

學號:F74074122

姓名:歐禮寬

系級:資訊 111

開發環境:

OS: Windows Subsystem for Linux Ubuntu 18.04.1

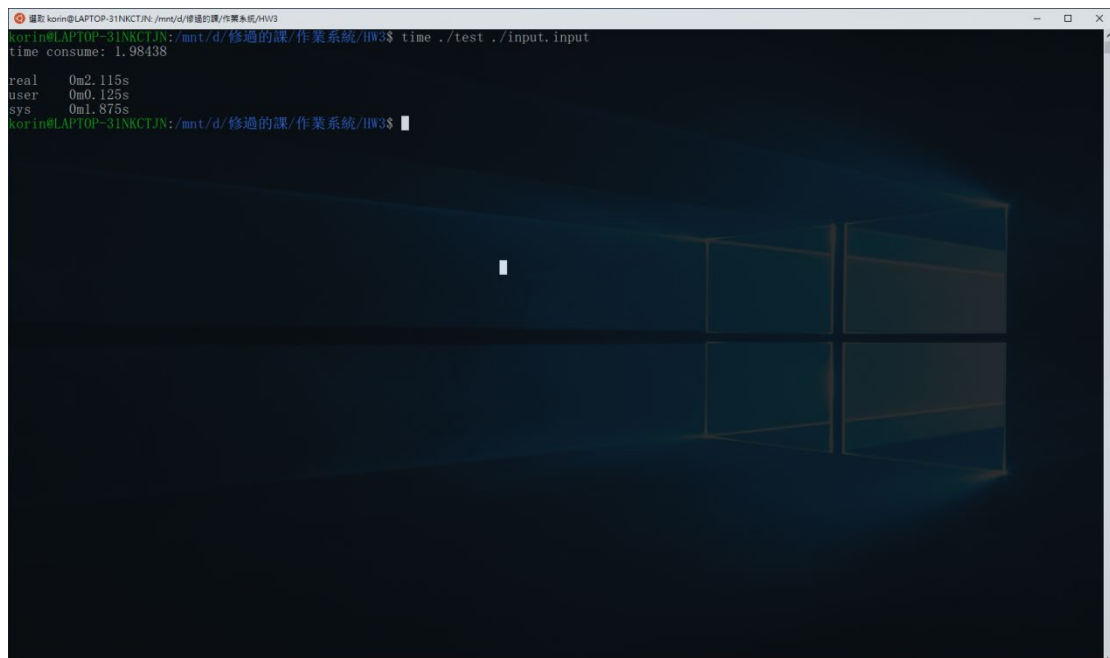
CPU: Intel® Core™ i5-8250U CPU @ 1.60GHz

Memory: 8GB

Programming Language: C++ gcc version 7.4.0

程式執行時間:

跑助教給的測資約 2 秒



```
korin@LAPTOP-31NKCTJN:/mnt/d/修過的課/作業系統/HW3$ time ./test ./input.input
time consume: 1.98438
real    0m2.115s
user    0m0.125s
sys     0m1.875s
korin@LAPTOP-31NKCTJN:/mnt/d/修過的課/作業系統/HW3$
```

程式開發與使用說明:

程式會依序執行每行指令(PUT、GET、SCAN)。

PUT 指令會檢查 memory 當中有沒有相同的 key，有的話就更新 value，沒有的話將 key-value 存進第一塊 memory(struct keyvalue)，當 memory 到一數值時，就將資料存進 disk(檔案)，並排序檔案內的資料，使得後續對 disk 資料操作能更有效率；GET 指令會先去 memory 找資料，沒有的話再去 disk 找資料，找到的資料我會存在第二塊 memory，後續要是要用到就可以直接從這裡拿，不須讀檔；SCAN 指令一樣先去 memory 資料，沒有的話再去 disk 找資料，找到的資料存在第三塊 mememory，因為 SCAN 找的資料都是連續的，而我的檔案都排序過，所以讀出來的資料也都是連續的，所以 SCAN 的後幾筆資料都只要一直找第三塊 memory 的下一筆資料。執行完指令後，程式會將 memory 尚未寫入 disk 的資料寫進去，好讓下一次執行能取得上一次的資料在結束。

