OS LAB 1

130050001 : Ghurye Sourabh Sunil
130050037 : Utkarsh Mall

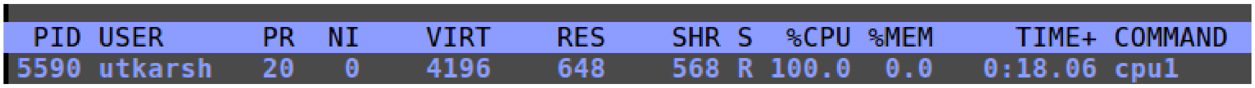
Q1.

Machine has 4 CPU cores. #Command used : cat cpuinfo | grep "cpu cores"
Total Memory : 8146916 kB. Fraction free : 0.334 #Command used : cat meminfo
Context switches : 3182071 #Command used : cat stat
No. of forks : 4825 #Command used : cat stat

Q2.

cpu1:

CPU is bottleneck resource
Command used: top
top command showed 100% CPU usage by this process.



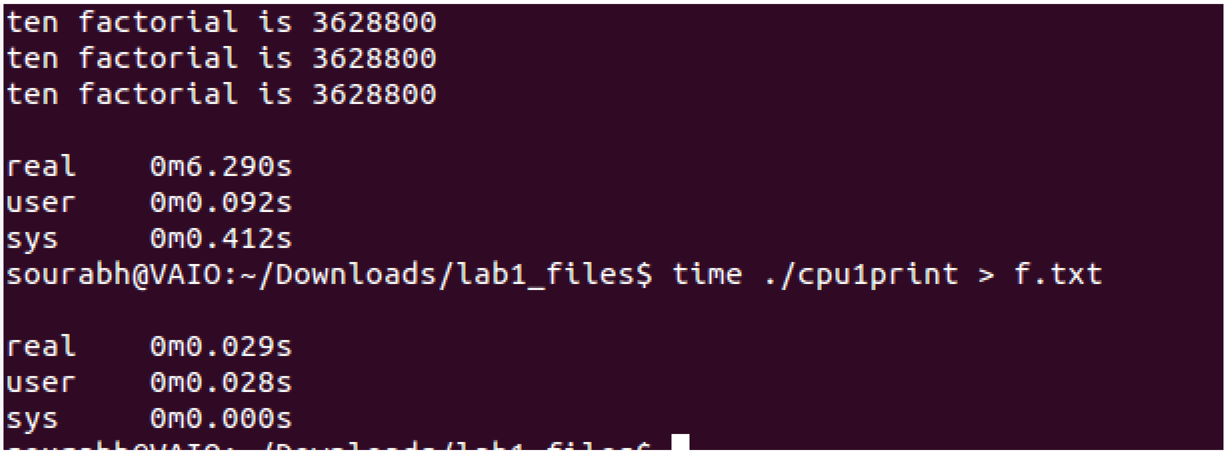
```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 5590 utkarsh   20   0    4196    648    568 R 100.0  0.0   0:18.06 cpu1
```

Since the program uses only 4-5 variables in its execution, the execution requires solely the CPU.

cpu1print:

console buffer size is the bottleneck resource.

Experiment : compare with printing to file by restricting the loop to 10000 iterations. Time reduces significantly.



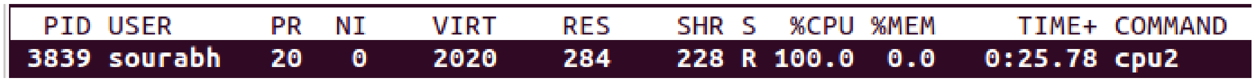
```
ten factorial is 3628800
ten factorial is 3628800
ten factorial is 3628800

real	0m6.290s
user	0m0.092s
sys	0m0.412s
sourabh@VAIO:~/Downloads/lab1_files$ time ./cpu1print > f.txt

real	0m0.029s
user	0m0.028s
sys	0m0.000s
sourabh@VAIO:~/Downloads/lab1_files$
```

cpu2:

CPU is bottleneck resource

Command used: top

top command showed 100% CPU usage by this process.



```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 3839 sourabh   20   0    2020    284    228 R 100.0  0.0   0:25.78 cpu2
```

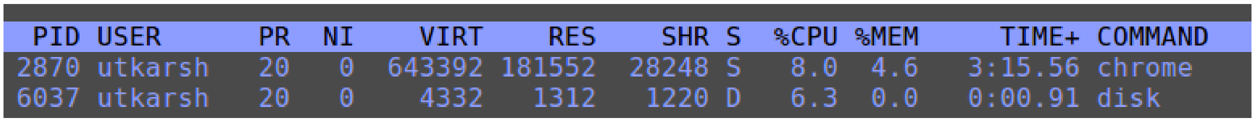
Even though it has “gettimeoftheday” syscall, he execution requires mostly the CPU.

