COEN 241: Assignment 1

Name: Sanfer Samson Noronha
Student ID: W1652189

1. Installing QEMU and making QEMU disk image

a. Downloading Ubuntu ISO - Download LTS for ARM (apple silicon)
b. Installing QEMU - brew install qemu
c. Creating QEMU image - qemu-img create ubuntu.img 20G -f qcow2
d. Installing VM -
   Run:

```
qemu-system-aarch64 \
-accel hvf -cpu cortex-a57 -M virt,highmem=off -m 2G \
-smp 2 \
-drive file=/opt/homebrew/Cellar/qemu/6.2.0_1/share/qemu/edk2-aarch64-code.fd,if=pflash,format=raw,readonly=on \
-drive if=none,file=ubuntu.img,format=qcow2,id=hd0 \
-device virtio-blk-device,drive=hd0,serial="trial_2" \
-device virtio-net-device,netdev=net0 \
-netdev user,id=net0 \
-vga none -device ramfb \
-cdrom ubuntu-20.04.4-live-server-arm64.iso \
-device usb-ehci -device usb-kbd -device usb-mouse -usb -nographic
```

   To run your image, we use the same set of commands as above, we just remove the cdrom argument.

2. Experimental Setup

System: 2022 Macbook Air
Chip: M2

Testing 3 configurations:

a. 2 cores with 2GB memory allocation
b. 4 cores with 4GB memory allocation
c. 6 cores with 6GB memory allocation

3. Docker installation

Make sure Rosetta is installed in order to ensure instruction set compatibility.
Run: softwareupdate –install-rosetta

To install docker engine, simply download the docker dmg image from the following link

Test if docker is installed correctly:
docker run hello-world

a. QEMU and Docker Results

Config 1 : 2GB RAM and 2 cores

a. CPU testing
   Case: 2000

```
WARNING: the --test option is deprecated. You can pass a script name or path on the co
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 76870.86

General statistics:
    total time:                          30.0003s
    total number of events:              2306274

Latency (ms):
         min:                                    0.01
         avg:                                    0.01
         max:                                    0.14
         95th percentile:                        0.01
         sum:                                29830.54

Threads fairness:
    events (avg/stddev):           2306274.0000/0.00
    execution time (avg/stddev):   29.8305/0.00
```

Iteration 1

```
WARNING: the --test option is deprecated. You can pass a script name or path on the
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 77309.95

General statistics:
    total time:                          30.0002s
    total number of events:              2319373

Latency (ms):
         min:                                    0.01
         avg:                                    0.01
         max:                                    2.23
         95th percentile:                        0.01
         sum:                                29836.92

Threads fairness:
    events (avg/stddev):           2319373.0000/0.00
    execution time (avg/stddev):   29.8369/0.00
```

Iteration 2

```
WARNING: the --test option is deprecated. You can pass a script name or path on the
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 77536.54

General statistics:
    total time:                          30.0002s
    total number of events:              2326169

Latency (ms):
         min:                                    0.01
         avg:                                    0.01
         max:                                    0.38
         95th percentile:                        0.01
         sum:                                29832.03

Threads fairness:
    events (avg/stddev):           2326169.0000/0.00
    execution time (avg/stddev):   29.8320/0.00
```

Iteration 3

```
WARNING: the --test option is deprecated. You can pass a script name or path on th
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 77037.63

General statistics:
    total time:                          30.0002s
    total number of events:              2311203

Latency (ms):
         min:                                    0.01
         avg:                                    0.01
         max:                                    1.05
         95th percentile:                        0.01
         sum:                                29834.41

Threads fairness:
    events (avg/stddev):           2311203.0000/0.00
    execution time (avg/stddev):   29.8344/0.00
```

Iteration 4

```
WARNING: the --test option is deprecated. You can pass a script name or path on the
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 74888.92

General statistics:
    total time:                          30.0001s
    total number of events:              2246725

Latency (ms):
         min:                                    0.01
         avg:                                    0.01
         max:                                    0.09
         95th percentile:                        0.02
         sum:                                29837.47

Threads fairness:
    events (avg/stddev):           2246725.0000/0.00
    execution time (avg/stddev):   29.8375/0.00
```