編譯: g++ keyvaluestore.cpp

執行: ./a.out ./1.input

效能分析報告:

整理資料作法及分析:

我整理資料的做法是依據數字的最高位數(0~9)，各有一塊相對應的 memory 空間去存取，並用 C++的 map 輔助以增加更新某個 key 的速度，而存入 disk 前，我會用 HW1 用過的 external sort 對資料做排序，再存入 disk，雖然會增加時間，卻能對後續資料的取得有幫助，像是能使用 binary search 快速的找出某一筆 key；再來就是排序後的資料會有連續的特性，而 SCAN 就是 GET 很多連續的資料，這時我只要找到第一筆資料，之後的資料就只要看我找到的那一筆資料的下面幾筆就好了。

程式執行下:

以下 4 張圖為執行 PUT 的情況，CPU 會忽高忽低我覺得應該是要寫入 disk 時，CPU 會 idle 的原因，而磁碟使用料也是忽高忽低，因為只有 memory 到一定量才會去寫檔，而從記憶體也可以看出，程式一開始 malloc 的記憶體好像要等到用到時，作業系統才會真的給我用。

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	26% CPU	68% 記憶體	6% 磁碟	0% 網路	0% GPU	GPU 3
a.out		23.2%	1,199.0 ...	11.4 MB/...	0 Mbps	0%	
Discord (32 位元)		0.7%	77.4 MB	0 MB/秒	0 Mbps	0%	
> 工作管理員		0.7%	23.8 MB	0 MB/秒	0 Mbps	0%	
> Visual Studio Code (...)		0.7%	276.1 MB	0 MB/秒	0 Mbps	0%	GPU
> Windows 檔案總管		0.4%	86.3 MB	0 MB/秒	0 Mbps	0%	
桌面視窗管理員		0.4%	47.5 MB	0.1 MB/秒	0 Mbps	0.2%	GPU
System		0.3%	0.1 MB	5.4 MB/秒	0 Mbps	0%	
Avast Antivirus		0%	14.2 MB	0 MB/秒	0 Mbps	0%	
Avast Antivirus engin...		0%	32.0 MB	0 MB/秒	0 Mbps	0%	
系統插斷		0%	0 MB	0 MB/秒	0 Mbps	0%	
CTF 載入程式		0%	4.8 MB	0 MB/秒	0 Mbps	0%	
> 服務主機: 已連線的裝...		0%	6.7 MB	0.1 MB/秒	0 Mbps	0%	
> Avast Software Analy...		0%	49.7 MB	0 MB/秒	0 Mbps	0%	
> Microsoft Text Input ...		0%	2.4 MB	0 MB/秒	0 Mbps	0%	
用戶端伺服器執行階段...		0%	1.2 MB	0 MB/秒	0 Mbps	0.1%	GPU
< >							
⬆ 較少詳細資料(D)						結束工作(E)	

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	24% CPU	78% 記憶體	33% 磁碟	0% 網路	1% GPU	GPU 3
a.out		16.0%	1,404.1 ...	60.1 MB/...	0 Mbps	0%	
> 剪取與繪圖 (3)	🟢	1.5%	41.6 MB	0.1 MB/秒	0 Mbps	0%	GPU
> 工作管理員		1.5%	23.8 MB	0 MB/秒	0 Mbps	0%	
桌面視窗管理員		1.2%	62.9 MB	0.1 MB/秒	0 Mbps	0.7%	GPU
System		0.9%	0.1 MB	7.2 MB/秒	0 Mbps	0%	
> Visual Studio Code (...)		0.7%	276.8 MB	0 MB/秒	0 Mbps	0%	GPU
用戶端伺服器執行階段...		0.5%	1.2 MB	0 MB/秒	0 Mbps	0.1%	GPU
> 服務主機: 已連線的裝...		0.4%	6.7 MB	0.1 MB/秒	0 Mbps	0%	
系統插斷		0.2%	0 MB	0 MB/秒	0 Mbps	0%	
Avast Antivirus		0.2%	14.2 MB	0 MB/秒	0 Mbps	0%	
> Windows 檔案總管		0.1%	87.3 MB	0.1 MB/秒	0 Mbps	0%	
CTF 載入程式		0.1%	4.9 MB	0.1 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.1%	77.9 MB	0.1 MB/秒	0 Mbps	0%	
WMI Provider Host		0%	3.2 MB	0 MB/秒	0 Mbps	0%	
> 服務主機: 網路服務		0%	3.2 MB	0 MB/秒	0 Mbps	0%	
< >							
⬆ 較少詳細資料(D)						結束工作(E)	

