Performance evaluation

There are three experiment sets in this file.

Experiment set 1

```
Now begin to test
Peer node[0] do [10001] times register, [9999] times search, [9999] times obtain
Total time[33931601]us
Average register time is [150]us
Average search time is [164]us
Average download time is [3077]us
TEST DONE
```

The time we measured here: register and search time is when we receive from the server. And download time is the time receive from the server plus the time the file is written to disk.
The above is experiment set 1 with only 1 node.

```
Now begin to test
Peer node[0] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[72002359]us
Average register time is [100]us
Average search time is [109]us
Average download time is [6990]us
TEST DONE

Now begin to test
Peer node[1] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[72677178]us
Average register time is [152]us
Average search time is [248]us
Average download time is [6867]us
TEST DONE
```

The above is experiment set 1 with exactly 2 nodes.

```
Now begin to test
Peer node[0] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[111151751]us
Average register time is [72]us
Average search time is [90]us
Average download time is [10952]us
TEST DONE
```

```
Now begin to test
Peer node[1] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[115192414]us
Average register time is [234]us
Average search time is [294]us
Average download time is [10989]us
TEST DONE
```

```
Now begin to test
Peer node[2] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[125910569]us
Average register time is [387]us
Average search time is [2349]us
Average download time is [9854]us
TEST DONE
```

```
Now begin to test
Peer node[3] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[121868468]us
Average register time is [341]us
Average search time is [1163]us
Average download time is [10681]us
TEST DONE
```

The above is exactly with 4 nodes.

```
Peer node[0] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[282395208]us
Average register time is [321]us
Average search time is [835]us
Average download time is [27082]us
TEST DONE
```

```
Now begin to test
Peer node[3] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[244305036]us
Average register time is [69]us
Average search time is [80]us
Average download time is [24280]us
TEST DONE
```

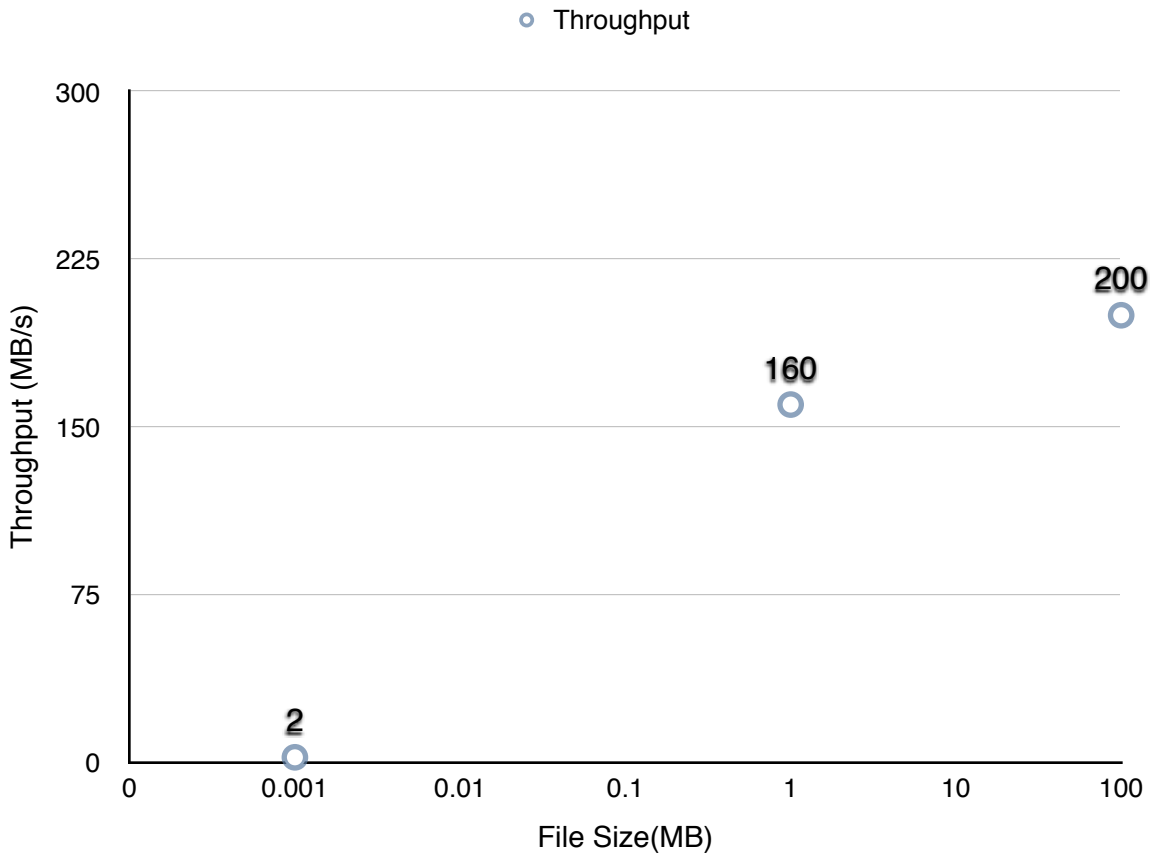
```
Now begin to test
Peer node[5] do [10002] times register, [10000] times search, [10000] times obta
in
Total time[304525899]us
Average register time is [637]us
Average search time is [2815]us
Average download time is [26999]us
TEST DONE
```

The above is exactly with 8 nodes that we only capture three here.

