README FOR PA1

MANUAL

VIKRAM G AND Mouna Giri

PA1

This Assignment has four components implemented:

1. CPU
2. MEMORY
3. DISK
4. NETWORK
5. **CPU BENCHMARK**

* Benchmark measures the speed of the processor in terms of Giga FLOPS and Giga IOPS running multiple instructions concurrently.
* The code is inside the file threaddemo.c
* The code checks for 1, 2, 4 and 8 threads.
* RUN:

1. Go to the file location [Inside the directory in which the C file is located]
2. Run the below command to compile the C file.

$ gcc –pthread threaddemo.c –o Output

1. Run the below command to execute the code

$ ./Output

* When you run the code by following the above steps you will get the output like the following:

[cc@pa1-mon ~]$ gcc -pthread threaddemo1.c -o Output13

[cc@pa1-mon ~]$ ./Output13

CPU performance benchmark:

The processor speed in terms of GigaIOPS with 1  threads is 1.578502

The processor speed in terms of GigaIOPS with 2  threads is 3.825424

The processor speed in terms of GigaIOPS with 4  threads is 3.773795

The processor speed in terms of GigaIOPS with 8  threads is 2.863006

The processor speed in terms of GigaFLOPS with 1  threads is 2.067615

The processor speed in terms of GigaFLOPS with 2  threads is 4.000017

The processor speed in terms of GigaFLOPS with 4  threads is 3.805590

The processor speed in terms of GigaFLOPS with 8  threads is 2.991353

**2. MEMORY BENCHMARK**

* Benchmark measures the memory speed of your host
* The code checks for 1, 2, 4 and 8 threads.
* The code asks the user inputs for the size of the block to check the speed. The block size to be selected are 8b, 80b, 8MB and 80MB.
* This benchmark code performs:

1. Read and Write
2. Sequential Write
3. Random Write

* The code is inside the file memory.c
* The Result displayed calculates the latency for 8b for the different number of threads and throughput for 80b, 8MB and 80MB for the different number of threads.
* RUN:

1. Go to the file location [Inside the directory in which the C file is located]
2. Run the below command to compile the C file.

$ gcc –pthread memory.c –o Output\_mem

1. Run the below command to execute the code

$ ./Output\_mem

1. Enter the desired number when the below message is displayed in the console.

Memory benchmarking

Select the block size : Enter 1 for 8B,2 for 8KB, 3  for 8MB, 4 for 80MB

2

1. After the execution of the program, you re-run to try the other block by typing the below command

$ ./Output\_mem

* When you run the code by following the above steps you will get the output like the following:

[cc@pa1-mon ~]$ gcc -pthread memory1.c -o Output5

[cc@pa1-mon ~]$ ./Output5

Memory benchmarking

Select the block size : Enter 1 for 8B,2 for 8KB, 3  for 8MB, 4 for 80MB

2

sequential read+write memory access using different number of threads and their latency and throughput

memory function for size 8192

Throughput of memory with 1 threads is 2745.236084

Throughput of memory with 2 threads is 6876.717773

Throughput of memory with 4 threads is 8135.304199

Throughput of memory with 8 threads is 6768.013184

sequential write memory access using different number of threads and their latency and throughput

memory function for size 8192

Throughput of memory with 1 threads is 3940.436035

Throughput of memory with 2 threads is 9223.484375

Throughput of memory with 4 threads is 11397.903320

Throughput of memory with 8 threads is 13363.653320

random write memory access using different number of threads and their latency and throughput

memory function for size 8192

Throughput of memory with 1 threads is 3015.912842

Throughput of memory with 2 threads is 7803.409180

Throughput of memory with 4 threads is 9363.463867

Throughput of memory with 8 threads is 8568.579102

**3. DISK BENCHMARK**

* Benchmark measures the disk speed.
* The code checks for 1, 2, 4 and 8 threads.
* The code asks the user inputs for the size of the block to check the speed. The block size to be selected are 8b, 80b, 8MB and 80MB.
* This benchmark code performs:
  1. Read and Write(Sequential)
  2. Sequential Read
  3. Random Read
* The code is placed in three files and they are
  + DiskTask\_ReadnWrite.java for checking Read and Write(Sequential)
  + DiskTask\_Sequential.java for checking Sequential Read
  + DiskTask\_Random.java for checking Random Read
* The Result displayed calculates the latency for 8b for the different number of threads and throughput for 80b, 8MB and 80MB for the different number of threads.
* RUN:

1. Go to the file location [Inside the directory in which the java files are located]
2. Place the Large Text file name largeFile.txt under the same directory.
3. Run the below command to compile all the three java file.

$ javac DiskTask\_Random.java

$ javac DiskTask\_ReadnWrite.java

$ javac DiskTask\_Sequential.java

1. Run the below command to execute the code

$ java DiskTask\_Random

1. Enter the desired number when the below message is displayed in the console.

NOTE: Please enter the block size in bytes. For 8M – 8192b and 80MB – 81920b

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit]

1. After the execution of the program, you will get the below message on the console. You can enter the other block size else type 0 to quit.

NOTE: Please enter the block size in bytes. For 8M – 8192b and 80MB – 81920b

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit]

1. Run the below command to execute the code

$ java DiskTask\_Sequential

1. Enter the desired number when the below message is displayed in the console.

NOTE: Please enter the block size in bytes. For 8M – 8192b and 80MB – 81920b

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit]

1. After the execution of the program, you will get the below message on the console. You can enter the other block size else type 0 to quit.

NOTE: Please enter the block size in bytes. For 8M – 8192b and 80MB – 81920b

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit]

1. Run the below command to execute the code

$ java DiskTask\_ReadnWrite

1. Enter the desired number when the below message is displayed in the console.

NOTE: Please enter the block size in bytes. For 8M – 8192b and 80MB – 81920b

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit]

1. After the execution of the program, you will get the below message on the console. You can enter the other block size else type 0 to quit.

NOTE: Please enter the block size in bytes. For 8M – 8192b and 80MB – 81920b

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit]

* When you run the code by following the above steps you will get the output like the following:

[cc@pa1-mon ~]$ java DiskTask\_Random

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit] 8

8 is the Block Size

No of threads running is 1

Latency for random Read9.0427772635594E-5 ms

No of threads running is 2

Latency for random Read1.041881136316806E-4 ms

No of threads running is 4

Latency for random Read1.9754204449336975E-4 ms

No of threads running is 8

Latency for random Read3.863589203860611E-4 ms

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit] 80

80 is the Block Size

No of threads running is 1

Throughput for Random Read104.45398472239087 MB/sec

No of threads running is 2

Throughput for Random Read97.28826238226101 MB/sec

No of threads running is 4

Throughput for Random Read49.69796480729612 MB/sec

No of threads running is 8

Throughput for Random Read24.92746790276488 MB/sec

Please enter the blocksize[8b, 80b, 8M , 80Mb and 0-quit] 80

**4.NETWORK BENCHMARK**

Benchmark measures the network speed over the loopback interface card

·         The code checks for 1, 2, 4 and 8 threads.

·        This benchmark code performs:

TCP protocol stack, fixed packet/buffer size (64KB): transfers a file of size around 500MB from client to server using TCP protocol stack.

·         The code is inside the file net\_client.c and net\_server.c

·         One additional text file must pasted in the file location where C files will be placed and the file name is new.txt

·         The Result displayed calculates the latency and throughput.

·         RUN:

a.      Go to the file location [Inside the directory in which the C file is located]

b.      Run the below command to compile and run the C file.

$ gcc –pthread net\_server.c –o server

$ ./server.c

c.       Open one more terminal to compile and run the client.c file

$ gcc –pthread net\_client.c –o client

       ./client.c

Terminal will prompt to enter the number of threads, please enter the number of threads between 1 or 2 or 4 or 8.

The ouput file will generate a text file in same location as C files as Downloadedfile.txt which will have same content as new file.