Iteration 5

Table here:

Serial Number of Iterations	Events per second
1	76870
2	77309
3	77536
4	77037
5	77488

Docker:

```
└➜  ./DOCKER_2000_CPU.sh
WARNING: the --test option is deprecated. You can pass a script name or path
sysbench 1.0.20-f6f6117dc4 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 61850.10

General statistics:
    total time:                          30.0001s
    total number of events:              1855549

Latency (ms):
         min:                                    0.02
         avg:                                    0.02
         max:                                    0.30
         95th percentile:                        0.02
         sum:                                29861.91

Threads fairness:
    events (avg/stddev):           1855549.0000/0.00
    execution time (avg/stddev):   29.8619/0.00
```

Iteration 1

```
WARNING: the --test option is deprecated. You can pass a script name or path on
sysbench 1.0.20-f6f6117dc4 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 61929.69

General statistics:
    total time:                          30.0001s
    total number of events:              1857937

Latency (ms):
         min:                                    0.02
         avg:                                    0.02
         max:                                    0.93
         95th percentile:                        0.02
         sum:                                29862.16

Threads fairness:
    events (avg/stddev):           1857937.0000/0.00
    execution time (avg/stddev):   29.8622/0.00
```

Iteration 2

```
sysbench 1.0.20-f6f6117dc4 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

WARNING: the --test option is deprecated. You can pass a script name or path
Threads started!

CPU speed:
    events per second: 61344.60

General statistics:
    total time:                          30.0003s
    total number of events:              1840439

Latency (ms):
         min:                                    0.02
         avg:                                    0.02
         max:                                    1.04
         95th percentile:                        0.02
         sum:                                29860.28

Threads fairness:
    events (avg/stddev):           1840439.0000/0.00
    execution time (avg/stddev):   29.8603/0.00
```

Iteration 3

```
WARNING: the --test option is deprecated. You can pass a script name or path o
sysbench 1.0.20-f6f6117dc4 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 61354.63

General statistics:
    total time:                          30.0001s
    total number of events:              1840684

Latency (ms):
         min:                                    0.02
         avg:                                    0.02
         max:                                    0.34
         95th percentile:                        0.02
         sum:                                29861.39

Threads fairness:
    events (avg/stddev):           1840684.0000/0.00
    execution time (avg/stddev):   29.8614/0.00
```

Iteration 4

```
WARNING: the --test option is deprecated. You can pass a script name or path on
sysbench 1.0.20-f6f6117dc4 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 2000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 61540.98

General statistics:
    total time:                          30.0001s
    total number of events:              1846277

Latency (ms):
         min:                                    0.02
         avg:                                    0.02
         max:                                    0.43
         95th percentile:                        0.02
         sum:                                29859.74

Threads fairness:
    events (avg/stddev):           1846277.0000/0.00
    execution time (avg/stddev):   29.8597/0.00
```

Iteration 5

Table here

Serial Number of Iterations	Events per second
1	61850
2	61929
3	61344
4	61354
5	61540

Result: QEMU is faster than Docker in CPU performance

Case: 20000

```
WARNING: the --test option is deprecated. You can pass a script name or path on the
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  3967.12

General statistics:
    total time:                          30.0004s
    total number of events:              119021

Latency (ms):
         min:                                    0.25
         avg:                                    0.25
         max:                                    0.55
         95th percentile:                        0.26
         sum:                                29987.24

Threads fairness:
    events (avg/stddev):           119021.0000/0.00
    execution time (avg/stddev):   29.9872/0.00
```

Iteration 1

```
WARNING: the --test option is deprecated. You can pass a script name or path on the co
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  3952.41

General statistics:
    total time:                          30.0004s
    total number of events:              118576

Latency (ms):
         min:                                    0.25
         avg:                                    0.25
         max:                                    1.41
         95th percentile:                        0.27
         sum:                                29979.21

Threads fairness:
    events (avg/stddev):           118576.0000/0.00
    execution time (avg/stddev):   29.9792/0.00
```