disk:

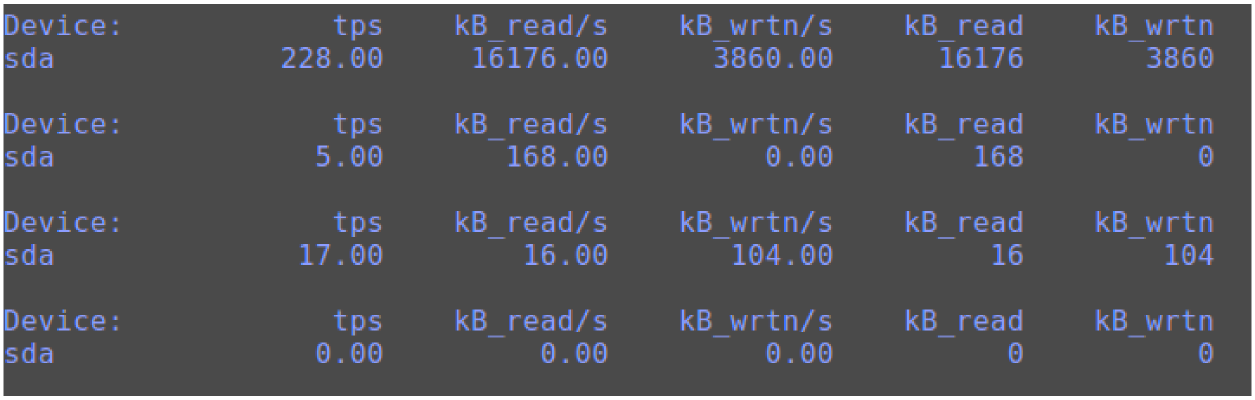
Disk bandwidth is bottleneck resource

Command used: top, iostat

top command showed a mere 6.3% CPU usage by this process. iostat showed significant change in disk I/O bandwidth when program was running and stopped.



```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 2870 utkarsh   20   0  643392 181552  28248 S   8.0  4.6   3:15.56 chrome
 6037 utkarsh   20   0    4332   1312   1220 D   6.3  0.0   0:00.91 disk
```



```
Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda             228.00     16176.00      3860.00      16176       3860

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               5.00       168.00         0.00        168          0

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda              17.00        16.00       104.00         16        104

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.00         0.00         0.00          0          0
```

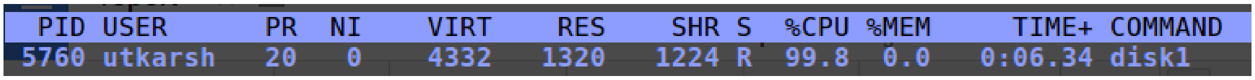
In every iteration a new file is required to be read by the program hence the OS is not able to cache new information in each iteration, due to which a significant disk I/O bandwidth is required by **disk**.

disk1:

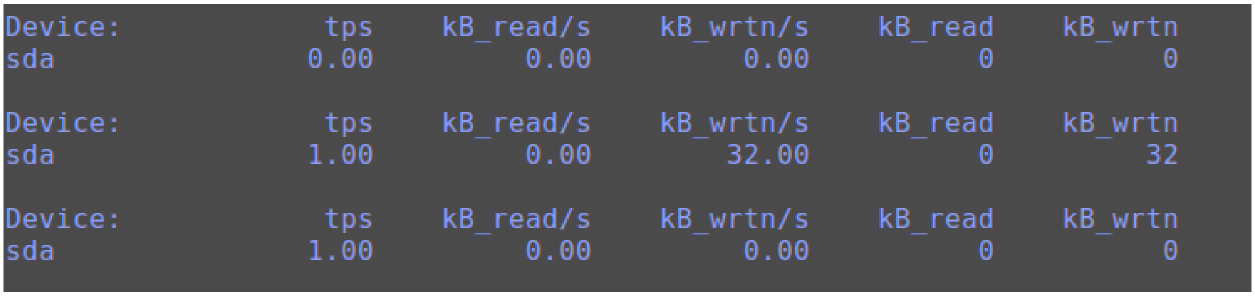
CPU is bottleneck resource

Command used: top, iostat

top command showed 99.8% CPU usage by this process. iostat did not show significant change in disk I/O bandwidth when program was running and stopped.



```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+  COMMAND
 5760 utkarsh   20   0    4332   1320   1224 R  99.8  0.0    0:06.34 disk1
```

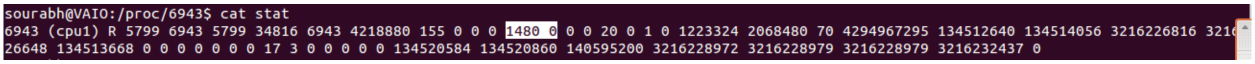


```
Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.00         0.00         0.00          0          0

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               1.00         0.00        32.00          0         32

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               1.00         0.00         0.00          0          0
```

disk1 reads only 1 file “foo0.txt” which will be stored in main memory/cache after first iteration hence disk I/O is not the bottleneck in this case compared to **disk** program.

