

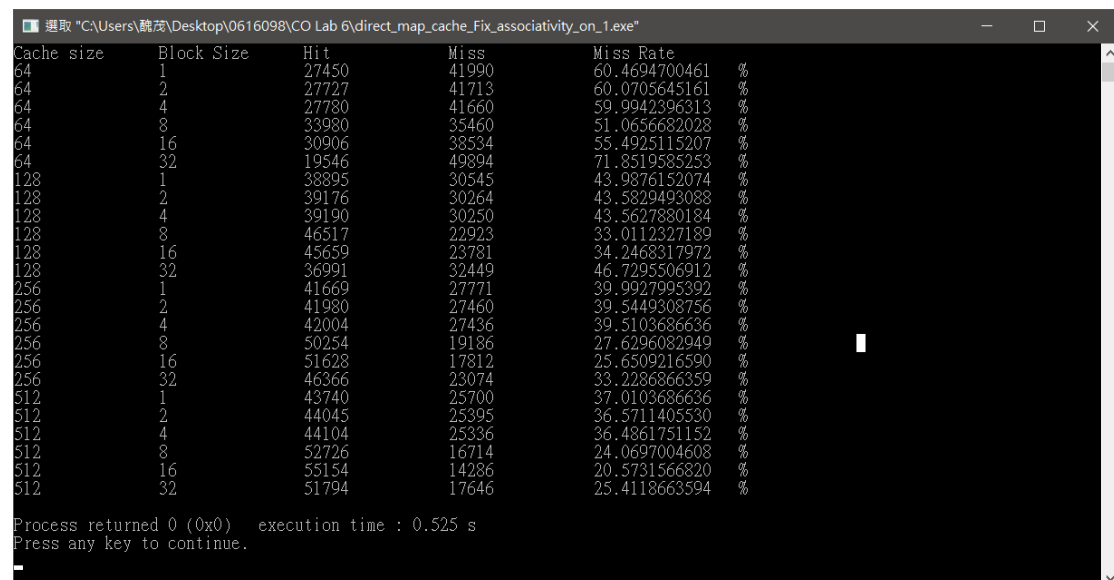
Computer Organization Lab 6

0616098 黃秉茂

Result:

(1) direct_map_cache_Fix_associativity_on_1.cpp (使用的資料為 Trace.txt)