Iteration 2

```
WARNING: the --test option is deprecated. You can pass a script name or path on the co
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  3942.26

General statistics:
    total time:                          30.0005s
    total number of events:              118273

Latency (ms):
         min:                                    0.25
         avg:                                    0.25
         max:                                    0.64
         95th percentile:                        0.27
         sum:                                29976.30

Threads fairness:
    events (avg/stddev):           118273.0000/0.00
    execution time (avg/stddev):   29.9763/0.00
```

Iteration 3

```
WARNING: the --test option is deprecated. You can pass a script name or path on the co
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  3967.00

General statistics:
    total time:                          30.0002s
    total number of events:              119013

Latency (ms):
         min:                                    0.25
         avg:                                    0.25
         max:                                    0.91
         95th percentile:                        0.26
         sum:                                29984.94

Threads fairness:
    events (avg/stddev):           119013.0000/0.00
    execution time (avg/stddev):   29.9849/0.00
```

Iteration 4

```
WARNING: the --test option is deprecated. You can pass a script name or path on the
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  3960.42

General statistics:
    total time:                          30.0005s
    total number of events:              118820

Latency (ms):
         min:                                    0.25
         avg:                                    0.25
         max:                                    4.19
         95th percentile:                        0.26
         sum:                                29986.03

Threads fairness:
    events (avg/stddev):           118820.0000/0.00
    execution time (avg/stddev):   29.9860/0.00
```

Iteration 5

Table:

Serial Number of Iterations	Events per second
1	3967
2	3952
3	3942
4	3967
5	3960

Docker:

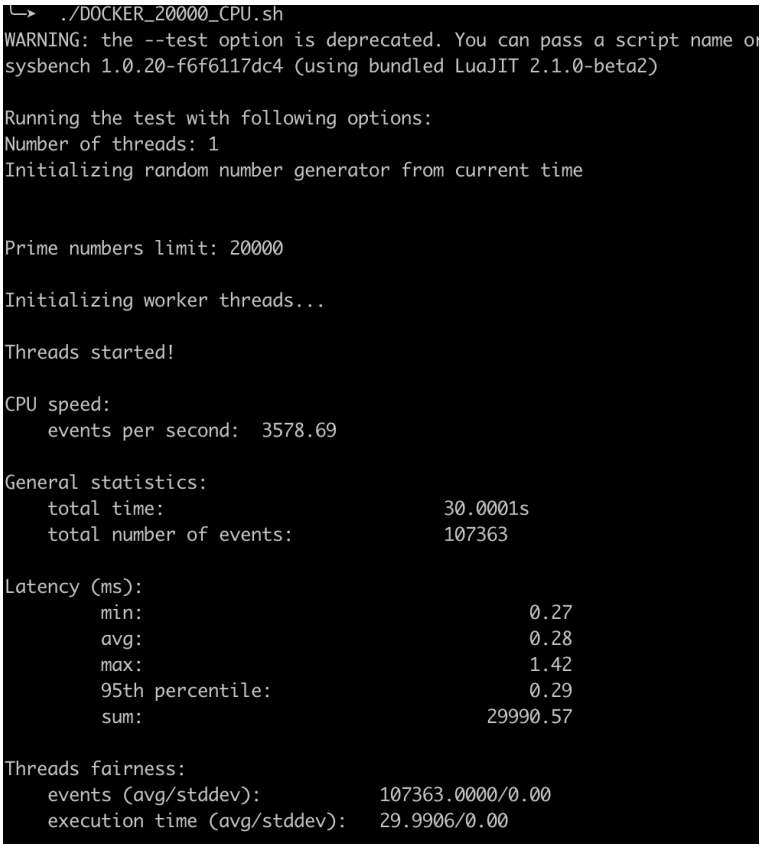


Table here:

