Design Documentation

Benchmarking:

Benchmarking is used to measure performance of the system using a specific key performance indicator resulting in a metric of performance that is then compared to others.

So here a computer program is run and tested performing various operations in order to assess the relative performance of an object, normally running a number of standard tests and trials against it.

So, Benchmarking is performed on:

1. CPU
2. Disk and
3. Network

It is performed on Amazon's EC2 as it is the oldest and most mature cloud server platforms that defines CPU tiers across its 8 different instance sizes in terms of ECU i.e EC2 Compute Unit.

Terminal used: Amazon web services.

Instance type: t2.micro(free intance)

Testing method: Bash script

Coding language: Java

OS Platform: Linux

Properties: High Frequency Intel Xeon processors with Turbo upto 3.3GHz

Consistent baseline performance with Burstable CPU and provides balance of compute, network and memory resources.

1. CPU
   In CPU benchmarking, the processor speed is measured in terms of Floating Point Operations and Input Output Operations i.e (Giga Flops, 10^9 Flops) and (Giga IOps, 10^9 IOps).

   So the processor speed is measured for both of them by running four threads simultaneously.
   Using java.util.concurrent package, the CyclicBarrier inbuilt class is imported in order to maintain the concurrency of all the threads invoked at the same time, this will increase the performance of the processor.
   Then 12 operations are performed in the run method of all the threads by iterating it 15,000 times inside a for loop to measure the performance depending on the call to the threads as needed i.e 1 thread, 2 threads and consequently for 4 threads.
   So total of 6 experiments are performed with threads and each for IOps and Flops.

   The theoretical peak performance of this Amazon Intel Xeon processor is around 6.45 GFlops/core.
   Compared to this the practically achieved efficiency is increased in number of flops/second to 13.
   Also while performing run on all three of them and that too for 10-minutes for each one, 600 samples for each Flops and IOps were achieved and are plotted in the graph.

   SPEC is used to measure the design tradeoffs for evaluating peak performance. Multiple operations are run in for many times to test the speed of cpu. So for IOps the integer operations are run in the same manner and with the timers mentioned in the code for both before starting to run these operations and at the end of it so the time elapse (number of operations * number of iterations it takes) are divided by the time elapse i.e (ending time – starting time) in order to know the values of the Flops and IOps. These time is obtained from the System in terms of nanoseconds in order to achieve high precision and then converted to seconds as per the need.

   Then performance is again measured by running it on Linpack benchmark which measures the system floating point computing power. Thus these multiplication, division, addition, etc. modulo operations performed are consisted by Linpack as floating point operations(64-bit).

2. Disk:

As discussed above the Amazon t2.micro instance is used to measure the performance of the disk in terms of speed by performing many experiments as:

By importing java.nio package and using java.nio.FileSystems and java.nio.channels to create and eventually open files passing parameters of their path and name to read and write files but on my processor I found the Buffered Reader and Buffered Writer as the fastest way of reading and writing the files of varying sizes and it gives me the better performance to measure the disk speed at a very good rate.

So for both the threads first the sequential write(PrintWriter) operation is performed by maintain array of specific bytes of [1024] filling it using fill method with the characters for generating file with the needed file size and then sequential read by accessing file after creating it and then writing it sequentially with varying data file sizes as 1B, 1KB and 1MB for each of the two and then similarly for random read and write for all the three files by using writeUTF and reading it line by with the above mentioned file sizes and accessing them, so total 12 experiments are performed like this on both the threads simultaneously as well giving in all 24 operations.

Thus the throughput is measured for all these experiments by calculating start and stop timing using System.nanotimes() by placing timers before the file write and after it is closed for both of them simultaneously.

And so the various file sizes are then divided by these time elapse i.t difference between the start and end timings to get the actuall throughput in MBPS.

Also latency is measured for all the 24 experiments in terms of milliseconds by converting time difference from nanoseconds to milliseconds.

At last the IOzone benchmark is used to test the actually disk performance in terms speed as it is the filesystem benchmark tool.

3. Network

   Here the speed of a network is measured between both the server and client as both Transmission Control Protocol and User Datagram Protocol.

   TCP is the one in which server is only one that is responsible for establishing connection with the multiple clients and eventually sending the acknowledgement as ServerConnected by connecting to the ServerSocket at one particular port so that the particular currently running client knows it and thus accepts the connection request at that particular port and if in this process the port is not free then catch will generate the exception for the error in the connection of either of them.

   Now the client will send and receive the files using FileInputStream, BufferedInputStream, OutputStream and DataOutputStream and InputStream and DataInputStream respectively. Thus these are performed three times with each of the files with varying sizes like 1B, 1KB and 64KB and that too for both the threads so in all performs 6 experiments for TCP and the files which are using in the transaction are stored in the JRE System Library.

   Also the clientConnection class will be used for selecting the particular client among many in order to perform the above mentioned send and receive operations.

   Now for UDP using java.net package, the DatagramSocket is used in order to establish network connection at the server side on the available free port, so now the server is rready to receive the data packet with DatagramPacket and eventually send it to the client.

   Now client will take as an input the data of varying sizes as 1B, 1KB and 64KB from the user and connects itself to the localhost using InetAddress for sending and receiving bytes to the server and then client will perform some action like here change the array elements in bytes to Uppercase characters and sending the modified sentence again back server. This is done for both the threads and so again 12 experiments each.

   And the timer is placed using System.nanotime() once before sending the packet and after receiving the packet and that difference which divides the various file sizes gives the actual throughput in Mbps of the network for all the 24 experiments and eventually this time is used in milliseconds which depicts the latency.

   And finally iperf benchmark, available for network is used to test the performance in best manner.

Conclusion

The system is tested on the Amazon Ec2 Free tier Ubunutu environment for CPU, Network and Disk.

More better performance can be achieved with the synchronized threading mechanism same as the cyclic barrier that I used to maintain concurrency between the threads, it can improve it well.

Performance is degraded a bit in comparison with theoretical processor performance and due to mutex for 1 second samples each for the CPU it is interrupted more to find 600 Samples for IOps and Flops.