更改 direct_map_cache 的路徑，多 include 其他函式庫，跑個雙層 for-loop 和處理輸出部分就完成了。

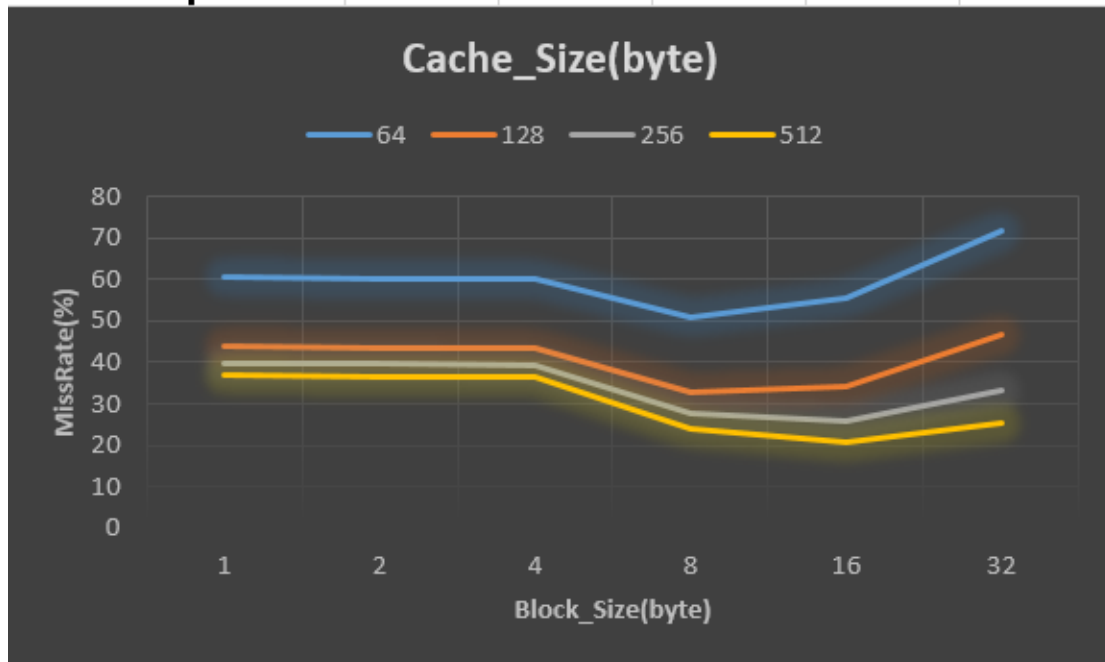


Cache size	Block Size	Hit	Miss	Miss Rate
64	1	27450	41990	60.4694700461 %
64	2	27727	41713	60.0705645161 %
64	4	27780	41660	59.9942396313 %
64	8	33980	35460	51.0656682028 %
64	16	30906	38534	55.4925115207 %
64	32	19546	49894	71.8519585253 %
128	1	38895	30545	43.9876152074 %
128	2	39176	30264	43.5829493088 %
128	4	39190	30250	43.5627880184 %
128	8	46517	22923	33.0112327189 %
128	16	45659	23781	34.2468317972 %
128	32	36991	32449	46.7295506912 %
256	1	41669	27771	39.9927995392 %
256	2	41980	27460	39.5449308756 %
256	4	42004	27436	39.5103686636 %
256	8	50254	19186	27.6296082949 %
256	16	51628	17812	25.6509216590 %
256	32	46366	23074	33.2286866359 %
512	1	43740	25700	37.0103686636 %
512	2	44045	25395	36.5711405530 %
512	4	44104	25336	36.4861751152 %
512	8	52726	16714	24.0697004608 %
512	16	55154	14286	20.5731566820 %
512	32	51794	17646	25.4118663594 %

Process returned 0 (0x0) execution time : 0.525 s
Press any key to continue.

以不同的 Cache Size 和 Block Size 觀察 Miss Rate 的變化，結果如下圖：

	Block_Size					
Cache_Size	1	2	4	8	16	32
64	60.4695	60.0706	59.9942	51.0657	55.4925	71.852
128	43.9876	43.5829	43.5628	33.0112	34.2468	46.7296
256	39.9928	39.5449	39.5104	27.6296	25.6509	33.2287
512	37.0104	36.5711	36.4862	24.0697	20.5732	25.4119



從結果中可以發現到，在相同 Block Size 下，Cache Size 從 64 Bytes 到 512 Bytes 時，因為可用的 Lines 增加，所以不容易發生重複，因此 Miss Rate 顯著下降。

而在相同 Cache Size 下，加大 Block Size 可以提升 spatial locality 效率，進而降低 Miss Rate。在 Block Size 由 1 到 4 時下降的極為緩慢，結果並不顯著；4 變成 8 時下降的效果最為明顯，但是在 16 到 32 時出現不同結果，Miss Rate 反而大幅增加，推測是因為 Block Size 已經大幅超越 spatial locality 的區域，反而會因為 block size 增加，line 減少，導致 index 容易重複，產生 replace。

(2) direct_mapped_cache_lru_Fix_block_size_on_32B.cpp
(使用的資料為 Trace.txt)

1. N-way set associative：將原本的 Cache size 切分為 N 份，因此 set 總共有 $\text{Cache size}/N$ 個，由 index bits 決定哪一個 set，並且每個 set 又各自給予 N 個 cache content，總共大小仍為 Cache size。

