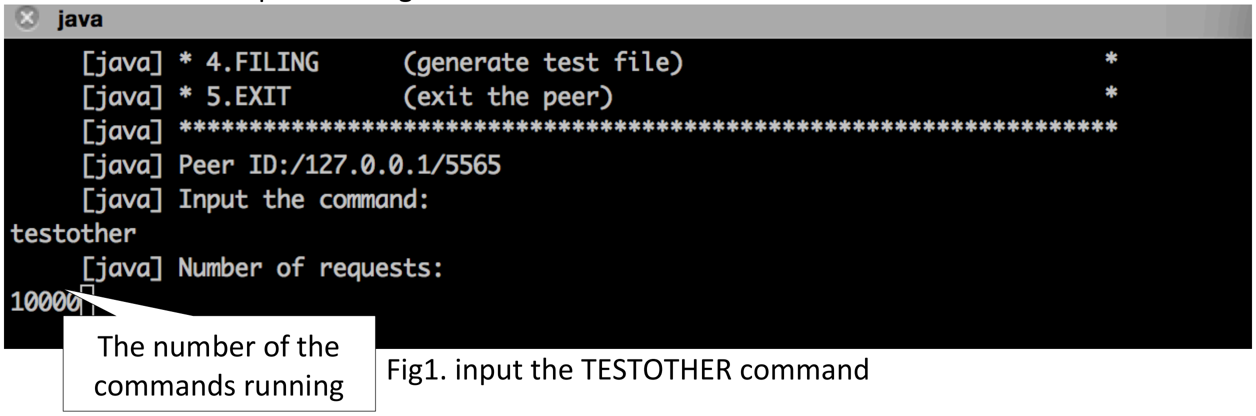
Performance Document

Name: Ning Zhang
CWID: A20336916
Department of Computer Science, Illinois Institute of Technology

1.Performance in Decentralized File Sharing System

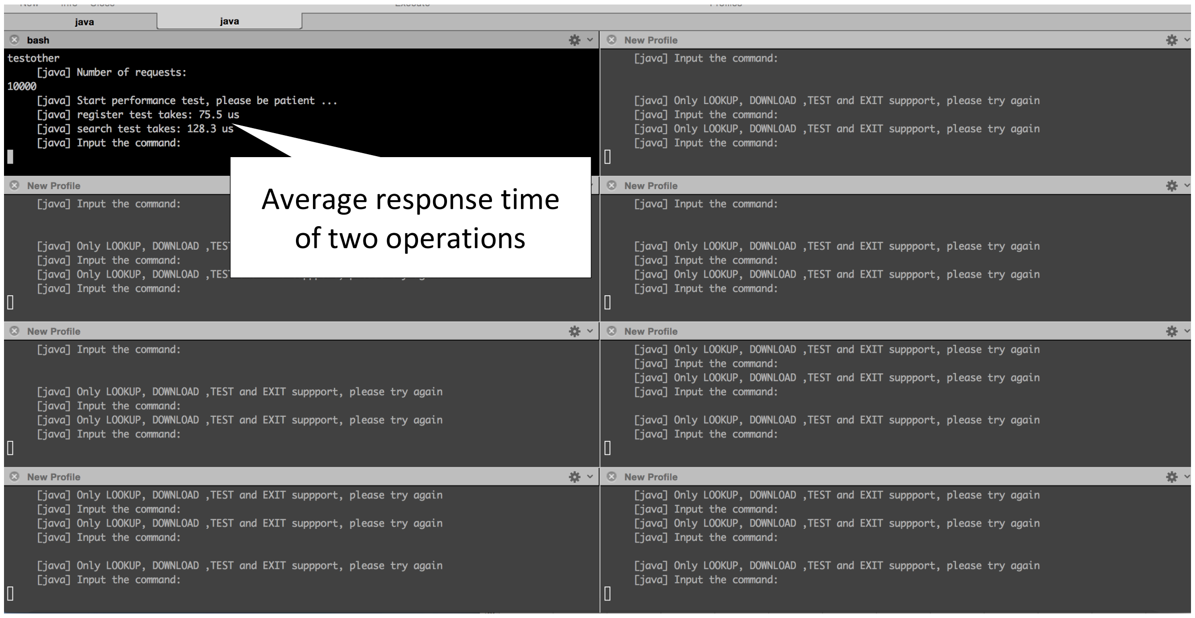
Firstly, we introduce the method of testing the performance. We design several test functions in the source code, which respectively runs 10K REGISTER operations, 10K SEARCH operations by using the TESTOTHER command and 10K OBTAIN operations by using the TESTOBTAIN command, then respectively calculating the response time per one kind of request. And we deploy 8 clients and servers to allow the clients to concurrently execute the commands.

At first, we make 1 client run the TESTOTHER command as Fig1 and output the response time of the two kinds of requests as Fig2.



```
     [java] * 4.FILING      (generate test file)                     *
     [java] * 5.EXIT        (exit the peer)                          *
     [java] *********************************************************
     [java] Peer ID:/127.0.0.1/5565
     [java] Input the command:
testother
     [java] Number of requests:
10000
```