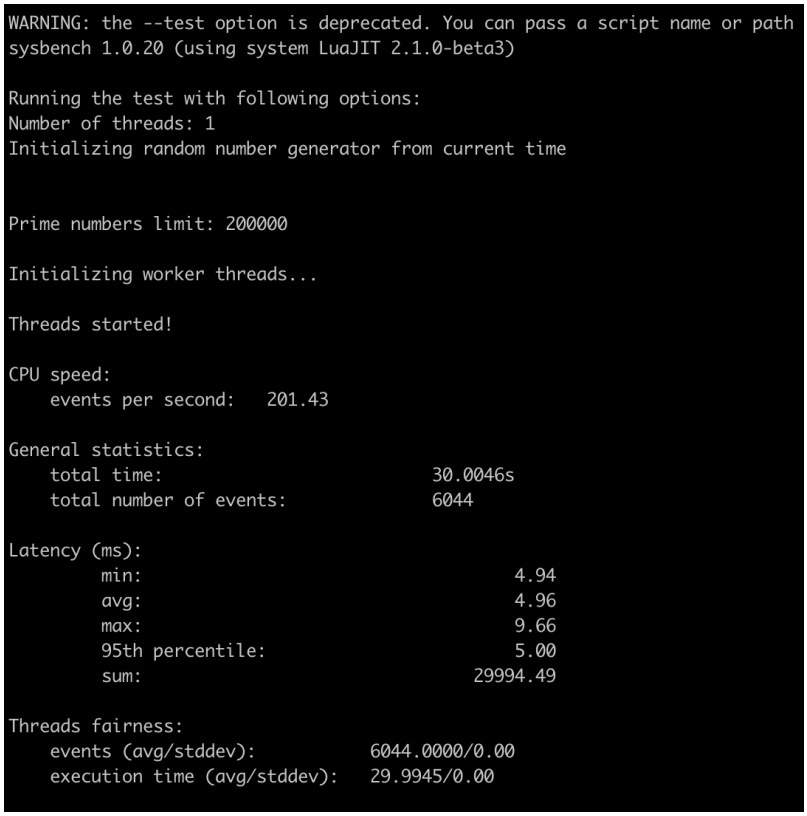
Serial Number of Iterations	Events per second
1	3578
2	3581
3	3582
4	3579
5	3565

Results: Even though events per second reduce, QEMU performance is better than docker for CPU

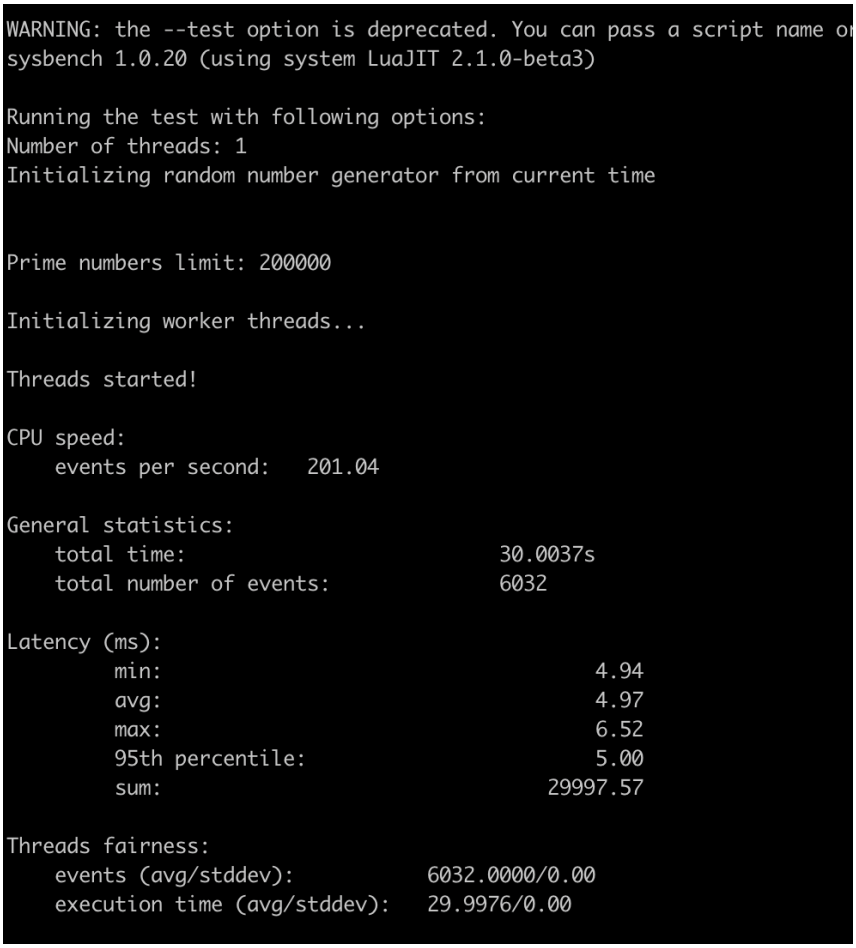
Case: 200000

```
WARNING: the --test option is deprecated. You can pass a script name or path on
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 200000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:   201.32

General statistics:
    total time:                          30.0011s
    total number of events:              6040

Latency (ms):
         min:                                    4.94
         avg:                                    4.97
         max:                                    9.74
         95th percentile:                        5.00
         sum:                                29992.20

Threads fairness:
    events (avg/stddev):           6040.0000/0.00
    execution time (avg/stddev):   29.9922/0.00
```