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	14% CPU	77% 記憶體	25% 磁碟	0% 網路	0% GPU	GPU 3
a.out		11.8%	1,404.1 ...	55.2 MB/...	0 Mbps	0%	
> 工作管理員		0.8%	24.3 MB	0.1 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.4%	78.4 MB	0 MB/秒	0 Mbps	0%	
桌面視窗管理員		0.3%	75.0 MB	0.1 MB/秒	0 Mbps	0.1%	GPU
> Visual Studio Code (...)		0.3%	278.7 MB	0 MB/秒	0 Mbps	0%	GPU
系統插斷		0.2%	0 MB	0 MB/秒	0 Mbps	0%	
System		0.1%	0.1 MB	0.2 MB/秒	0 Mbps	0%	
> NService Application...		0.1%	1.4 MB	0 MB/秒	0 Mbps	0%	
> Windows 檔案總管		0%	87.2 MB	0.1 MB/秒	0 Mbps	0%	
> 服務主機: 已連線的裝...		0%	6.8 MB	0.1 MB/秒	0 Mbps	0%	
Discord (32 位元)		0%	5.6 MB	0 MB/秒	0 Mbps	0%	
> 服務主機: 遠端程序呼...		0%	9.7 MB	0 MB/秒	0 Mbps	0%	
Google Chrome		0%	239.3 MB	0 MB/秒	0 Mbps	0%	
> 服務主機: DCOM 伺服...		0%	20.2 MB	0 MB/秒	0 Mbps	0%	
> 剪取與繪圖 (3)		0%	34.8 MB	0.1 MB/秒	0 Mbps	0%	GPU
< >							
⬆ 較少詳細資料(D)						結束工作(E)	

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	38% CPU	79% 記憶體	36% 磁碟	0% 網路	17% GPU	GPU 3
a.out		6.5%	1,404.1 ...	43.4 MB/...	0 Mbps	0%	
> Windows 檔案總管		5.5%	87.8 MB	0.1 MB/秒	0 Mbps	0%	
> 剪取與繪圖 (3)	🟢	5.5%	63.2 MB	0.1 MB/秒	0 Mbps	0.3%	GPU
桌面視窗管理員		3.7%	87.5 MB	0.2 MB/秒	0 Mbps	16.2%	GPU
Windows Audio Devi...		2.7%	232.5 MB	0 MB/秒	0 Mbps	0%	
Application Frame H...		2.4%	17.5 MB	0 MB/秒	0 Mbps	0.1%	
System		2.4%	0.1 MB	0.2 MB/秒	0 Mbps	0%	
> 工作管理員		1.4%	24.3 MB	0 MB/秒	0 Mbps	0%	
> Windows 殼層體驗主機		1.0%	67.5 MB	0.1 MB/秒	0 Mbps	0%	GPU
> Visual Studio Code (...)		0.9%	279.3 MB	0 MB/秒	0 Mbps	0%	GPU
> 服務主機: 已連線的裝...		0.9%	6.8 MB	0.1 MB/秒	0.1 Mbps	0%	
> 服務主機: 遠端程序呼...		0.8%	9.6 MB	0 MB/秒	0 Mbps	0%	
Shell Infrastructure H...		0.7%	7.9 MB	0 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.6%	78.7 MB	0 MB/秒	0 Mbps	0%	
用戶端伺服器執行階段...		0.6%	1.2 MB	0 MB/秒	0 Mbps	0.7%	GPU

以下兩張圖為 GET 的情況，CPU 使用量很高，磁碟使用量很低，因為我 GET 是用 binary search 整的檔案讀出一筆資料，所以我覺得正好符合這種情況。