Fig1. input the TESTOTHER command



```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 75.5 us
     [java] search test takes: 128.3 us
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

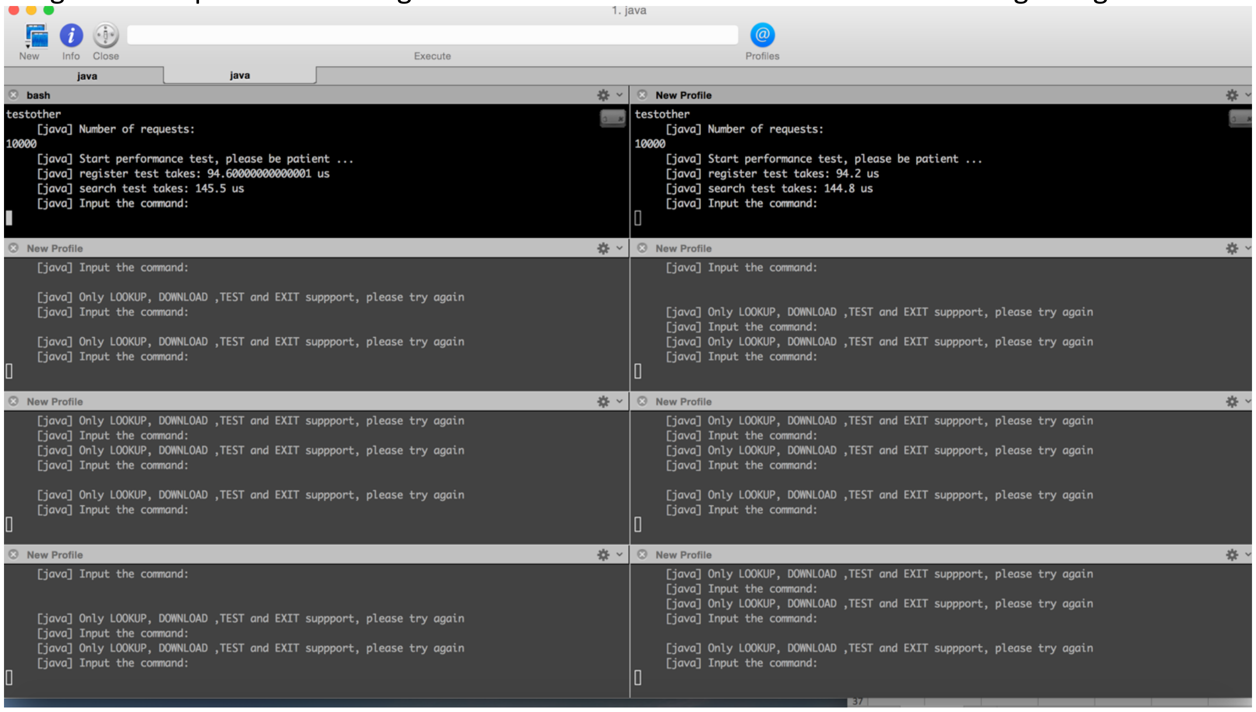
```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

Fig2. one client runs the TESTOTHER command concurrently

And we make two clients run the TESTOTHER function concurrently by using the terminal broadcasting input to make the two terminals concurrently run the TESTOTHER command. Then we get the output results as Fig3. Then we make four clients do the same thing as Fig4.



```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 94.60000000000001 us
     [java] search test takes: 145.5 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 94.2 us
     [java] search test takes: 144.8 us
     [java] Input the command:
```

```
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

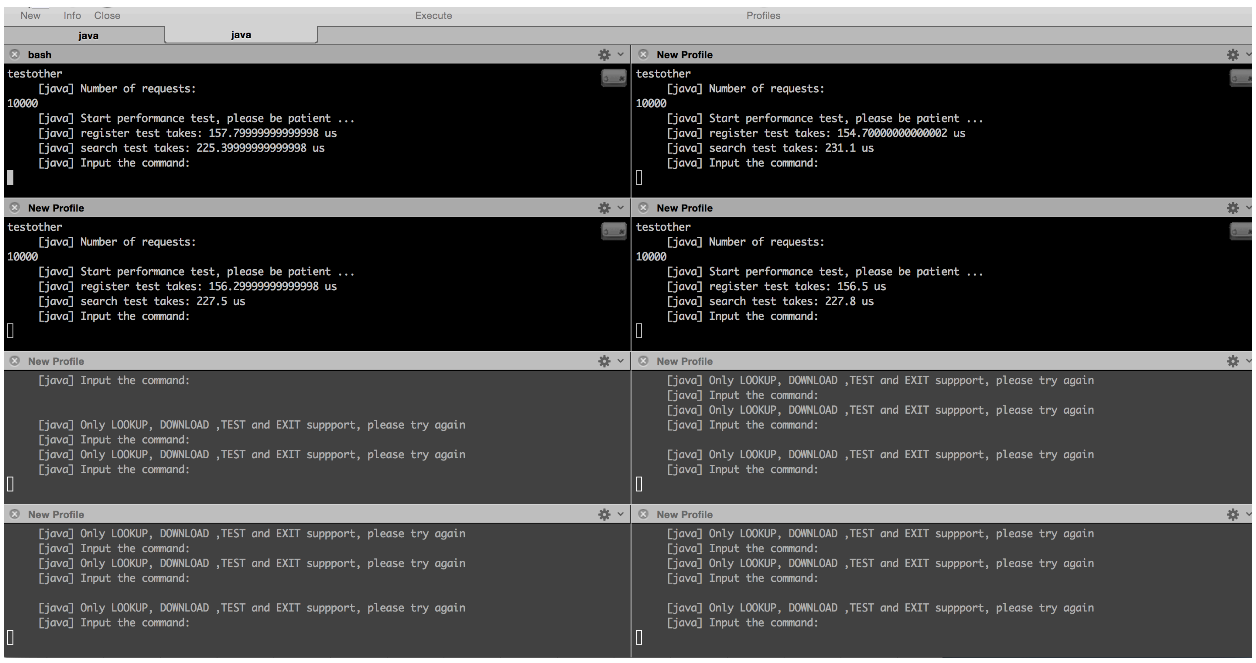
```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

