作業系統-書面報告

資訊三甲 11127138 林雨臻

指導老師：鍾武君

1. **開發環境：**
2. **硬體設備**：：

* 處理器 (CPU)：11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz
* 記憶體 (RAM)：16 GB
* 作業系統：Windows

1. **開發工具**：

* 編譯器：g++ (版本 12.2.0)
* 程式編輯器：vscode (Visual Studio Code)

1. **實驗方法和流程**
   1. FIFO\_Algorithm:

* 說明：當page frame已滿且需要替換時，移除佇列中的第一個進入的page。
* 資料結構：
  1. queue<int> pageQueue; // page frame
  2. unordered\_map<int, bool> Page\_Records; // page -> isExist
* 實作方法：
  1. 如果page不在frame中：
     1. pageFaults++
     2. 如果frame已滿，移除 pageQueue 的第一個page，並在 Page\_Records 中刪除此page。
     3. 移除選定page，重設其資料，並pageReplaces++。
     4. 將新page加入 pageQueue 並在 Page\_Records 中記錄。
  2. 如果頁面已存在於frame中：
     1. 保持原先排序。
  3. LRU\_Algorithm:
* 說明：當page frame已滿且需要替換時，移除佇列中最久未使用的page。
* 資料結構：

1. unordered\_map<int, int> pagePosition; // page -> timestamp

* 實作方法：
  1. 如果page不在frame中：

1. pageFaults++
2. 如果frame滿了，使用 find\_oldest() 找出最近最久未使用的page並將其移除。
3. 在frame中插入新page並記錄當前時間戳。
   1. 如果頁面已存在於frame中：
      1. 更新此page的時間戳。
      2. 依照最近使用的時間排序。
   2. LFU\_and\_FIFO\_Algorithm:

* 說明：當page frame已滿且需要替換時，移除佇列中frequency最少的，若frequency相等，則移除最先進入佇列的page，並將frequency歸零等待下次進入重新計算。
* 資料結構：

1. unordered\_map<int, page\_record> pageRecords; // page -> (in\_page\_frame\_flag, count, timestamp)
2. set<pair<int, pair<int, int>>, Compare> pageFrames\_data; // <page, <count, timestamp>>

* 實作方法：

1. 如果page不在frame中：
   1. pageFaults++
   2. 如果frame滿了，使用find\_least\_frequent() 找出使用次數最少且最早進入frame的page。
   3. 從 pageFrames\_data 中移除選中的page，重設其資料，並pageReplaces++。
   4. 將新的page加入frame，更新其使用次數為 1 並記錄當前時間戳。
2. 如果頁面已存在於frame中：
3. 更新此page的時間戳。
4. 在 pageFrames\_data 中移除舊資料，插入更新後的page資料。
   1. MFU\_and\_FIFO\_Algorithm:

* 說明：當page frame已滿且需要替換時，移除佇列中frequency最多的，若frequency相等，則移除最先進入佇列的page，並將frequency歸零等待下次進入重新計算。
* 資料結構：

1. unordered\_map<int, page\_record> pageRecords; // page -> (in\_page\_frame\_flag, count, timestamp)
2. set<pair<int, pair<int, int>>, Compare> pageFrames\_data; // <page, <count, timestamp>>

* 實作方法：

1. 如果page不在frame中：
   1. pageFaults++
   2. 如果frame已滿，使用 find\_most\_frequent() 尋找使用次數最多的page。
   3. 從 pageFrames\_data 中移除選中的page，重設其資料、pageReplaces++。
   4. 將新page加入frame，更新其資料。
2. 如果頁面已存在於frame中：
   1. 更新page的使用次數。
   2. 在 pageFrames\_data 中移除舊資料，插入更新後的page資料
   3. LFU\_and\_LRU\_Algorithm:

* 說明：當page frame已滿且需要替換時，移除佇列中frequency最少的，若frequency相等，移除佇列中最久未使用的page。
* 資料結構：

1. unordered\_map<int, page\_record> pageRecords; // page -> (in\_page\_frame\_flag, count, timestamp)
2. set<pair<int, pair<int, int>>, Compare> pageFrames\_data; // <page, <count, timestamp>>

* 實作方法：

1. 如果page不在frame中：
   * 1. 增加page錯誤計數 (pageFaults++)
     2. 如果frame已滿，呼叫 find\_least\_frequent\_and\_recently() 尋找最少使用且最久未使用的page。
     3. 移除選定page，重設其資料，並pageReplaces++。
     4. 將新page加入frame，更新其資料。
2. 如果頁面已存在於框中：
   * 1. 更新page的使用次數和時間戳。
     2. 為了使輸出順序與老師一致，在 pageFrames\_data 中移除舊page，插入更新後的page。
3. 探討結果和原因
   1. **畢雷笛反例(Belady’s Anomaly)：  
      定義：**某些頁面替換算法中，當page frames增加時，page fault反而增多的現象。通常發生在FIFO（First In First Out）替換方法中，因為它只考慮page進入的先後順序，忽略了page使用的頻率或時間。  
      在 FIFO中，增加page frames數量通常會預期減少page faults，因為有更多的空間可以存放page。但從下方兩張圖可以看出，使用FIFO時，當page frames從 3 增加到 4 ，page fault反而從 9 增加到 10。

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

* 1. **比較五種方法間pageFaults、pageReplace的差異 (**page 輸入：123412512345)**：**

|  |  |  |
| --- | --- | --- |
| 方法 (Page Frames = 3) | Page Faults | Page Replaces |
| FIFO | 9 | 6 |
| LRU | 10 | 7 |
| LFU + FIFO | 10 | 7 |
| MFU + FIFO | 9 | 6 |
| LFU + LRU | 10 | 7 |

(表一)

|  |  |  |
| --- | --- | --- |
| 方法 (Page Frames = 4) | Page Faults | Page Replaces |
| FIFO | 10 | 6 |
| LRU | 8 | 4 |
| LFU + FIFO | 8 | 4 |
| MFU + FIFO | 10 | 6 |
| LFU + LRU | 8 | 4 |

(表二)

* + 1. FIFO：因為不考慮page的使用頻率與時間，pageFaults 和 pageReplaces 相對較高。
    2. LRU：尋找最近沒使用的page進行替換，適合每種page與任務執行階段高度關聯的情況，如此才可降低替換的發生次數。
    3. LFU + FIFO：結合使用頻率與先進先出，能有效處理常用page，但在每個page的頻率變化大的情況下性能可能下降，因為仍會頻繁發生置換。
    4. MFU + FIFO：認為頻率高的page可能已經使用完畢，但我認為此法並不特別符合真實場景，就我的常理而言除非每種page都有限制相同的總數，否則我會認為目前出現次數最高的page，未來的出現次數也會更高。
    5. LFU + LRU：綜合考慮使用頻率與最近使用時間，最佳方案。

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 數字, 陳列 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述一張含有 文字, 螢幕擷取畫面, 數字, 行 的圖片

自動產生的描述