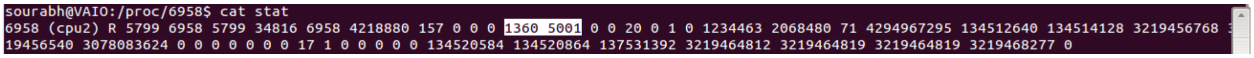
Q3.

cpu1 spends very small(approx 0) amount of time in kernel mode as there is no *syscall* to make



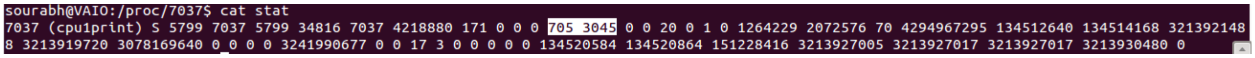
```
sourabh@VAIO:/proc/6943$ cat stat
6943 (cpu1) R 5799 6943 5799 34816 6943 4218880 155 0 0 0 1480 0 0 0 20 0 1 0 1223324 2068480 70 4294967295 134512640 134514056 3216226816 3216
26648 134513668 0 0 0 0 0 0 0 17 3 0 0 0 0 0 134520584 134520860 140595200 3216228972 3216228979 3216228979 3216232437 0
```

cpu2 spends more time in kernel mode because of the *syscall* corresponding to the “gettimeoftheday()”



```
sourabh@VAIO:/proc/6958$ cat stat
6958 (cpu2) R 5799 6958 5799 34816 6958 4218880 157 0 0 0 1360 5001 0 0 20 0 1 0 1234463 2068480 71 4294967295 134512640 134514128 3219456768 3
19456540 3078083624 0 0 0 0 0 0 0 17 1 0 0 0 0 0 134520584 134520864 137531392 3219464812 3219464819 3219464819 3219468277 0
```

cpu1print spends considerably more amount of time in kernel mode because the printf() has several *syscalls* corresponding to printing out, clearing a buffer, etc.



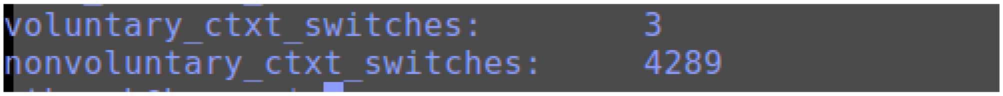
```
sourabh@VAIO:/proc/7037$ cat stat
7037 (cpu1print) S 5799 7037 5799 34816 7037 4218880 171 0 0 0 705 3045 0 0 20 0 1 0 1264229 2072576 70 4294967295 134512640 134514168 321392148
8 3213919720 3078169640 0 0 0 0 3241990677 0 0 17 3 0 0 0 0 0 134520584 134520864 151228416 3213927005 3213927017 3213927017 3213930480 0
```

Fraction of time spend in kernel mode
cpu1print(~81%) > cpu2(~75%) > cpu1(~0%)

Q4.

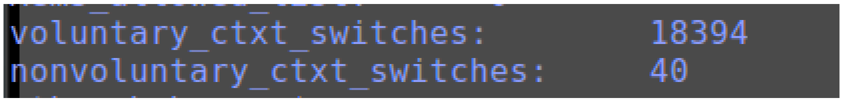
Command used: cat /proc/<pid>/status

The ratio of voluntary to involuntary context switches is much higher for program **disk** compared to **cpu1,** in fact while running **disk** most of the context switches occurring are voluntary.



```
voluntary_ctxt_switches:        3
nonvoluntary_ctxt_switches:     4289
```

context switch data for program **cpu1**



```
voluntary_ctxt_switches:        18394
nonvoluntary_ctxt_switches:     40
```

context switch data for program **disk**

Since **disk** has a file read operation in every of its iteration, it has a lot of disk I/Os hence it is voluntarily switching the context. Contrary to this, **cpu1** only requires processor for the execution hence the number of voluntary switches are very less.