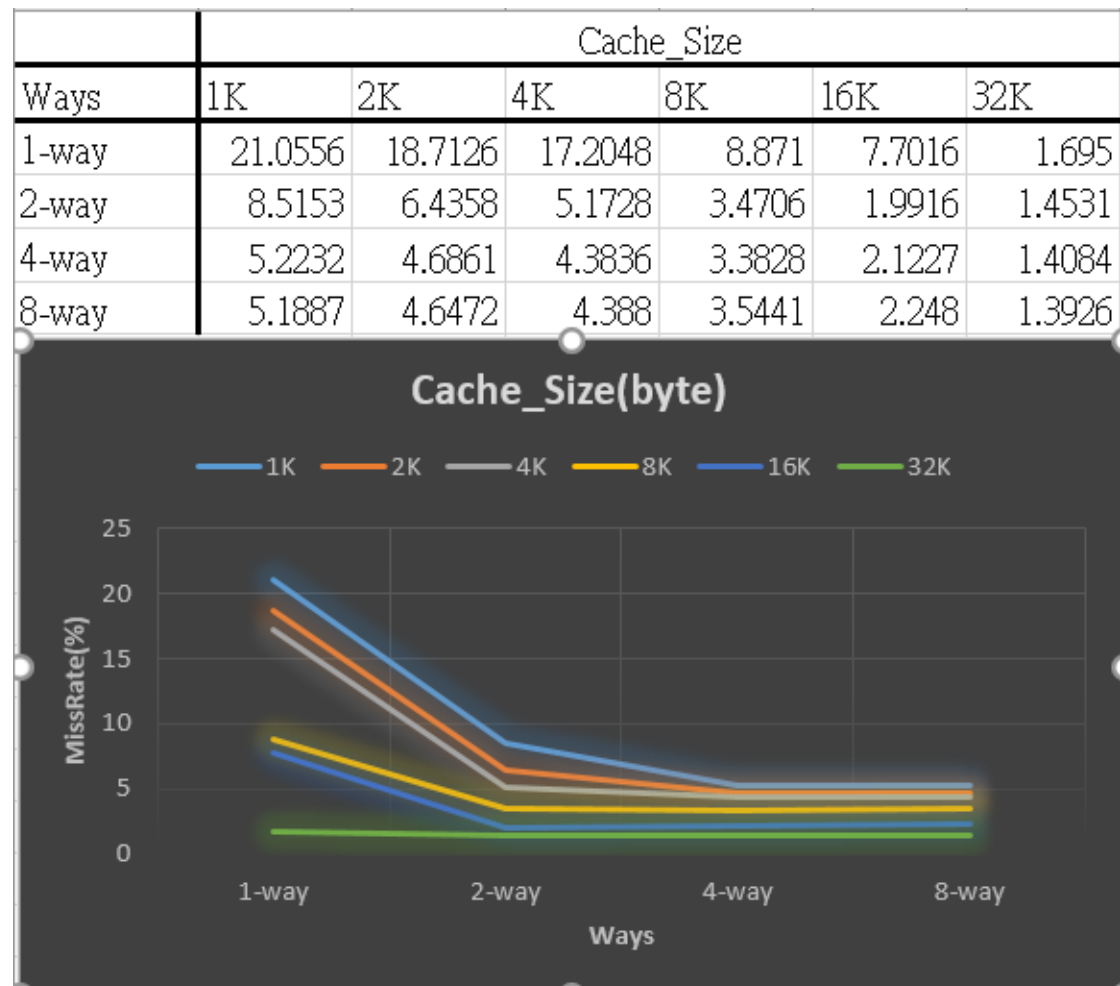
2. LRU (Least-Recent Used)：在每個 Cache content 中儲存 count 值，共 $\log_2(N)$ 個 bits，用來記錄該 content 在同一 set 中的使用新舊順序，並按照以下方法進行：

- 一開始每個 set 中的 N 個 content 各自寫入 0, 1, 2, ..., N-1 的值，0 為最舊，N-1 為最新。
- 若搜尋時沒有找到目標，則會將 count 為 0 的 content 進行 replace，同時 count 更新為 N-1，表示 replace 後的資料變成最新的，並將相同 set 其餘 content 的 count 都減一。
- 若搜尋時找到目標，則將該目標的 count 改為 N-1 表示其為最新，其餘 content 的 count 若大於目標原本 count，則 count 減一，否則 count 不變，如此一來目標順序被提升至最新，中間的 content 也會被往下推一個順位。

```
選取 "C:\Users\曉茂\Desktop\0616098\CO Lab 6\direct_mapped_cache_lru_Fix_block_size_on_32B.exe"
Cache size    Ways    Hit    Miss    Miss Rate
1K            1       54819   14621   21.0555875576 %
1K            2       63527   5913    8.5152649770 %
1K            4       65813   3627    5.2232142857 %
1K            8       65837   3603    5.1886520737 %
2K            1       56446   12994   18.7125576037 %
2K            2       64971   4469    6.4357718894 %
2K            4       66186   3254    4.6860599078 %
2K            8       66213   3227    4.6471774194 %
4K            1       57493   11947   17.2047811060 %
4K            2       65848   3592    5.1728110599 %
4K            4       66396   3044    4.3836405530 %
4K            8       66393   3047    4.3879608295 %
8K            1       63280   6160    8.8709677419 %
8K            2       67030   2410    3.4706221198 %
8K            4       67091   2349    3.3827764977 %
8K            8       66979   2461    3.5440668203 %
16K           1       64092   5348    7.7016129032 %
16K           2       68057   1383    1.9916474654 %
16K           4       67966   1474    2.1226958525 %
16K           8       67879   1561    2.2479838710 %
32K           1       68263   1177    1.6949884793 %
32K           2       68431   1009    1.4530529954 %
32K           4       68462   978     1.4084101382 %
32K           8       68473   967     1.3925691244 %

Process returned 0 (0x0)   execution time : 0.944 s
Press any key to continue.
```