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	30% CPU	77% 記憶體	3% 磁碟	0% 網路	1% GPU	GPU 3
a.out		26.0%	686.9 MB	0.1 MB/秒	0 Mbps	0%	
> 工作管理員		1.1%	23.1 MB	0 MB/秒	0 Mbps	0%	
桌面視窗管理員		1.1%	71.8 MB	0.1 MB/秒	0 Mbps	0.9%	GPU
> 服務主機: 已連線的裝...		0.4%	6.8 MB	0.1 MB/秒	0.1 Mbps	0%	
> 剪取與繪圖 (3)	🟢	0.4%	41.5 MB	0.1 MB/秒	0 Mbps	0%	GPU
System		0.3%	0.1 MB	0.1 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.1%	77.5 MB	0 MB/秒	0 Mbps	0%	
> Avast Software Analy...		0.1%	50.1 MB	0 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.1%	8.1 MB	0 MB/秒	0 Mbps	0%	
> Windows 檔案總管		0.1%	89.6 MB	0.1 MB/秒	0 Mbps	0%	
Registry		0.1%	3.3 MB	0.1 MB/秒	0 Mbps	0%	
系統掛斷		0.1%	0 MB	0 MB/秒	0 Mbps	0%	
> Windows 殼層體驗主機		0%	54.7 MB	0.1 MB/秒	0 Mbps	0%	GPU
> 服務主機: 剪貼簿使用...		0%	2.5 MB	0 MB/秒	0 Mbps	0%	
> Microsoft Windows S...		0%	12.9 MB	0.1 MB/秒	0 Mbps	0%	
< >							
⬆ 較少詳細資料(D)						結束工作(E)	



工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	35% CPU	74% 記憶體	2% 磁碟	0% 網路	0% GPU	GPU 3
a.out		26.4%	686.9 MB	0.1 MB/秒	0 Mbps	0%	
System		1.5%	0.1 MB	3.4 MB/秒	0 Mbps	0%	
> Runtime Broker		1.4%	4.0 MB	0.3 MB/秒	0 Mbps	0%	
> 服務主機: DCOM 伺服...		0.7%	20.9 MB	0 MB/秒	0 Mbps	0%	
> Windows 檔案總管		0.7%	89.2 MB	0 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.7%	76.6 MB	0 MB/秒	0 Mbps	0%	
> Avast Software Analy...		0.7%	50.1 MB	0.1 MB/秒	0 Mbps	0%	
Shell Infrastructure H...		0.6%	8.3 MB	0 MB/秒	0 Mbps	0%	
> 工作管理員		0.6%	23.1 MB	0.1 MB/秒	0 Mbps	0%	
> Microsoft 內容		0.5%	3.3 MB	0.1 MB/秒	0 Mbps	0%	
> Visual Studio Code (...)		0.4%	301.3 MB	0 MB/秒	0 Mbps	0%	GPU
桌面視窗管理員		0.3%	52.4 MB	0 MB/秒	0 Mbps	0.2%	GPU
> 服務主機: 遠端程序呼...		0.3%	10.0 MB	0 MB/秒	0 Mbps	0%	
> Microsoft Windows S...		0.1%	12.8 MB	0.1 MB/秒	0 Mbps	0%	
> 服務主機: SSDP Disco...		0.1%	1.0 MB	0 MB/秒	0 Mbps	0%	

以下兩張圖為 SCAN 的情況 CPU 使用率一樣很高，而我原本以為磁碟使用率應該會很，因為我 SCAN 的時候是寫成讓他讀 disk 裡的一段資料而非 GET 的一筆，但從圖中可以看出其實並沒有很高。

