CS 553 CLOUD COMPUTING

Programming Assignment -1

RONAKKUMAR MAKADIYA (CWID: A20332994)
KAUSTUBH BOJEWAR (CWID: A20329244)
SOURABH CHOUGALE (CWID: A20326997)

Design Document

We have calculated following five types of Benchmarks

- CPU Benchmarking
- GPU Benchmarking
- Memory Benchmarking
- Disk Benchmarking
- Network Benchmarking

CPU Benchmarking:

- We have written code in C to measure the benchmarks more accurately since Java and other high level programming languages would slant the results due to its running environment. At first, we would calculate the time taken for running an empty loop. Using this exact time would be calculated after subtracting it from the total time to get the accurate results.
- The program consists of one loop which performs addition for floating and integer type values for calculating GFLOPS and GIOPS respectively. The empty loop time would be subtracted from the total time. The loop iterating variables are volatile to prevent the compiler optimizing the empty loops. As it is difficult to calculate CPU speed in a single iteration so for that we have make than loop to run for INT_MAX times.
- For multi-threading we have used the pthread library. Each of the threads would calculate the GFLOPS and IOPS.

Improvement and extension to the program:

- We would get more accurate benchmark results if we could isolate the effect of cache. The operation needs to be random enough so that the processor does not optimize it by serving the result from cache.
- We could differentiate effect of FPU and CPU on the benchmark.

GPU Benchmarking:

- GPU benchmarking calculates performance of graphic processor.
- We used NVCC compiler and CUDA for compiling GPU and running the programs on Jarvis.
- We used CUDA parallel programming model for benchmarking.
- We calculated FLOPS (Floating Point Operations per Second) and IOPS (Input/output operations per second).
- Program flow: Data is sent from host (CPU) to device (GPU). Integer and Floating point operations are performed on device (GPU) and sent back to the host (CPU).
- _global_ keyword used in programming indicates that the operations are running on the GPU.
- We are using cudaEventRecord() to calculate the time required for operations.

Improvement and extension to the program:

- We can extend the program to more than 1 GPU devices and copy data from them.

Memory Benchmarking:

- We used C language for memory benchmarking and G++ compiler to compile the program.
- There are two operations performed on memory one is read and another is write.
- Also reading and writing is performed either sequentially or randomly. For sequential access of memory read and write operation performed sequentially and for random access we calculated random offset.
- Three block sizes are considered while reading and writing. 1 byte, 1 Kilobyte and 1 Megabyte.
- A set of above operations is performed using 1 thread and 2 threads.
- CalculateBenchmark() is main function which implements all computations of benchmarking.
- memcpy() and memset() are two functions used for memory read and write of data. We can also use alternative methods to read and write operations.

Improvement and extension to the program:

- We used memcpy() and memset() functions to test this benchmark. We can use alternative methods to test such as swap pointers only, not the data itself etc.
- Isolate the effect of cache.

Disk Benchmarking

- The design of program includes declaration of two functions each for sequential and random operations.
- These functions are called in main() block using threads. A new thread is created every time and using clock() function , start and end time are calculated.
- The time difference between both the time would result in Latency. Read and write speed for both sequential and random is calculated.
- As this program is totally menu driven, it is based on user choice what to calculate.
- Fread(),fwrite() and fseek functions are used to perform several read write operations.

Improvement and extension to the program:

- These benchmarks are implemented in virtual box, due to this there is some variations in the values. These programs can be executed in other environments also.
- We can extend this project to find the rate of data transfer in virtual disk through virtualization.
- As there are possible bad sectors in disk which may affect the latency and may increase it. So we have to check for such bad sectors.
- The concurrency level can be increased.
- Since we are running the virtual machine only one core, the rate and amount of data transfer is limited to this core. We can extend it to multicore environment to get more accurate results.

Network Benchmarking:

- We used Java programming for network benchmarking since transmission control protocols like TCP and UDP are easy to implement through java sockets.
- Benchmarking is divided into two types of programs: One is for client side and another is for server side.
- The basic functionality of program is transmission of data blocks or packets of varying size from client to server and acknowledgment from server to client.
- The start time and end time of this transmission is calculated and from them latency and throughput is calculated.

Improvement and extension to the program:

- We can increase the block size which is to be transmitted.
- We can test this benchmark from one node to another node.
- We can extend program to wireless network.
- Many times due to slower connection latency of network may go up and its throughput fallen down.
- To avoid possible data loss through the network we should have rigid network architecture.