第三次作業

Page Replacement 方法

1. 開發環境

   Window 10

   使用 visual Studio Code 環境

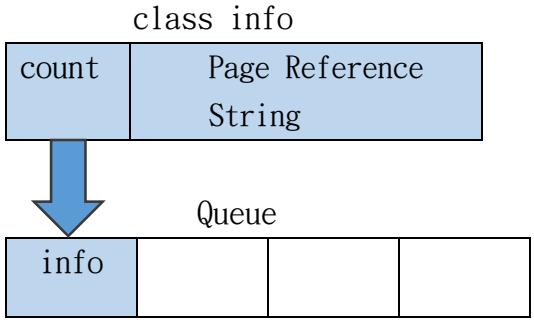
   程式語言 Python

2. 實作方法、資料結構及運作流程

   資料結構 : 一個 count 用來計數，一個放 Page Reference String

   Page_fault = 0 # 紀錄發生幾次 Page fault

   Page_Replaces = 0 # 紀錄發生幾次 Page Replaces



結構如圖

基本流程都一樣:

判斷有沒有發生 page fault

判斷有沒有發生 Page Replaces

甲、FIFO

先判斷有沒有發生 page fault，有就 Page_fault 次數加一，再判斷 queue 是否已滿(Page Replaces)，沒有就將新的 Page Reference String 插入 queue 最前面；有就將 Page_Replaces 次數加一， FIFO 先進去先出去的判斷方式，找到該 Page Reference String 抽換掉，將新的新的 Page Reference String 插入 queue 最前面。

乙、LRU

與 FIFO 大致相同，僅需修改兩部分。第一，沒有發生 Page_fault 時，代表 queue 內已有該值，要將 Page Reference String 更新。第二，queue 已滿(Page Replaces)，要使用 LRU 方法代換，此方法是找最不常被參考、修改的 Page Reference String 做抽換。

丙、LFU + FIFO

（此方法會使用到 count 計數）

先判斷有沒有發生 page fault，有就 Page_fault 次數加一；若是沒有，就找到重複被參考的 Page Reference String 將 count 加一，再判斷 queue 是否已滿(Page Replaces)，沒有就將新的 Page Reference String 插入 queue 最前面；有就將 Page_Replaces 次數加一，使用 LFU + FIFO 判斷方式，先找 count 最小的 Page Reference String，如果一樣就找最先進 queue 的 Page Reference String，找到該 Page Reference String 抽換掉，將新的新的 Page Reference String 插入 queue 最前面。

丁、MFU + FIFO

大致與 LFU + FIFO 相同，僅需修改 queue 已滿(Page Replaces)，判斷條件使用 MFU + FIFO，先找 count 最大的 Page Reference String，如果一樣就找最先進 queue 的 Page Reference String

戊、LFU + LRU

己、大致與 LFU + FIFO 相同，僅需修改兩部分。第一，沒有發生 Page_fault 時，代表 queue 內已有該值，要將 Page Reference String 更新，同時 count 次數加一。第二，queue 已滿(Page Replaces)，判斷條件使用 LFU + LRU，先找 count 最小的 Page Reference String，如果一樣就找最不常被使用的 Page Reference String。

3. 分析不同方法之間比較

（以下為實驗結果 input2.txt）

甲、Page Fault 次數

所有方法在超過 Page Frame 6 之後次數皆不在減少，維持在 Page Fault 次數 6

Page Frame 越大越不容易發生 Page Fault，但是有一定上限

乙、Page Replace 次數

所有方法在超過 Page Frame 6 之後次數皆不在減少，維持在 Page Replace 次數 0

Page Frame 越大越不容易發生 Page Replace

Page Fault	Page Replaces	Page Frames
20	19	1
15	13	2
15	12	3
10	6	4
9	4	5
6	0	6
6	0	7
6	0	8
6	0	9
6	0	10

Page Fault	Page Replaces	Page Frames
20	19	1
17	15	2
12	9	3
10	6	4
7	2	5
6	0	6
6	0	7
6	0	8
6	0	9
6	0	10

FIFO 實驗結果

Page Fault	Page Replaces	Page Frames
20	19	1
15	13	2
13	10	3
9	5	4
7	2	5
6	0	6
6	0	7
6	0	8
6	0	9
6	0	10

LRU 實驗結果

Page Fault	Page Replaces	Page Frames
20	19	1
15	13	2
15	12	3
12	8	4
8	3	5
6	0	6
6	0	7
6	0	8
6	0	9
6	0	10

LFU+FIFO 實驗結果

Page Fault	Page Replaces	Page Frames
20	19	1
15	13	2
11	8	3
9	5	4
7	2	5
6	0	6
6	0	7
6	0	8
6	0	9
6	0	10

MFU+FIFO 實驗結果

LFU+LRU 實驗結果

4. 結果與討論

（以下為實驗結果 input1_method6.txt）

甲、實驗數據與發現[畢雷笛反例]

乙、FIFO 與 MFU+FIFO 會發生畢雷笛反例

（對照下圖）

當 Page Frame = 4 時， page fault = 9

當 Page Frame = 4 時， page fault = 10 [Page Fault 不減反增]

畢雷笛反例:所以指的是增加 Page Frame 反而造成更多的 page fault 和 page Replace

Page Fault	Page Replaces	Page Frames
12	11	1
9	6	2
10	6	3
5	0	4
5	0	5
5	0	6
5	0	7
5	0	8
5	0	9
5	0	10

FIFO 實驗結果

Page Fault	Page Replaces	Page Frames
12	11	1
12	10	2
10	7	3
8	4	4
5	0	5
5	0	6
5	0	7
5	0	8
5	0	9
5	0	10

LRU 實驗結果

Page Fault	Page Replaces	Page Frames
12	11	1
12	10	2
10	7	3
8	4	4
5	0	5
5	0	6
5	0	7
5	0	8
5	0	9
5	0	10

LFU+FIFO 實驗結果

Page Fault	Page Replaces	Page Frames
12	11	1
12	10	2
9	6	3
10	6	4
5	0	5
5	0	6
5	0	7
5	0	8
5	0	9
5	0	10

MFU+FIFO 實驗結果

Page Fault	Page Replaces	Page Frames
12	11	1
12	10	2
10	7	3
8	4	4
5	0	5
5	0	6
5	0	7
5	0	8
5	0	9
5	0	10

LFU+LRU 實驗結果