Fig3. two clients run the TESTOTHER command concurrently



```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 157.79999999999998 us
     [java] search test takes: 225.39999999999998 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 154.70000000000002 us
     [java] search test takes: 231.1 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 156.29999999999998 us
     [java] search test takes: 227.5 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 156.5 us
     [java] search test takes: 227.8 us
     [java] Input the command:
```

```
     [java] Input the command:


     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

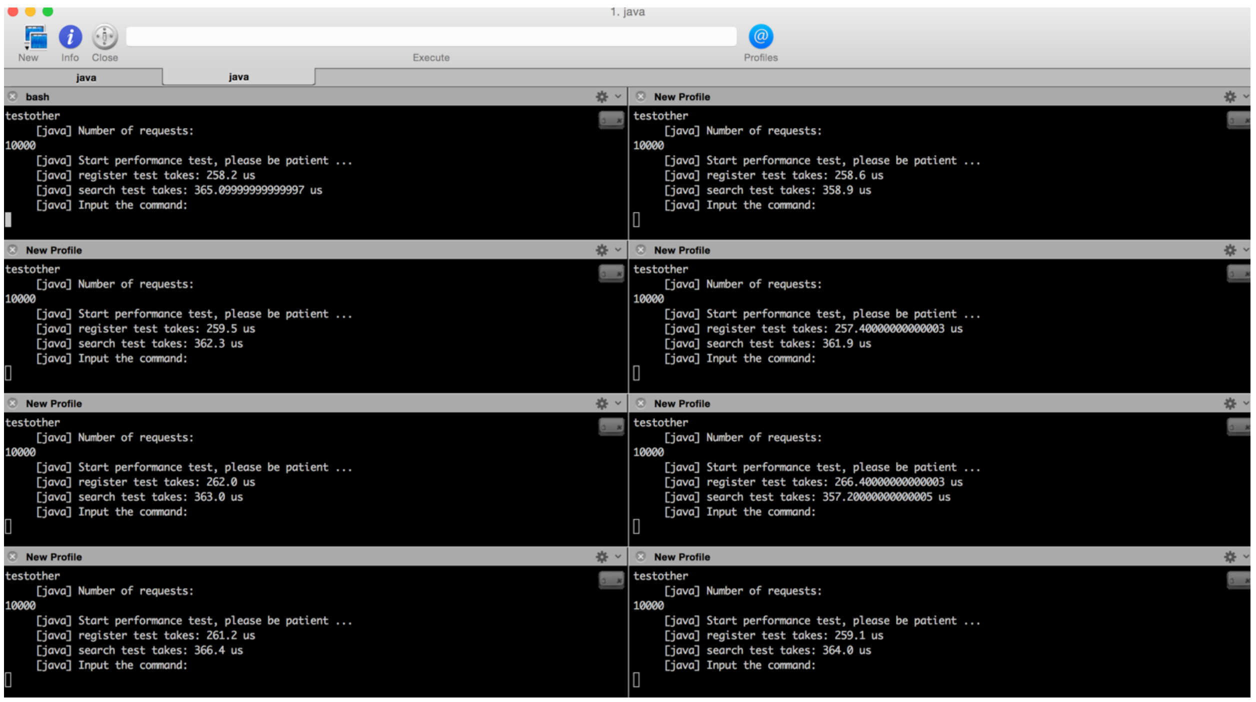
```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

Fig4. four clients run the TESTOTHER command concurrently

In the end, we make eight clients run the TESTOTHER function concurrently by using the terminal broadcasting input to make the eight terminals concurrently run the TESTOTHER command as Fig5.



```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 258.2 us
     [java] search test takes: 365.09999999999997 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 258.6 us
     [java] search test takes: 358.9 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 259.5 us
     [java] search test takes: 362.3 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 257.40000000000003 us
     [java] search test takes: 361.9 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 262.0 us
     [java] search test takes: 363.0 us
     [java] Input the command:
```

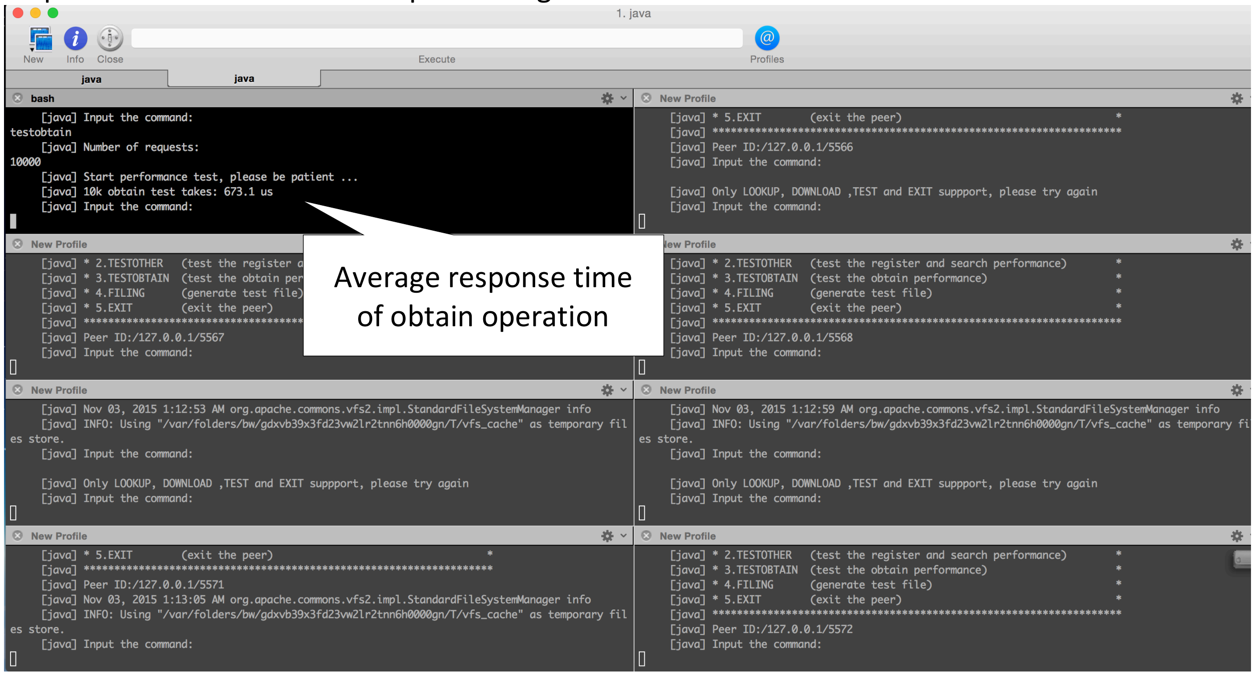
```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 266.40000000000003 us
     [java] search test takes: 357.20000000000005 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 261.2 us
     [java] search test takes: 366.4 us
     [java] Input the command:
```

```
testother
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] register test takes: 259.1 us
     [java] search test takes: 364.0 us
     [java] Input the command:
```