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	32% CPU	86% 記憶體	3% 磁碟	0% 網路	0% GPU	GPU 3
a.out		26.4%	1,373.6 ...	0.1 MB/秒	0 Mbps	0%	
> 服務主機: SysMain		2.6%	1.1 MB	0 MB/秒	0 Mbps	0%	
> Google Chrome (13)		0.7%	608.4 MB	0.1 MB/秒	0 Mbps	0%	
Windows Audio Devi...		0.6%	25.2 MB	0 MB/秒	0 Mbps	0%	
> 工作管理員		0.4%	22.6 MB	0 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.4%	76.3 MB	0 MB/秒	0 Mbps	0%	
> 服務主機: DCOM 伺服...		0.3%	20.4 MB	0 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.2%	7.3 MB	0 MB/秒	0 Mbps	0%	
服務及控制站應用程式		0.2%	3.0 MB	0 MB/秒	0 Mbps	0%	
> 服務主機: Diagnostic ...		0.2%	15.3 MB	0.1 MB/秒	0 Mbps	0%	
> Avast Software Analy...		0.1%	53.9 MB	0 MB/秒	0 Mbps	0%	
系統擷斷		0.1%	0 MB	0 MB/秒	0 Mbps	0%	
> NService Application...		0%	1.8 MB	0 MB/秒	0 Mbps	0%	
> Visual Studio Code (...)		0%	394.8 MB	0 MB/秒	0 Mbps	0%	GPU
Avast Antivirus		0%	22.8 MB	0 MB/秒	0 Mbps	0%	
< >							
⬆ 較少詳細資料(D)						結束工作(E)	

工作管理員							
檔案(F) 選項(O) 檢視(V)							
處理程序 效能 應用程式歷程記錄 開機 使用者 詳細資料 服務							
名稱	狀態	30% CPU	81% 記憶體	2% 磁碟	0% 網路	0% GPU	GPU 3
a.out		25.1%	1,373.6 ...	0.1 MB/秒	0 Mbps	0%	
> Google Chrome (13)		1.7%	618.4 MB	0 MB/秒	0 Mbps	0%	
> 工作管理員		0.6%	22.6 MB	0 MB/秒	0 Mbps	0%	
Discord (32 位元)		0.5%	78.5 MB	0 MB/秒	0 Mbps	0%	
Windows Audio Devi...		0.4%	25.3 MB	0 MB/秒	0 Mbps	0%	
System		0.3%	0.1 MB	0.1 MB/秒	0 Mbps	0%	
Google Chrome		0.3%	15.9 MB	0.2 MB/秒	0 Mbps	0%	
> Avast Service		0.3%	33.1 MB	0.1 MB/秒	0 Mbps	0%	
> Visual Studio Code (...)		0.3%	434.1 MB	0 MB/秒	0 Mbps	0%	GPU
桌面視窗管理員		0.3%	66.5 MB	0 MB/秒	0 Mbps	0.1%	GPU
> Windows 檔案總管		0.1%	75.7 MB	0 MB/秒	0 Mbps	0%	
Avast Antivirus engin...		0.1%	39.0 MB	0.1 MB/秒	0 Mbps	0%	
> NService Application...		0.1%	1.8 MB	0 MB/秒	0 Mbps	0%	
用戶端伺服器執行階段...		0.1%	0.5 MB	0 MB/秒	0 Mbps	0.1%	GPU
系統插斷		0.1%	0 MB	0 MB/秒	0 Mbps	0%	

改進程式的過程:

PUT 資料到 memory 時,如果 key 存在要跟新資料,而如判斷 key 存不存在,我是用遍歷的方式判斷,發隨著 memory 的增大,速度會下降許多,所以後來新增一個 C++的 map 來記錄 key 是否存在,藉此提升查詢的速度。

GET 資料時,如果資料不再 memory 而要去找 disk,

我最一開始是從檔案開頭一直讀一段很大的資料到記憶體，再用資料的最小值及最大值判斷key是否存在，存在的話再用 binary search 的方式找 key，但當檔案很大的時且資料又在檔案後半部，最非常的耗時間，因為要一直讀取，所以後來改成對 disk 做 binary search，所以每次就只要讀檔案中的一筆資料且每次都能保證找完檔案的一半，這樣就能減少讀取花費的時間及找尋的次數。

SCAN 資料時我最一開始是直接當作多次 GET，然後每次去找 memory 及 disk，但因為我檔案裡的 key 都排序過，所以當我找到 SCAN 的第一筆資料，且將他後面的一段資料也都讀出來，下次就不用再讀一次檔案，只要記住上一筆資料的位置，然後判斷他的下一筆資料是否是要找的 key 值就好了。