Iteration 1


WARNING: the --test option is deprecated. You can pass a script name or path
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
Running the test with following options:
Number of threads: 1
Initializing random number generator from current time
Prime numbers limit: 200000
Initializing worker threads...
Threads started!
CPU speed:
events per second: 201.43
General statistics:
total time: 30.0046s
total number of events: 6044
Latency (ms):
min: 4.94
avg: 4.96
max: 9.66
95th percentile: 5.00
sum: 29994.49
Threads fairness:
events (avg/stddev): 6044.0000/0.00
execution time (avg/stddev): 29.9945/0.00

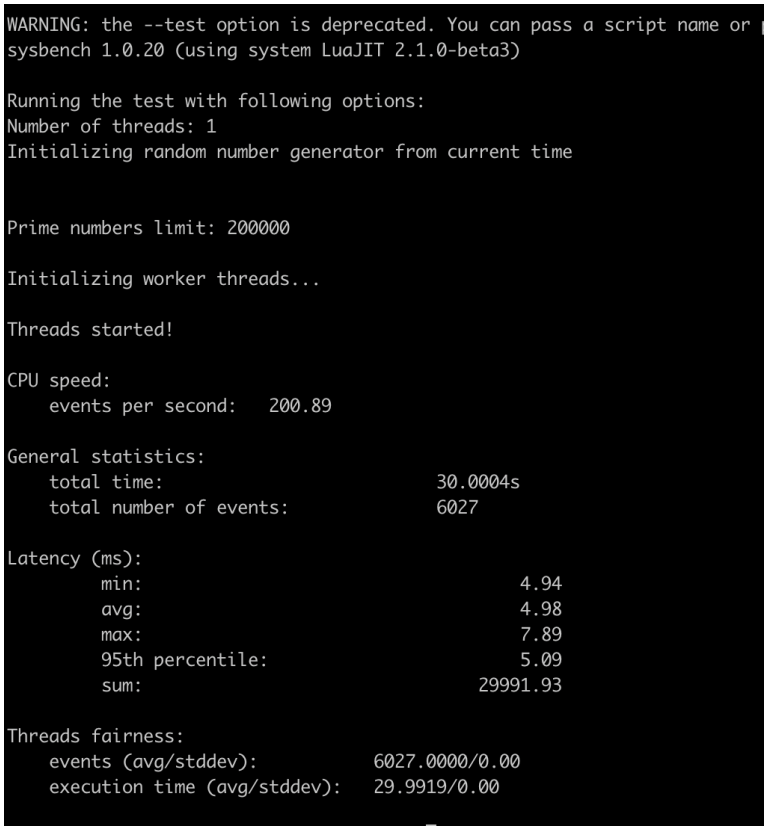
Iteration 2


WARNING: the --test option is deprecated. You can pass a script name or
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
Running the test with following options:
Number of threads: 1
Initializing random number generator from current time
Prime numbers limit: 200000
Initializing worker threads...
Threads started!
CPU speed:
events per second: 201.04
General statistics:
total time: 30.0037s
total number of events: 6032
Latency (ms):
min: 4.94
avg: 4.97
max: 6.52
95th percentile: 5.00
sum: 29997.57
Threads fairness:
events (avg/stddev): 6032.0000/0.00
execution time (avg/stddev): 29.9976/0.00

Iteration 3


WARNING: the --test option is deprecated. You can pass a script name o
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)
Running the test with following options:
Number of threads: 1
Initializing random number generator from current time
Prime numbers limit: 200000
Initializing worker threads...
Threads started!
CPU speed:
events per second: 201.67
General statistics:
total time: 30.0040s
total number of events: 6051
Latency (ms):
min: 4.94
avg: 4.96
max: 6.22
95th percentile: 5.00
sum: 29998.64
Threads fairness:
events (avg/stddev): 6051.0000/0.00
execution time (avg/stddev): 29.9986/0.00

Iteration 4



Iteration 5

Table:

Serial Number of Iterations	Events per second
1	201
2	201
3	201
4	201
5	201

Docker:

Table here:

Serial Number of Iterations	Events per second
1	189
2	189
3	190
4	190
5	189

Conclusion: QEMU VM is faster than Docker in terms of CPU performance on M2 chip

b. File I/O testing