Fig5. eight clients run the TESTOTHER command concurrently

Before we run the TESTOBTAIN function, we must use the FILING command to automatically generate the 1KB files. Then we make 1 client run the TESTOBTAIN command and output the response time of the obtain request as Fig6.



```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 673.1 us
     [java] Input the command:
```

```
     [java] * 5.EXIT        (exit the peer)                                    *
     [java] *******************************************************************
     [java] Peer ID:/127.0.0.1/5566
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] * 2.TESTOTHER   (test the register a
     [java] * 3.TESTOBTAIN  (test the obtain per
     [java] * 4.FILING      (generate test file)
     [java] * 5.EXIT        (exit the peer)
     [java] ***********************************
     [java] Peer ID:/127.0.0.1/5567
     [java] Input the command:
```

```
     [java] * 2.TESTOTHER   (test the register and search performance)       *
     [java] * 3.TESTOBTAIN  (test the obtain performance)                    *
     [java] * 4.FILING      (generate test file)                             *
     [java] * 5.EXIT        (exit the peer)                                  *
     [java] *******************************************************************
     [java] Peer ID:/127.0.0.1/5568
     [java] Input the command:
```

```
     [java] Nov 03, 2015 1:12:53 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
     [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fil
es store.
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

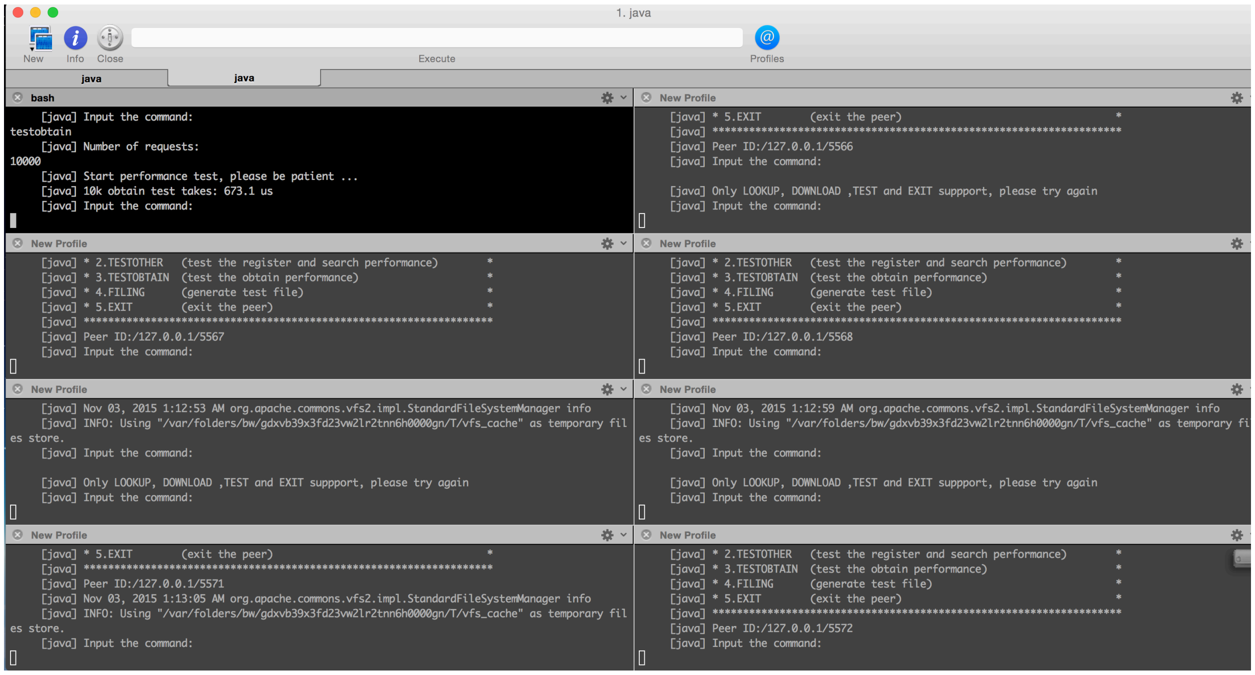
```
     [java] Nov 03, 2015 1:12:59 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
     [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fi
es store.
     [java] Input the command:

     [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
     [java] Input the command:
```

```
     [java] * 5.EXIT        (exit the peer)                                  *
     [java] *******************************************************************
     [java] Peer ID:/127.0.0.1/5571
     [java] Nov 03, 2015 1:13:05 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
     [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fil
es store.
     [java] Input the command:
```

```
     [java] * 2.TESTOTHER   (test the register and search performance)       *
     [java] * 3.TESTOBTAIN  (test the obtain performance)                    *
     [java] * 4.FILING      (generate test file)                             *
     [java] * 5.EXIT        (exit the peer)                                  *
     [java] *******************************************************************
     [java] Peer ID:/127.0.0.1/5572
     [java] Input the command:
```

Fig6. one client runs the TESTOBTAIN command concurrently

And we make two clients run the TESTOBTAIN function concurrently by using the terminal broadcasting input to make the two terminals concurrently run the TESTOBTAIN command. Then we get the output results as Fig7. Then we make four clients do the same thing as Fig8.



```
    [java] Input the command:
testobtain
    [java] Number of requests:
10000
    [java] Start performance test, please be patient ...
    [java] 10k obtain test takes: 673.1 us
    [java] Input the command:
```

```
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5566
    [java] Input the command:

    [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
    [java] Input the command:
```

```
    [java] * 2.TESTOTHER   (test the register and search performance)   *
    [java] * 3.TESTOBTAIN  (test the obtain performance)                *
    [java] * 4.FILING      (generate test file)                         *
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5567
    [java] Input the command:
```

```
    [java] * 2.TESTOTHER   (test the register and search performance)   *
    [java] * 3.TESTOBTAIN  (test the obtain performance)                *
    [java] * 4.FILING      (generate test file)                         *
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5568
    [java] Input the command:
```

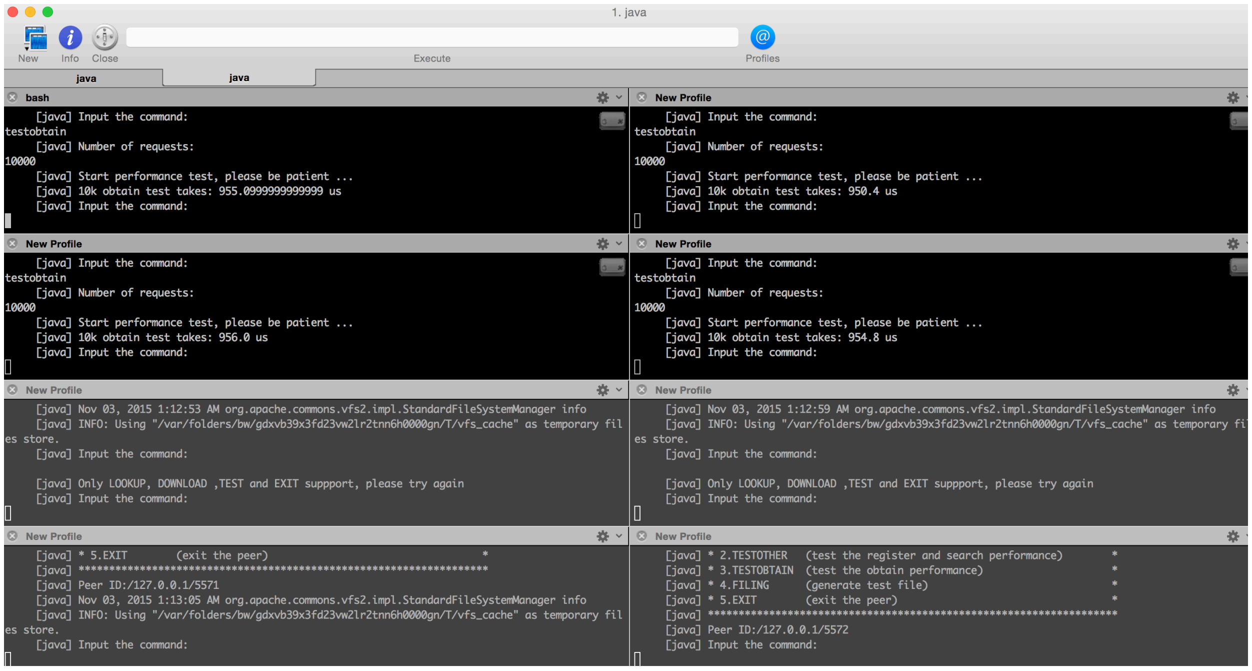
```
    [java] Nov 03, 2015 1:12:53 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
    [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fil
es store.
    [java] Input the command:

    [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
    [java] Input the command:
```

```
    [java] Nov 03, 2015 1:12:59 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
    [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fi
es store.
    [java] Input the command:

    [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
    [java] Input the command:
```

```
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5571
    [java] Nov 03, 2015 1:13:05 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
    [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fil
es store.
    [java] Input the command:
```

```
    [java] * 2.TESTOTHER   (test the register and search performance)   *
    [java] * 3.TESTOBTAIN  (test the obtain performance)                *
    [java] * 4.FILING      (generate test file)                         *
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5572
    [java] Input the command:
```

Fig7. two clients run the TESTOBTAIN command concurrently



```
    [java] Input the command:
testobtain
    [java] Number of requests:
10000
    [java] Start performance test, please be patient ...
    [java] 10k obtain test takes: 955.0999999999999 us
    [java] Input the command:
```

```
    [java] Input the command:
testobtain
    [java] Number of requests:
10000
    [java] Start performance test, please be patient ...
    [java] 10k obtain test takes: 950.4 us
    [java] Input the command:
```

```
    [java] Input the command:
testobtain
    [java] Number of requests:
10000
    [java] Start performance test, please be patient ...
    [java] 10k obtain test takes: 956.0 us
    [java] Input the command:
```

```
    [java] Input the command:
testobtain
    [java] Number of requests:
10000
    [java] Start performance test, please be patient ...
    [java] 10k obtain test takes: 954.8 us
    [java] Input the command:
```

```
    [java] Nov 03, 2015 1:12:53 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
    [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fil
es store.
    [java] Input the command:

    [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
    [java] Input the command:
```

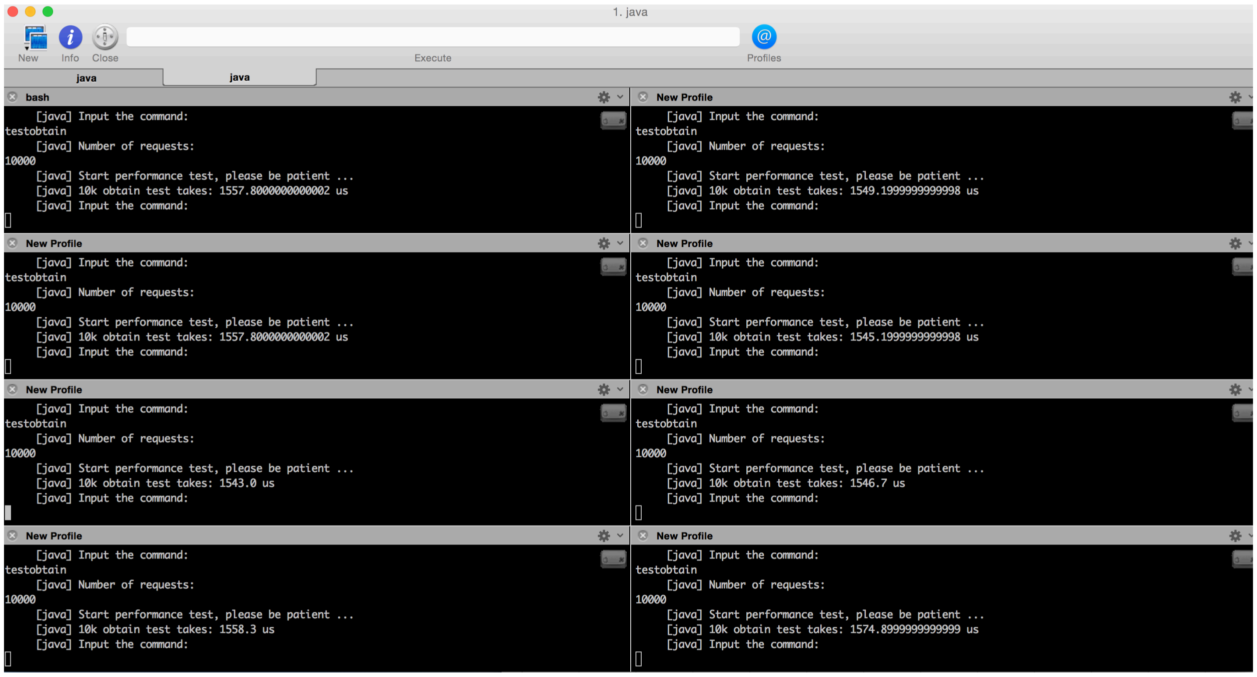
```
    [java] Nov 03, 2015 1:12:59 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
    [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fi
es store.
    [java] Input the command:

    [java] Only LOOKUP, DOWNLOAD ,TEST and EXIT suppport, please try again
    [java] Input the command:
```

```
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5571
    [java] Nov 03, 2015 1:13:05 AM org.apache.commons.vfs2.impl.StandardFileSystemManager info
    [java] INFO: Using "/var/folders/bw/gdxvb39x3fd23vw2lr2tnn6h0000gn/T/vfs_cache" as temporary fil
es store.
    [java] Input the command:
```

```
    [java] * 2.TESTOTHER   (test the register and search performance)   *
    [java] * 3.TESTOBTAIN  (test the obtain performance)                *
    [java] * 4.FILING      (generate test file)                         *
    [java] * 5.EXIT        (exit the peer)                              *
    [java] ****************************************************************
    [java] Peer ID:/127.0.0.1/5572
    [java] Input the command:
```

Fig8. four clients run the TESTOBTAIN command concurrently

In the end, we make eight clients run the TESTOBTAIN function concurrently by using the terminal broadcasting input to make the eight terminals concurrently run the TESTOBTAIN command as Fig9.