以不同的 Cache Size 和 Associativity 觀察 Miss Rate 的變化，結果如下圖：

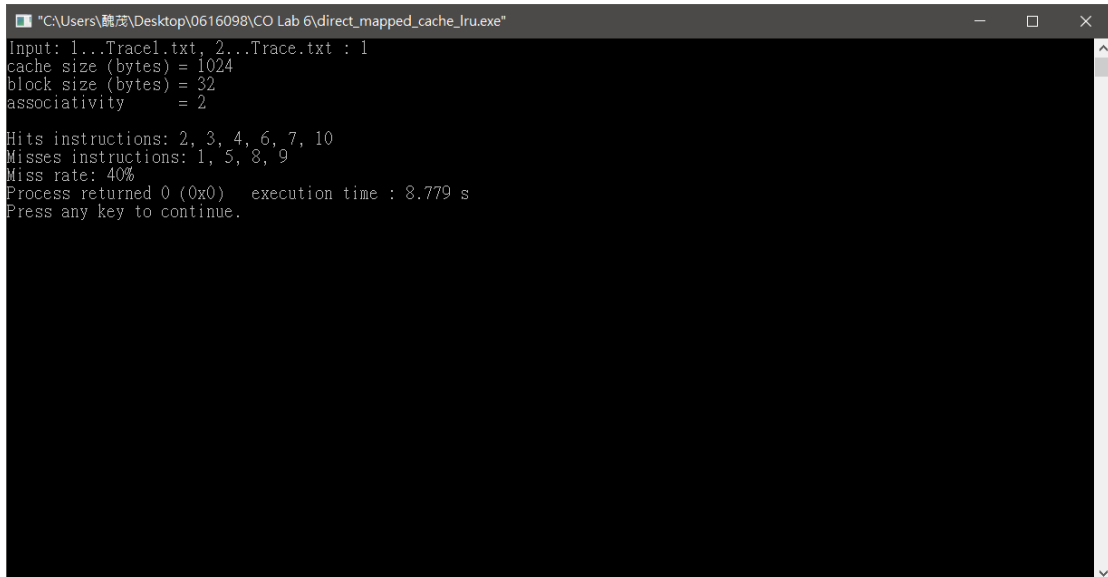


從結果可以看到，Ways 和 Cache size 增加都會顯著降低 Miss Rate，Cache size 的影響算是很明顯，尤其是 Associativity 很低時，因為可用的 Lines 增加，所以不容易發生重複，因此 Miss Rate 顯著下降。

而 Associativity 造成的改變則是在 1-way 變成 2-way 時最為顯著，推測原因為其 spatial locality 特徵明顯，容易重複 index，因此使用更多的 ways 可以避免頻繁 replace 導致的 miss，並且利用 LRU 安排 replace 次序，進一步降低 miss rate。

(3) direct_mapped_cache_lru.cpp

改變 `direct_mapped_cache_lru_Fix_block_size_on_32B.cpp`，讓 `block_size` 成為變數，並用 `vector` 紀錄 instruction 就結束了。



```
"C:\Users\翽茂\Desktop\0616098\CO Lab 6\direct_mapped_cache_lru.exe"
Input: 1...Tracel.txt, 2...Trace.txt : 1
cache size (bytes) = 1024
block size (bytes) = 32
associativity      = 2

Hits instructions: 2, 3, 4, 6, 7, 10
Misses instructions: 1, 5, 8, 9
Miss rate: 40%
Process returned 0 (0x0)   execution time : 8.779 s
Press any key to continue.
```

Problems you met and solutions:

最困難的在於 LRU 要如何紀錄使用頻率，後來特別在 cache content 多了 cnt 來記錄。而因為 set 內能存指一筆資料所以 cache 要用二維的方式儲存，後來用 double pointer 跟 2D-array 的方式完成。比較難的大概就是處理這些問題和理解 cache 的實際操作。

Summary:

我更了解了 cache 的操作，也了解到當有 spatial locality 的特性時，cache 是多麼的重要，也知道 Associativity 能夠造成的影響，這個 lab 讓我更理解 cache 的細節和實作，很多地方尤其是 LRU 更需要一些巧思。