
Block Size	Cache size	# Hits	# Misses	% Miss Ratio	%Hit Ratio
2	8	198	168	45.9	54.1
4		196	170	46.5	53.5
8		199	167	45.6	54.4
2	16	215	151	41.3	58.7
4		266	100	27.3	72.7
8		240	126	34.4	65.6
16		264	102	27.8	72.2
2	32	266	100	27.3	72.7
4		303	63	17.2	82.8
8		299	67	18.3	81.7

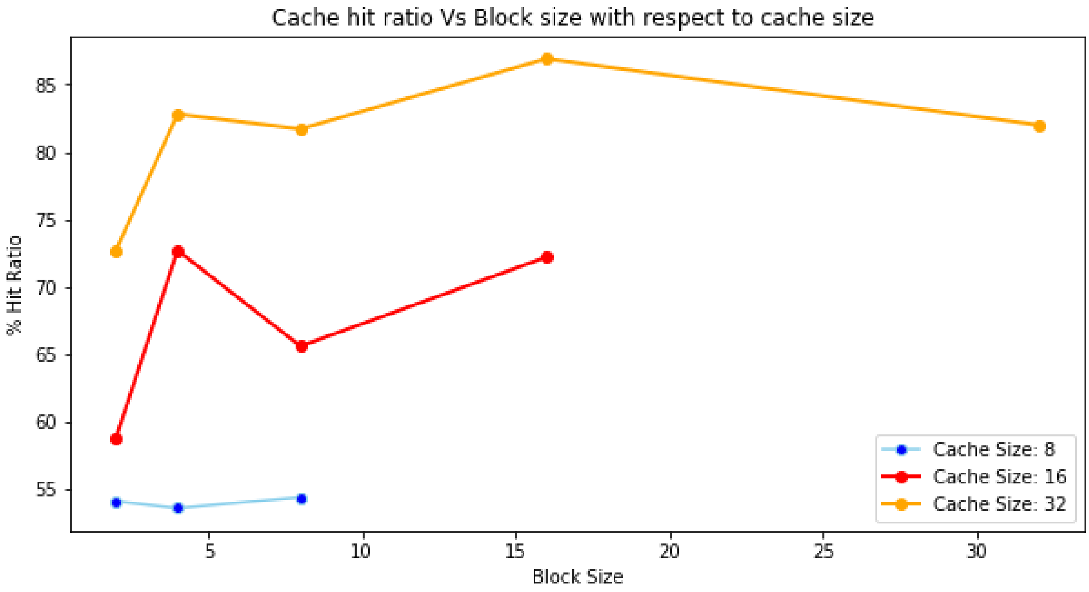
16		318	48	13.1	86.9
32		300	66	18.1	81.9

The following are the Observations from **“Instruction Cache”**

LRU Replacement Algorithm					
Block Size	Cache size	# Hits	# Misses	% Miss Ratio	%Hit Ratio
2	8	171	199	53.8	46.2
4		258	112	30.3	69.7
8		301	69	18.6	81.4
2	16	171	199	53.8	46.2
4		258	112	30.3	69.7
8		302	68	18.4	81.6
16		328	42	11.4	88.6
2	32	273	97	26.2	73.8
4		316	54	14.6	85.4
8		336	34	9.2	90.8
16		352	18	4.8	95.2
32		360	10	2.7	97.3

b) Plot a single graph of Cache hit ratio Vs Block size with respect to cache size = 8, 16 and 32. Comment on the graph that is obtained.

The plot based on “data cache”



As the implementation is Associative Mapped technique, A block of main memory can load into any cache line and is interpreted as tag and word. It is more flexible because it is similar to General Tickets in a Train booking system but the complexity of comparator circuit is high. Cache searching is high due to it. As cache size increases, Hit ratio achieved is higher but with an optimal block size.

c) Fill up the following table for three different replacement algorithms and state which replacement algorithm is better and why?

<table>
<tr><td colspan="5">Replacement Algorithm: Random</td></tr>
<tr><td>Block Size</td><td>Cache size</td><td>Miss</td><td>Hit</td><td>Hit ratio</td></tr>
<tr><td>2</td><td>4</td><td>218</td><td>148</td><td>40.4 %</td></tr>
<tr><td>2</td><td>8</td><td>183</td><td>183</td><td>50 %</td></tr>
<tr><td>2</td><td>16</td><td>132</td><td>234</td><td>64 %</td></tr>
<tr><td>2</td><td>32</td><td>99</td><td>267</td><td>73 %</td></tr>
<tr><td>2</td><td>64</td><td>93</td><td>273</td><td>74.6 %</td></tr>
<tr><td colspan="5">Replacement Algorithm: FIFO</td></tr>
<tr><td>Block Size</td><td>Cache size</td><td>Miss</td><td>Hit</td><td>Hit ratio</td></tr>
<tr><td>2</td><td>4</td><td>225</td><td>141</td><td>38.5 %</td></tr>
<tr><td>2</td><td>8</td><td>171</td><td>195</td><td>53.3 %</td></tr>
<tr><td>2</td><td>16</td><td>155</td><td>211</td><td>57.6 %</td></tr>
<tr><td>2</td><td>32</td><td>106</td><td>260</td><td>71.1 %</td></tr>
<tr><td>2</td><td>64</td><td>89</td><td>277</td><td>75.7 %</td></tr>
<tr><td colspan="5">Replacement Algorithm: LRU</td></tr>
<tr><td>Block Size</td><td>Cache size</td><td>Miss</td><td>Hit</td><td>Hit ratio</td></tr>
<tr><td>2</td><td>4</td><td>225</td><td>141</td><td>38.5 %</td></tr>
</table>