- For QEMU
    - Sequential Rewrite

```
Running the test with following options:
Number of threads: 16
Initializing random number generator from current time


Extra file open flags: (none)
128 files, 24MiB each
3GiB total file size
Block size 16KiB
Periodic FSYNC enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing sequential rewrite test
Initializing worker threads...

Threads started!


File operations:
    reads/s:                      0.00
    writes/s:                     39389.37
    fsyncs/s:                     50483.02

Throughput:
    read, MiB/s:                  0.00
    written, MiB/s:               615.46

General statistics:
    total time:                          30.0202s
    total number of events:              2695992

Latency (ms):
         min:                                    0.00
         avg:                                    0.18
         max:                                  189.95
         95th percentile:                        0.68
         sum:                               476387.27

Threads fairness:
    events (avg/stddev):           168499.5000/1556.94
    execution time (avg/stddev):   29.7742/0.01
```

Serial number of iterations	Results
1	reads/s = 0 writes/sec = 39389.37 fsyncs/sec = 50483.02 events/sec = 89866
2	reads/s = 0 writes/sec = 35536.65 fsyncs/sec = 45551.08 events/sec = 81072
3	reads/s = 0 writes/sec = 37838.16 fsyncs/sec = 48500.17 events/sec = 86321
4	reads/s = 0 writes/sec = 37560.00

	fsyncs/sec = 48143.70 events/sec = 85691
5	reads/s = 0 writes/sec = 40348.27 fsyncs/sec = 51713.14 events/sec = 92248

- Combined random read write

Serial number of iterations	Results
1	reads/s = 1300.86 writes/sec = 8670.37 fsyncs/sec = 27815.02 events/sec = 49444
2	reads/s = 12925 writes/sec = 8616.65 fsyncs/sec = 27637.08 events/sec = 49145
3	reads/s = 10645.87 writes/sec = 7096.16 fsyncs/sec = 22777.17 events/sec = 40479
4	reads/s = 11293 writes/sec = 7529.00 fsyncs/sec = 2415.70 events/sec = 42936
5	reads/s = 12491 writes/sec = 8327.27 fsyncs/sec = 26712.14 events/sec = 47486

- For Docker
  - Sequential rewrite

Serial number of iterations	Results
1	reads/s = 0 writes/sec = 33944.01 fsyncs/sec = 43579.68 events/second = 77564.73

