HW4 buffer 與效能

408410019 資工二 徐怡娟

基本設定

(1) Buffer 長度輸入-1 時，預設為 1024bytes 大小

(2) Tab 有些佔 8、4、2 格，程式內一律視為 8 格，保證每行必定少於 80 個字

(3) 可以處理 Unicode 與 wchar

須加上 #include <locale.h>、setlocale(LC_CTYPE,"C.UTF-8"); 才能讓 wchar 函式在程式中作用，否則仍然無法正常運作。

1. Buffer size 與執行速度

秒	0	-1	4KB	16KB	64KB	1MB	8MB
Real time	14.47	1.18	0.49	0.47	0.48	0.49	0.49
User	5.32	0.60	0.42	0.42	0.42	0.43	0.43
System	9.13	0.58	0.70	0.05	0.05	0.06	0.05

2. 使用 ltrace 觀察函式呼叫情況

測試了好幾次，每次都是 setlocale 佔所有 function call 時間最久

```
accepted@ubuntu:~/system-programming/ch05/hw4$ ltrace -c ./fileperf input.txt testOut.txt -1
% time     seconds  usecs/call     calls      function
------ ----------- ----------- --------- --------------------
 46.57    0.004970        4970         1 setlocale
 14.35    0.001531        1531         1 fputc
 12.26    0.001308        1308         1 fopen
  9.45    0.001008        1008         1 fprintf
  7.23    0.000772         772         1 getwc
  6.23    0.000665         665         1 atoi
  2.95    0.000315         315         1 setvbuf
  0.97    0.000103         103         1 fclose
------ ----------- ----------- --------- --------------------
100.00    0.010672                     8 total
```

3. 使用 strace 觀察 system call 的情況

這個程式都是由 lib C 的 function 來控制檔案讀寫，在 lib C 裡做很多優化的工作讓他比單純直接用 read/write 等 system call function 效率還高可由 ltrace 檢視，read 因為有 buffer 的關係不需要遇到一個字一個字讀，因此 call 的次數相較於 write 少很多(write 在程式裡讀到完整的單字就輸出一次)，裡面有用到上課提到的 mmap 直接在 RAM 配一塊記憶體給 file，讓作業系統能直接從 RAM 操作在 Disk 的 file，提升整體效率。

```
accepted@ubuntu:~/system-programming/ch05/hw4$ strace -c ./fileperf input.txt testOut.txt -1
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 98.19    1.145684           3    324857           write
  0.93    0.010821        1352         8           close
  0.88    0.010288           3      3349           read
  0.00    0.000000           0         8           fstat
  0.00    0.000000           0        10           mmap
  0.00    0.000000           0         4           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         3           brk
  0.00    0.000000           0         6           pread64
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         2         1 arch_prctl
  0.00    0.000000           0         8           openat
------ ----------- ----------- --------- --------- ----------------
100.00    1.166793              328258         2 total
```

4. 分析 function call 和 system call 的成本差異

上方截圖的 usecs/call 欄位為函式單次呼叫所花費的時間，整體看起來每次 function call 花費的時間比 system call 多，但實際運作起來 function call 會配合 3.中所說明的一些方法整合其他 system call 或利用 buffer 的方式讓最後得到的整體效率提高。