```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1557.8000000000002 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1549.1999999999998 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1557.8000000000002 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1545.1999999999998 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1543.0 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1546.7 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1558.3 us
     [java] Input the command:
```

```
     [java] Input the command:
testobtain
     [java] Number of requests:
10000
     [java] Start performance test, please be patient ...
     [java] 10k obtain test takes: 1574.8999999999999 us
     [java] Input the command:
```

Fig9. eight clients run the TESTOBTAIN command concurrently

Through the results of tests above, we can get the performance of these three operations as Fig10.

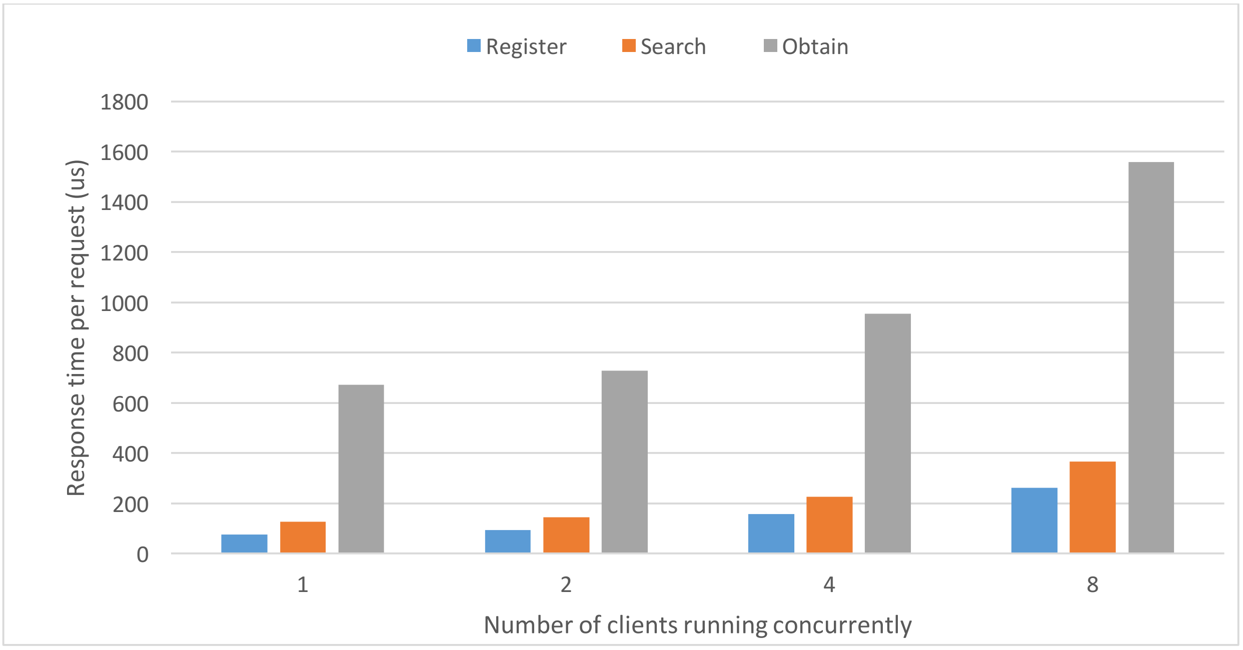


Fig10. Performance of three operations in Decentralized file sharing system

In this figure, X-axis is the response time per request; Y-axis is the number of clients running concurrently. The blue bar is the response time of one Register operation. The orange bar is the response time of one Search operation. The gray bar is the response time of one Obtain operation to download one 1KB file. We can see that the more clients run concurrently, the more response time is spent by the operation. It is because the more clients run concurrently, the more source of the pc is spent. And the more clients run concurrently, the network has more congestion. And the competition will increase when the more clients run concurrently. So the response time of one operation increases as the number of clients running concurrently increases. When there is one client running, the Obtain operation cost the most time, because the Obtain operation needs to download one 1KB file from the network and it has 1KB write operations in the filesystem. And the cost time of one Search operation is more than that of one Register operation, even though the Register operation has two write operation to hash table and the Search operation has one read operation to hash table. Because the Search operation must return the search results to the client which is another network communication.

2.Throughput in Decentralized File Sharing System