2	reads/s = 0 writes/s=35435.34 fsyncs/sec = 42342.32 events/second = 75343.24
3	reads/s = 0 writes/s = 33958.73 fsyncs/sec = 43684.54 events/second = 77846.34
4	reads/s = 0 writes/s = 32091.93 fsyncs/s = 41321.54 events/second = 73654.18
5	reads/s = 0 writes/s = 34546.54 fsyncs/s = 44323.20 events/second = 78234.32

- Combined random read write

Serial number of iterations	Results
1	reads/s = 13950.71 writes/sec = 9300.24 fsyncs/sec = 29826.54 events/second = 53042.66
2	reads/s = 13155.49 writes/s=8769.99 fsyncs/sec = 28129.15 events/second = 50019.5
3	reads/s = 14353.95 writes/s = 9569.02 fsyncs/sec = 30688.31 events/second = 54574.26
4	reads/s = 14674.66 writes/s = 9782.83 fsyncs/s = 31370.93 events/second = 55796.33
5	reads/s = 14488.90 writes/s = 9659.14 fsyncs/s = 30975.53

	events/second = 55094.1

Conclusion: We see that for sequential rewrite, QEMU is faster than docker but for combined read write, docker is faster than QEMU

Config 2 : 4GB RAM and 4 cores

a. Cpu testing
    i. QEMU

Case 2000:

Serial Number of Iterations	Events per second
1	113926.26
2	113201.06
3	113018.63
4	114854.19
5	113129.79

    ii. With docker

Case 2000:

Serial Number of Iterations	Events per second
1	52698.93
2	52132.09
3	51893.90
4	56403.96
5	56134.25

For all remaining cases, Qemu remains faster than docker in terms of CPU performance.

b. File IO testing
    i. QEMU file IO

1. Sequential Rewrite (QEMU)

Serial number of iterations	Results
1	reads/s = 0 writes/sec = 38097.14 fsyncs/sec = 48828.63 events/second = 87008.56
2	reads/s = 0 writes/s=39530.62 fsyncs/sec = 50664.21 events/second = 90199.3
3	reads/s = 0 writes/s = 35886.20 f syncs/sec = 46000.81 events/second = 81888.26
4	reads/s = 0 writes/s = 33329.60 fsyncs/s = 42727.80 events/second = 76088.93
5	reads/s = 0 writes/s = 33704.58 fsyncs/s = 43206.91 events/second = 76908.83

2. Combined random read write

Serial number of iterations	Results
1	reads/s = 14487.51 writes/s = 9898.23 fsyncs/s = 31739.38 events/second = 56464.43
2	reads/s = 13910 writes/s = 9273 fsyncs/s = 29740 events/second = 52915.76
3	reads/s = 14165.54 writes/s = 9443.58 fsyncs/s =30284.24 events/second = 53880.36

4	reads/s = 12349.31 writes/s = 8232.87 fsyncs/s = 26409.53 events/seconds = 46994.56
5	reads/s = 12001.06 writes/s = 8000.63 fsyncs/s = 25669.69 events/seconds =45638.56

ii. Docker

1. Sequential rewrite

Serial number of iterations	Results
1	reads/s = 0 writes/sec = 21687.04 fsyncs/sec = 27826.67 events/second = 49552.66
2	reads/s = 0 writes/s= 22690.64 fsyncs/sec = 29019.49 events/second = 51806.5
3	reads/s = 0 writes/s = 25688.43 fsyncs/sec = 32946.24 e vents/second = 58659.23
4	reads/s = 0 writes/s = 22382.72 fsyncs/s = 28716.33 events/second = 51146.33
5	reads/s = 0 writes/s = 21880.69 fsyncs/s =28071.27 events/second = 49996.26

