DESIGN DOCUMENT

BENCHMARKING TOOL

PA 1

DESIGNED BY:

SAMI AHMAD KHAN
A20352677
COURSE: CS 553 CLOUD COMPUTING

INTRODUCTION

What is Benchmarking?

In computing, a benchmark is the act of running a computer program, a set of programs, or other operations, in order to assess the relative performance of an object, normally by running a number of standard tests and trials against it. The term 'benchmark' is also mostly utilized for the purposes of elaborately designed benchmarking programs themselves.

Benchmarks provide a method of comparing the performance of various subsystems across different chip/system architectures.

Types of benchmarks

1. Real program
    - word processing software
    - tool software of CAD
    - user's application software (i.e.: MIS)
2. Component Benchmark / Microbenchmark
    - core routine consists of a relatively small and specific piece of code.
    - measure performance of a computer's basic components [3]
    - may be used for automatic detection of computer's hardware parameters like number of registers, cache size, memory latency, etc.
3. Kernel
    - contains key codes
    - normally abstracted from actual program
    - popular kernel: Livermore loop
    - linpack benchmark (contains basic linear algebra subroutine written in FORTRAN language)
    - results are represented in MFLOPS
4. Synthetic Benchmark
    - Procedure for programming synthetic benchmark:
        - take statistics of all types of operations from many application programs

        - get proportion of each operation
        - write program based on the proportion above
    - Types of Synthetic Benchmark are:
        - Whetstone
        - Dhrystone
    - These were the first general purpose industry standard computer benchmarks. They do not necessarily obtain high scores on modern pipelined computers.
5. I/O benchmarks
6. Database benchmarks: to measure the throughput and response times of database management systems (DBMS')
7. Parallel benchmarks: used on machines with multiple cores, processors or systems consisting of multiple machines

OBJECTIVE:

The objective of this assignment is to design benchmarking application that will benchmark different parts of a computer system, from the CPU, memory, disk, and network.

Following are the hardwares on which I have performed benchmarking:

1. CPU Benchmarking:

    Measuring the processor speed, in terms of floating point operations per second (Giga FLOPS, 109 FLOPS) and integer operations per second (Giga IOPS, 109 IOPS) at varying concurrency levels (1 and 2)

2. Memory Benchmarking through:
    - Measuring the memory speed of the host by using read and write operations (e.g. memcpy), sequential access, random access, varying block sizes (1B, 1KB, 1MB), and varying the concurrency (1 thread & 2 threads) .
    - The metrics measured are throughput (Megabytes per second, MB/sec) and latency (milliseconds, ms)

3. Network Benchmarking through:

    Measuring the network speed between 2 instances. The parameter space includes the TCP protocol stack, UDP, varying packet/buffer size (1B, 1KB,

64KB), and varying the concurrency (1 thread & 2 threads).

TOOL USED:

The tools used in designing this system are:

- Eclipse Luna 4.4 for Java codes
- Sublime Text for C code
- Amazon AWS for testing on cloud

DESIGNING AND DOCUMENTATION

OVERVIEW

Following are the steps in the designing of the programs:

1. CPU BENCHMARKING:

For benchmarking the CPU, I have performed the following steps:

- Created a class **Cpu,** that will first ask the user to input the number of concurrent threads that he wants to run. He should enter either for 1 thread or 2 or maximum 4 threads.
- This class will then run the object of new class **CpuLoader** in threads as per the number of concurrent threads entered by the user.
- **CpuLoader** consists of the following methods:

  1. iopsCalculator() – It will calculate and return the total time for performing multiplication and addition of 3 randomly generated Integer number.
  2. flopsCalculator() – It will calculate and return the total time for performing multiplication and addition of 3 randomly generated Float number.
  3. operationsCalc(float operations) – It will take the total time returned by 1 and 2 and calculate GFLOPS and IOPS accordingly using the formula:
  GFLOPS/IOPS = (7 * Integer.MAX_VALUE) / (operations / 1000) / 10^9

2. MEMORY BENCHMARKING:

For benchmarking the Memory, I have performed the following steps:

- Created a file using .c extension and coded using C.
- I have allocated 3 block sizes of 1Byte, 1KB and 1MB in an array. And every thread which is created performs memcpy using all the 3 block sizes.
- Formulas used for calculating latency and throughput are:

Latency = (1000.0 *(End Time – Start Time) + (End Time - Start Time)/1000.0) / iterations through all block sizes

Throughput = (No of concurrent threads * block size / (1024.0 * 1024.0)) / (total no of executions/ 1000.0);

3. NETWORK BENCHMARKING:

For Network benchmarking I have calculated network speed TCP and UDP server client connection using the following steps:

1. **TCP:** For TCP network speed benchmarking I have created 3 classes:
    - **ServerTCP -->** This class starts the server and then call the object of another class **tcpServerManager** that will run in threads (1 thread and then 1 and 2 threads)
    - **tcpServerManager -->** This will implement *Runnable* and will run in threads. It will first receive a set of bytes from Client and then send the same to it.
    - **ClientUDP -->** This will also implement Runnable and will make threads and make 1:1 connection with the Server threads. It will first send a set of bytes to it and then receive the same. The total time taken from sending the bytes and receiving the same through different block sizes is calculated and then the Network speed is calculated accordingly.

2. UDP: For UDP network speed benchmarking also I have created 3 similar classes which perform similar functions.

SOFTWARE REQUIREMENT

- Windows, Linux, Macintosh

- Java Virtual machine and gcc compiler