Then we conduct a set of experiments to measure throughput we can achieve with different file sizes. We deploy 8 servers and 8clients as the demand of PA3. First, we register a variety of files sizes, ranging from 1KB, 10KB, 100KB, 1MB, 10MB, 100MB, and 1GB in size which are automatically generated by the FILING command. We make sure every fileserver has 100 files. Then we make all clients to obtain these files with different sizes respectively for 1 minutes. And we calculate throughput in bytes per second of all the clients put together as Table 1.

Table1. Throughput in Decentralized file sharing system

File Size	1KB	10KB	100KB	1MB	10MB	100MB	1GB
Throughput	333.3KB/s	5MB/s	14.3MB/s	50MB/s	151.2MB/s	250.6MB/s	299.7MB/s

Through the table, we can see that the throughput is increased by the increment of file size. When we obtain the file, we must get the file size first. In 1 minutes, the smaller size of file we obtain, the more files we get. Then we must obtain lots of file size before we get the files with smaller size. The throughput will reduce because of the cost of getting lots of file size. And the number of file operations like newfile(), openfile() and closefile() when we obtain files with smaller size is more than that when we obtain files with larger size. And the file with large file can fully use the aptitude of TCP package.

3.Comparison Between Decentralized and Centralized FileSharingSystem

We use the program of the PA1 homework which is a Centralized file sharing system to run these same tests as those above. And we compare these results with the results of testing the

Decentralized file sharing system. Then we can get the comparison results of the Register performance between these two systems in Fig11.

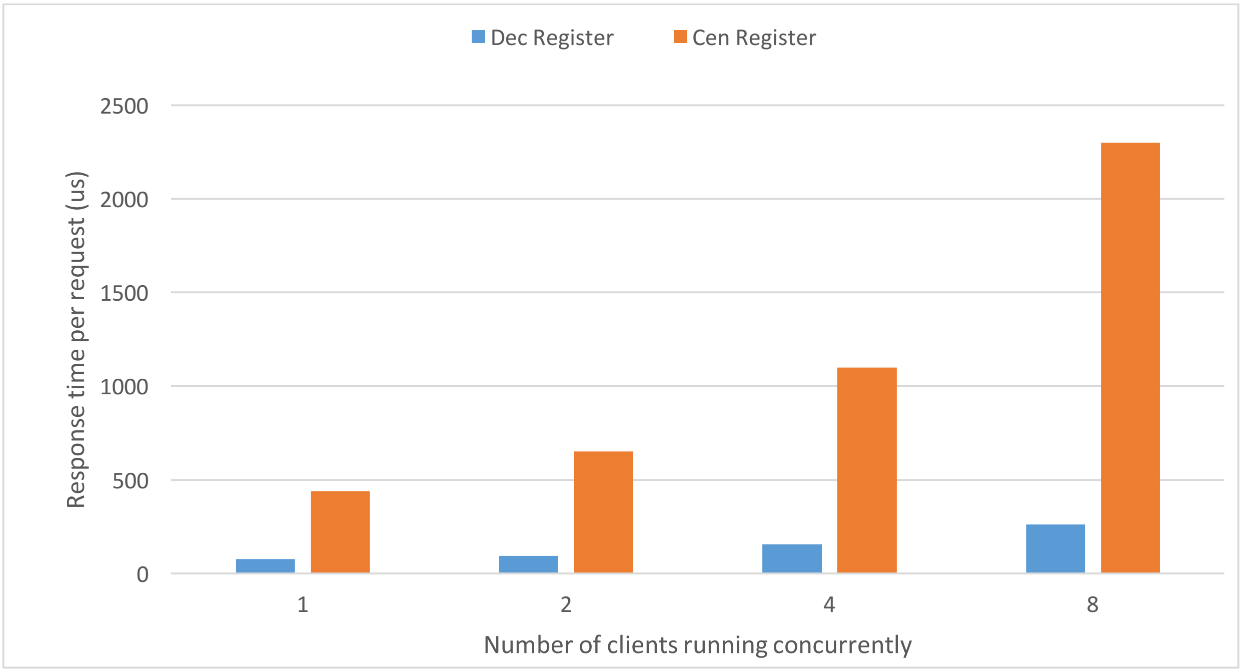


Fig11. Register performance between these two systems

In this figure, X-axis is the response time per request; Y-axis is the number of clients running concurrently. The blue bar is the response time of one Register operation in Decentralized file sharing system. The orange bar is the response time of one Register operation in Centralized file sharing system. We can see that the orange one is more that the blue one. Because the Decentralized file sharing system has multiple indexservers but the Centralized file sharing system only has one indexserver. There are so many requests in the Centralized file sharing system so that one indexserver can not handle these immediately. So the response time per request will increase.

