**Question – How much does the virtualization impact the performance of threads compared to native system in two of the following languages - :**

Languages:-

1. C
2. Java
3. Python

**Question: My experimental setup**

Answer -

Native Machine Configurations:-

1. Operating System: Ubuntu, 20.04 LTS on Google Cloud Platform

Machine Type: 8vCPUs or 8 core CPU, 32GB Memory

1. Operating System: macOS Mojave Version 10.14.6

Machine Type: 1.4 GHz Intel Core i5, 8GB Memory

Docker Configurations:

Docker version 20.10.0

**Question: Do you have variation in your test code?**

Variation in Test Code:

Answer: I have used four threaded applications with different time complexities to compare the performance of threads:  
  
  
Threaded Application Time Complexity  
  
1) Merge Sort nlogn

2) Matrix Multiplication n3

3) Linear Search n

4) Binary Search  logn

Variation in Data  
  
I have used array with sizes of 5, 10, 100, 1000, 10000 and 100000 elements and matrices of 5 by 5, 10 by 10, 100 by 100, 1000 by 1000, 10000 by 10000 and 100000 by 100000 elements.

Used random numbers as input generated by rand library function.

Number of threads in each case: 4

**Question: How you ran your tests?**

Answer -: I ran all my tests manually to note down the reading of elapsed time each time I executed the program. I used rand library function to get random numbers as input.

Library Functions Used for noting elapsed time

1. C - **clock\_t clock(void)**
2. Java – **System.nanoTime()**
3. Python – time.time()

**Test results:**

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

JAVA:

Linear Search Java –

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000659233 | 0.0021844 | 0.001530598 | 0.001058998 |
| 10 | 0.000566554 | 0.0018206 | 0.002901207 | 0.001232953 |
| 100 | 0.000576784 | 0.0023628 | 0.001311958 | 0.001149013 |
| 1000 | 0.000546759 | 0.0019909 | 0.0017975 | 0.001315969 |
| 10000 | 0.00058594 | 0.0022953 | 0.001701839 | 0.001372706 |
| 100000 | 0.002259258 | 0.0041781 | 0.003750459 | 0.003361903 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

BinarySearch Java

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000748456 | 0.0026525 | 0.001719777 | 0.001186782 |
| 10 | 0.000745076 | 0.0023796 | 0.001843136 | 0.001390079 |
| 100 | 0.000923904 | 0.0027493 | 0.002142747 | 0.001682688 |
| 1000 | 0.001170925 | 0.0039502 | 0.002327839 | 0.001913991 |
| 10000 | 0.001626652 | 0.0054484 | 0.002924975 | 0.002768748 |
| 100000 | 0.004189109 | 0.0084183 | 0.009463451 | 0.00707601 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

MergeSort Java

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000645975 | 0.0030837 | 0.001762666 | 0.001387455 |
| 10 | 0.000634766 | 0.0030877 | 0.002349834 | 0.001476158 |
| 100 | 0.000855289 | 0.0037935 | 0.001667962 | 0.001530873 |
| 1000 | 0.001122453 | 0.0038767 | 0.002195885 | 0.00321992 |
| 10000 | 0.006659232 | 0.0136752 | 0.002195885 | 0.014550656 |
| 100000 | 0.03224993 | 0.0288319 | 0.047223134 | 0.038257747 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Matrix Multiplication Java

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000702829 | 0.0027597 | 0.001543404 | 0.001449932 |
| 10 | 0.000725467 | 0.0029235 | 0.001670886 | 0.001286441 |
| 100 | 0.017265609 | 0.0174305 | 0.020810688 | 0.02184987 |
| 1000 | 1.024975366 | 0.9405257 | 1.058107357 | 0.588705198 |

Python :

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Linear Search Python

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000626802 | 0.002584457 | 0.00083375 | 0.009762287 |
| 10 | 0.000505924 | 0.001094103 | 0.000925541 | 0.000922441 |
| 100 | 0.000669956 | 0.001330137 | 0.000867367 | 0.000921488 |
| 1000 | 0.000688791 | 0.001207829 | 0.0009799 | 0.002235413 |
| 10000 | 0.001531839 | 0.002443552 | 0.001614809 | 0.001839876 |
| 100000 | 0.010164976 | 0.010019779 | 0.006597996 | 0.009417057 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Binary Search Python

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000560999 | 0.001418591 | 0.000962734 | 0.00105691 |
| 10 | 0.000906944 | 0.001077414 | 0.00083518 | 0.001205206 |
| 100 | 0.000486135 | 0.001334667 | 0.000869751 | 0.001011848 |
| 1000 | 0.000728846 | 0.001137257 | 0.000954151 | 0.001188517 |
| 10000 | 0.000545025 | 0.001470566 | 0.000933409 | 0.001098156 |
| 100000 | 0.000499964 | 0.002005577 | 0.0012362 | 0.001284838 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Merge Sort Python