Explain:

1. Since the experiment is tested in one machine, the time for each operation will grown when nodes increase.
2. since the hash table is implemented by myself, comparing with the hash table in assignment 2 in which hash table is using third party library, the performance is lower. The reason is that the implementation uses lock and unlock to exclusive access. Since this assignment do not require to be performance perfect, I loosely use this hash table implementation and conduct the experiment.
3. So as a result, the average register and search time would be growth accordingly with the nodes number.
4. While, for each group experiment, the download time is basically the same, which the majority of time is spent on the written to disk.

Experiment set 2:



(more detail data picture shows below)

```
Now begin to test
Peer node[7] do [3] times register, [1] times search, [1] times obtain
Total time[496758]us
Average register time is [255]us
Average search time is [1177]us
Average download time is [494813]us
TEST DONE
```

The above shows the average time transfer one 100M file within 8 nodes.
Note that this represents 100MB/0.5s = 200Mb/s

```
Now begin to test
Peer node[5] do [102] times register, [100] times search, [100] times obtain
Total time[617262]us
Average register time is [81]us
Average search time is [86]us
Average download time is [6003]us
TEST DONE
```

The above shows the average time transfer 100 1M file within 8nodes.
Note that this represents 1MB/0.006s = 160MB/s

```
Now begin to test
Peer node[2] do [102] times register, [100] times search, [100] times obtain
Total time[68483]us
Average register time is [75]us
Average search time is [79]us
Average download time is [527]us
TEST DONE
```

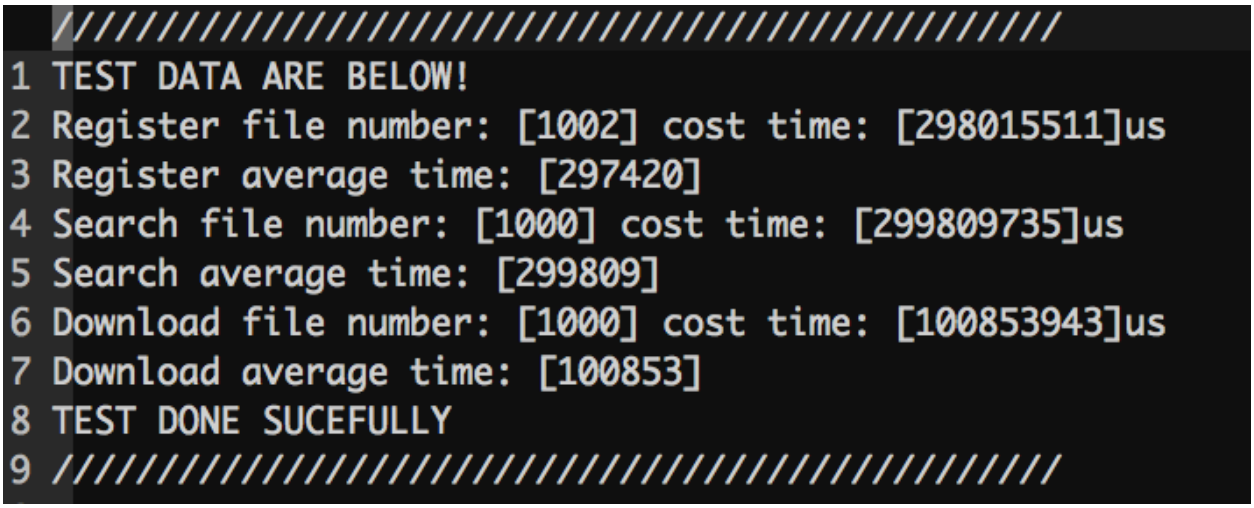
The above shows the average time transfer 100 1KB within 8 nodes.
Note that this represents 1KB/0.5ms = 2MB/s

Conclusion and explain:

1. When the file size is more larger, the throughput would be more closer to the disk written speed.
2. My driver written speed is 300MB/s maximum in best performance, the experiment data is correct considers the network latency. So the 200MB/s is my expect.
3. As the file size become small, the relatively network latency would be dominant, in the third experiment where the file size is 1KB, the throughput would be decreased as 2MB/s.

Experiment set 3: comparing the centralized and decentralized system performance.

As in the assignment 2 and assignment, I have a big performance improvement comparing assignment 1, which is implement the socket connection reuse. This reuse basically eliminate the connection time for each request, which is very dominant. As the data experiment in assignment 1 shows that, basically the connection time is 100ms, so after connection reuse, the communication time is decreased several us.



So for 1KB file, it basically takes 0.1s, thus the throughput is 10KB/S
We can expect that as the file size increase, this time can be amortized.
However, the connection reuse make a great improvement here.