2. Combined random read write (docker)

Serial number of iterations	Results
1	rreads/s = 13850.71 writes/sec = 9320.24

	fsyncs/sec = 28826.54 events/second = 53032.66
2	reads/s = 14155.49 writes/s=8754.94 fsyncs/sec = 27129.15 events/second = 50045.5
3	reads/s = 14251.83 writes/s = 9545.02 fsyncs/sec = 30457.31 events/second = 54375.26
4	reads/s = 14324.66 writes/s = 9231.83 fsyncs/s = 31432.93 events/second = 55987.33
5	reads/s = 14231.90 writes/s = 9658.14 fsyncs/s = 30324.53 events/second = 55432.1

Conclusion: A decrease in file I/O is observed in both QEMU VM and docker

Config 3: 6GB 6 cores

a. Cpu testing
    i. QEMU

Case 2000:

Serial Number of Iterations	Events per second
1	113197.63
2	114251.46
3	113318.53
4	114254.59
5	113529.89

    ii. Docker

Serial Number of Iterations	Events per second
1	54026.70
2	52534.09
3	52843.95
4	53403.88
5	56479.38

For all further test cases we see no further significant improvements in CPU performances for both QEMU and docker.

b. File IO testing
    i. QEMU
        1. Sequential Rewrite

Serial number of iterations	Results
1	reads/s = 0 writes/sec = 25473.58 fsyncs/sec = 32673.72 events/second = 58185
2	reads/s = 0 writes/s=27687.54 fsyncs/sec = 33764.65 events/second = 59124.32
3	reads/s = 0 writes/s = 24534.32 fsyncs/sec = 31986.87 events/second = 56126.76
4	reads/s = 0 writes/s = 25765.32 fsyncs/s = 32675.83 events/second = 58234.21
5	reads/s = 0 writes/s = 28675.77 fsyncs/s = 35687.32 events/second = 59543.65

        2. Combined Random Read write

Serial number of iterations	Results
1	reads/s = 9938.58 writes/s = 6625.83 fsyncs/s = 21268.98 events/second = 37854.03
2	reads/s = 9845.43 writes/s = 6541.32 fsyncs/s = 20267.32 events/second = 35643.76
3	reads/s = 9876.73 writes/s = 6543.21 fsyncs/s =21124.43 events/second = 35541.28
4	reads/s = 9765.53 writes/s = 7021.32 fsyncs/s = 30832.11 events/seconds = 39878.32
5	reads/s = 9763.23 writes/s = 6312.63 fsyncs/s = 24569.69 events/seconds =34532.56

ii. Docker IO testing

1. Sequential Rewrite

Serial number of iterations	Results
1	reads/s = 0 writes/sec = 13778.77 fsyncs/sec = 17701.47 events/second = 31506.16
2	reads/s = 0 writes/s= 13543.64 fsyncs/sec = 17634.49 events/second = 30234.5
3	reads/s = 0 writes/s = 12456.32 fsyncs/sec = 16323.87

	events/second = 29876.36
4	reads/s = 0 writes/s = 14328.87 fsyncs/s = 18785.62 events/second = 32324.32
5	reads/s = 0 writes/s = 13534.11 fsyncs/s =17431.89 events/second = 31232.21

2. Combined Random Read Write

Serial number of iterations	Results
1	reads/s = 9501.28 writes/sec = 6334.24 fsyncs/sec = 20336.56 events/second = 36197.8
2	reads/s = 9323.32 writes/s=6213.41 fsyncs/sec = 20223.12 events/second = 36097.87
3	reads/s = 9674.98 writes/s = 6984.73 fsyncs/sec = 20989.89 events/second = 37019.21
4	reads/s = 9594.32 writes/s = 6434.43 fsyncs/s = 20214.43 events/second = 35788.33
5	reads/s = 9287.90 writes/s = 6021.14 fsyncs/s = 18673.53 events/second = 32764.12

Conclusion: I/O performance actually decreases if we increase resource allocation. Performance in QEMU and docker are almost comparable but sometimes QEMU out performs docker slightly.

Git repo details:

Github username : sanfernoronha
Repository name: COEN241
Folder: HW1
Link: https://github.com/sanfernoronha/COEN241
Commit ID: c84544f8d9021538a25dfac21d6f07a1add6a76a