2	8	168	198	54.1 %
2	16	151	215	58.8 %
2	32	100	266	72.7 %
2	64	87	279	76.3 %

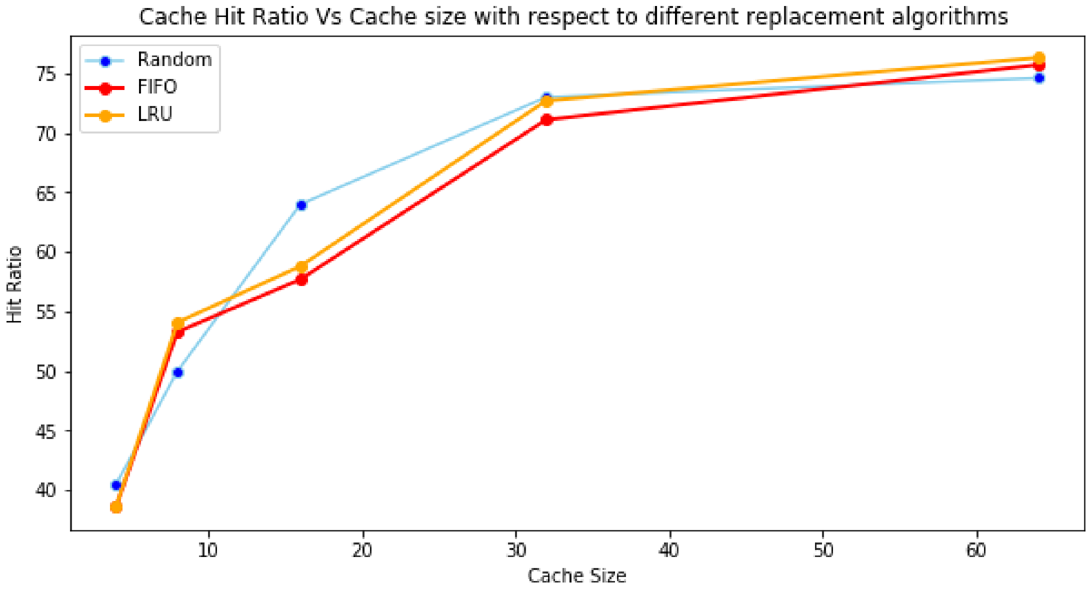
In FIFO, the block that has entered into cache memory first then it is replaced first by the newly received block from Main Memory.

In LRU, the block is replaced if it is there in cache for longest duration without referring by CPU.

In Random, there is no fixed procedure for selection of block replaced by newly received block in cache.

LRU performs better as the requests for the replacement in cache with good locality of reference. It performs with lease no of page faults.

d) Plot the graph of Cache Hit Ratio Vs Cache size with respect to different replacement algorithms. Comment on the graph that is obtained.



Obviously, we can achieve a good performance with the increase in cache size to an optimal level. As cache size increases, Hit ratio % is higher to achieve an optimal value. It exists with direct proportional relationship. In regards with less no of Page Faults and Optimal performance based on Locality, LRU performs in optimal fashion. It may not be same in Random Replacement Algorithm.

Part III: Set Associative Mapped Cache

Execute the above program by setting the following Parameters:

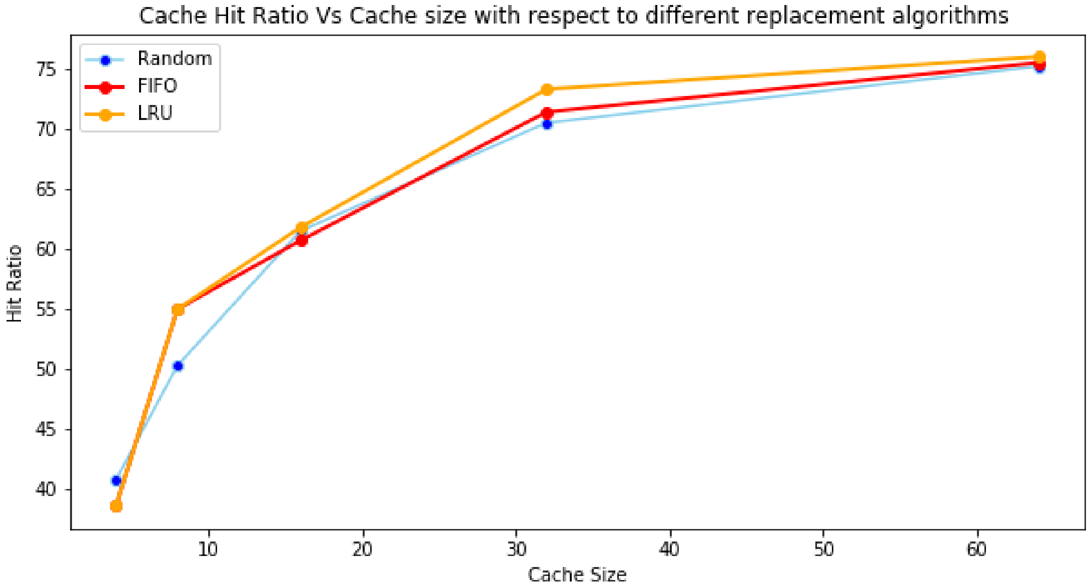
- Number of sets (Set Blocks): 2 way
- Cache Type: Set Associative
- Replacement: LRU/FIFO/Random

a) **Fill up the following table for three different replacement algorithms and state which replacement algorithm is better and why?**

<table>
<tr><td colspan="5">Replacement Algorithm: Random</td></tr>
<tr><td>Block Size</td><td>Cache size</td><td>Miss</td><td>Hit</td><td>Hit ratio</td></tr>
<tr><td>2</td><td>4</td><td>217</td><td>149</td><td>40.8 %</td></tr>
<tr><td>2</td><td>8</td><td>182</td><td>184</td><td>50.3 %</td></tr>
<tr><td>2</td><td>16</td><td>141</td><td>225</td><td>61.5 %</td></tr>
<tr><td>2</td><td>32</td><td>108</td><td>258</td><td>70.5 %</td></tr>
<tr><td>2</td><td>64</td><td>91</td><td>275</td><td>75.2 %</td></tr>
<tr><td colspan="5">Replacement Algorithm: FIFO</td></tr>
<tr><td>Block Size</td><td>Cache size</td><td>Miss</td><td>Hit</td><td>Hit ratio</td></tr>
<tr><td>2</td><td>4</td><td>225</td><td>141</td><td>38.5 %</td></tr>
<tr><td>2</td><td>8</td><td>165</td><td>201</td><td>54.9 %</td></tr>
<tr><td>2</td><td>16</td><td>144</td><td>222</td><td>60.6 %</td></tr>
<tr><td>2</td><td>32</td><td>105</td><td>261</td><td>71.3 %</td></tr>
<tr><td>2</td><td>64</td><td>90</td><td>276</td><td>75.4 %</td></tr>
<tr><td colspan="5">Replacement Algorithm: LRU</td></tr>
<tr><td>Block Size</td><td>Cache size</td><td>Miss</td><td>Hit</td><td>Hit ratio</td></tr>
<tr><td>2</td><td>4</td><td>225</td><td>141</td><td>38.5 %</td></tr>
<tr><td>2</td><td>8</td><td>165</td><td>201</td><td>54.9 %</td></tr>
<tr><td>2</td><td>16</td><td>140</td><td>226</td><td>61.7 %</td></tr>
<tr><td>2</td><td>32</td><td>98</td><td>268</td><td>73.2 %</td></tr>
<tr><td>2</td><td>64</td><td>88</td><td>278</td><td>75.9 %</td></tr>
</table>

As it is Set Associative mapping technique, which we can say it as combination of Direct and Associative mappings. The available cache memory is divided into no of sets if cache memory lines divides with some k value which is k-way set associative mapping. LRU performs better, with the increase in cache size, locality of reference is exhibited at maximum when compared with the FIFO & Random.

b) **Plot the graph of Cache Hit Ratio Vs Cache size with respect to different replacement algorithms. Comment on the graph that is obtained.**



With the increase in the size of Cache, % Hit Ratio increases to an optimal level.

c) **Fill in the following table and analyse the behaviour of Set Associate Cache. Which one is better and why?**

Replacement Algorithm: LRU				
Block Size, Cache size	Set Blocks	Miss	Hit	Hit ratio
2, 64	2 – Way	88	278	75.9 %
2, 64	4 – Way	88	278	75.9 %
2, 64	8 – Way	88	278	75.9 %

In the K – Way Set Associative Mapping Technique, It is addressed as Tag, Set, Word. No of sets is calculated by dividing cache memory lines with k. As experimented above, if there is an increase in the set blocks, the misses and hits remains same. Hence the Hit Ratio is same with no change.