Then we get the comparison results of the Search performance between these two systems in Fig12.

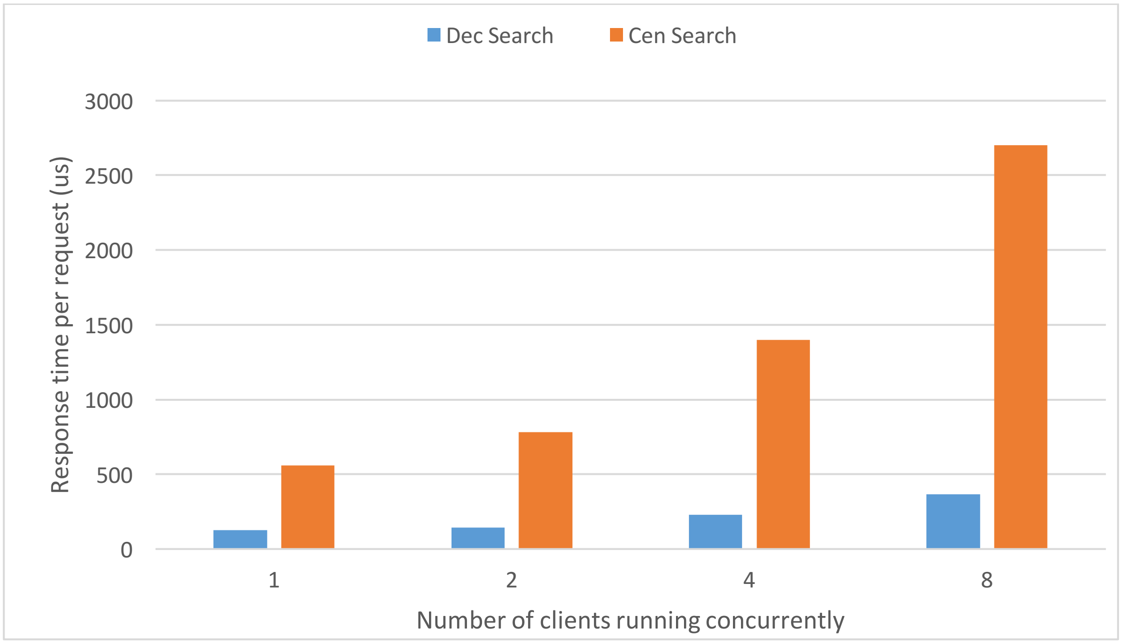


Fig12. Search performance between these two systems

In this figure, X-axis is the response time per request; Y-axis is the number of clients running concurrently. The blue bar is the response time of one Search operation in Decentralized file sharing system. The orange bar is the response time of one Search operation in Centralized file sharing system. We can see that the orange one is more that the blue one. Because the Decentralized file sharing system has multiple indexservers but the Centralized file sharing system only has one indexserver. There are so many requests in the Centralized file sharing system so that one indexserver can not handle these immediately. So the response time per request will increase.

Then we get the comparison results of the Obtain performance between these two systems in Fig13.

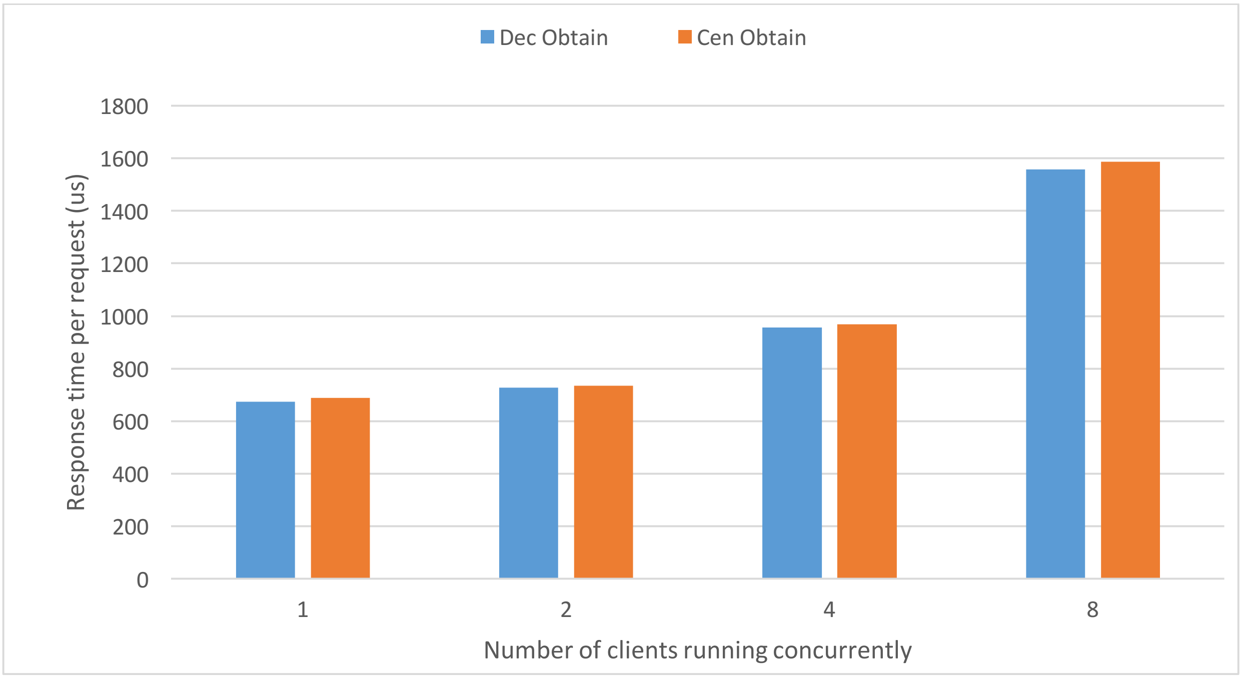


Fig13. Obtain performance between these two systems

In this figure, X-axis is the response time per request; Y-axis is the number of clients running concurrently. The blue bar is the response time of one Obtain operation in Decentralized file sharing system. The orange bar is the response time of one Obtain operation in Centralized file sharing system. We can see the blue one is almost the same as the orange one. Because the obtain operation do not need to communicate the indexserver but the communication between fileservers in both systems.

Then we get the comparison results of the throughput between these two systems in Fig14.

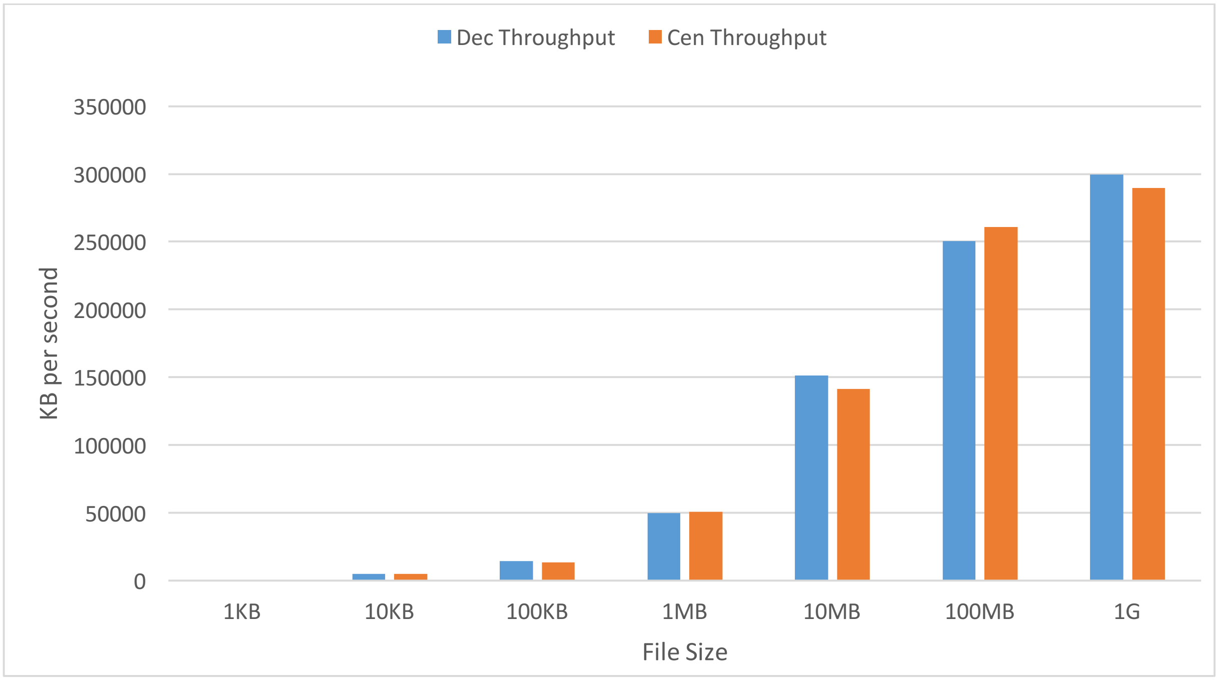


Fig14. Throughput between these two systems

In this figure, X-axis is the response time per request; Y-axis is the number of clients running concurrently. The blue bar is the throughput for different files with different sizes in Decentralized file sharing system. The orange bar is the throughput for different files with different sizes in Centralized file sharing system. We can see the blue one is almost the same as the orange one. Because we use obtain operation to measure the throughput an d the obtain operation do not need to communicate the indexserver but the communication between fileservers in both systems.