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000745058 | 0.001553059 | 0.001012802 | 0.001256943 |
| 10 | 0.001112938 | 0.001701832 | 0.001056194 | 0.00123477 |
| 100 | 0.002901077 | 0.002679348 | 0.001707077 | 0.002587795 |
| 1000 | 0.014178991 | 0.016524553 | 0.011401415 | 0.018471479 |
| 10000 | 0.167813063 | 0.22920084 | 0.144711733 | 0.232206106 |
| 100000 | 2.121235132 | 2.781701803 | 1.867905378 | 3.01830554 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Matrix Multiplication Python

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000989914 | 0.001284361 | 0.001174212 | 0.001393795 |
| 10 | 0.001870871 | 0.001380205 | 0.001559973 | 0.00153923 |
| 100 | 0.188896894 | 0.21668911 | 0.221670866 | 0.240695 |
| 1000 | 174.4348962 | 223.7264972 | 158.9420576 | 249.8707681 |

C :

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Linear Search C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000364 | 0.000659 | 0.000555 | 0.000414 |
| 10 | 0.00021 | 0.000431 | 0.00049 | 0.000456 |
| 100 | 0.000167 | 0.00044 | 0.000571 | 0.000876 |
| 1000 | 0.000344 | 0.00046 | 0.000651 | 0.000711 |
| 10000 | 0.000272 | 0.000595 | 0.000633 | 0.000519 |
| 100000 | 0.000751 | 0.000748 | 0.000851 | 0.000885 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Binary Search C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000336 | 0.000563 | 0.000442 | 0.00053 |
| 10 | 0.000379 | 0.000447 | 0.000501 | 0.000388 |
| 100 | 0.000335 | 0.000593 | 0.000426 | 0.00048 |
| 1000 | 0.00037 | 0.000501 | 0.00044 | 0.00049 |
| 10000 | 0.000419 | 0.000538 | 0.000924 | 0.000605 |
| 100000 | 0.000537 | 0.000677 | 0.000711 | 0.000705 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

MergeSort C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000387 | 0.000467 | 0.000494 | 0.000567 |
| 10 | 0.000634 | 0.001058 | 0.000585 | 0.00084 |
| 100 | 0.000559 | 0.00081 | 0.000616 | 0.000751 |
| 1000 | 0.001089 | 0.000985 | 0.000931 | 0.001051 |
| 10000 | 0.004274 | 0.010665 | 0.00409 | 0.005025 |
| 100000 | 0.027438 | 0.14572 | 0.032917 | 0.04858 |

**X Axis: Size of Array/Dimensions of Matrix**

**Y Axis: Runtime in seconds**

Matrix Multiplication C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Size** | **macOS (Mojave) (seconds)** | **macOS (Mojave) Docker (seconds)** | **GCP (seconds)** | **GCP Docker (seconds)** |
| 5 | 0.000343 | 0.000551 | 0.000396 | 0.000523 |
| 10 | 0.000216 | 0.000455 | 0.000505 | 0.000625 |
| 100 | 0.003096 | 0.006445 | 0.007871 | 0.009177 |
| 1000 | 2.526697 | 4.42745 | 5.545509 | 6.090742 |

How accurate are the results?

Sum of all timings- 841.560708755 seconds

Sample Size – 264 runs

Standard Deviation - 25.046838226495 seconds

Mean - 3.187729957405 seconds

Calculation of margin of error:

Formula:–

Qr code

Description automatically generated

* n - the sample size
* Zc - Critical value (used for confidence value)

○ 95% confident use Zc of 1.96

○ 99% confident use Zc of 2.78

* σ - Standard Deviation ● E - margin of error

○ If you are measuring in seconds how many seconds do you want the average away from the “true” run time

Here n = 264

For 99% Confidence, Zc = 2.78

σ = 25.046838226495

Putting these in the formula we get:

E = 3.971 seconds

Hence, for 99% confidence the margin of error is 3.971 seconds.

Total of timings for each environment -

macOS (Mojave) (native system) - 180.622208905 seconds

macOS (Mojave) Docker - 232.66801274 seconds  
Google Cloud Platform (Ubuntu 20.04 LTS) (native system) - 167.985920225 seconds

Google Cloud Platform Docker (Ubuntu 20.04 LTS) - 260.284566885

Conclusion –

1) Performance of threads in Python is much worse as compared to C and Java.  
  
2) Performance of threads on native systems is better than performance of threads in Docker virtualization environment.

**Docker Lost the battle**!!!

Question: What you learned from answering the question

Answer: Following are my observations for this experiment:

1. Docker does impose performance costs. Processes running within a container will not be quite as fast as those run on the native OS.
2. Performance of python really deteriorates as compared to C and Java in case of larger

Input.

Motivation:

Threads are a really important part of OS. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process. So, comparing the performance of threads in different languages and environments according to me is really important to figure out the best Language and environment for writing threaded applications in operating systems. Also, since docker   
enables more efficient use of system resources and enables application portability, it is important to figure out whether docker virtualization of native system enhances the performance of threads or not