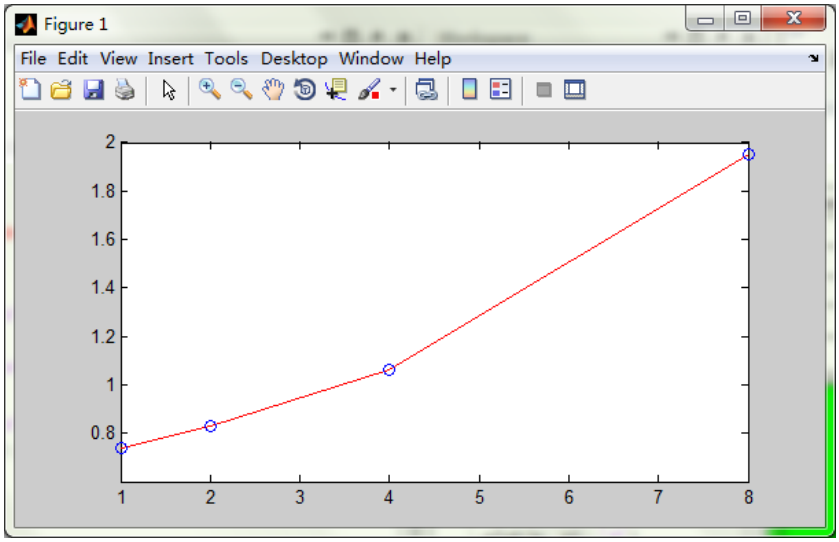
1, Operations evaluation and measurement

Deployed 8 servers. They are setup on the same machine (different directories).
Perform an evaluation on running 10k operations per client, and test all operations (registry, search, obtain).
The first experiment have 1 client doing this experiment. Then 2 clients concurrently. Then 4 clients. And finally 8 concurrent clients.
The aggregate achieved throughput in operations per second of all the clients put together has been computed.
Also, the average response time per operation request is computed by measuring the average response time seen by a client.

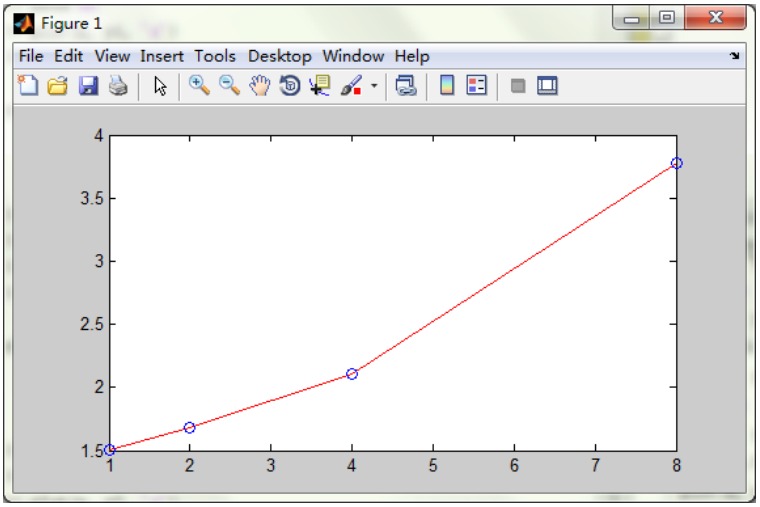
- **registry**

clients	response time (ms)
1	0.74
2	0.83
4	1.06
8	1.95



- **search**

clients	response time (ms)
1	1.51
2	1.68
4	2.11
8	3.78



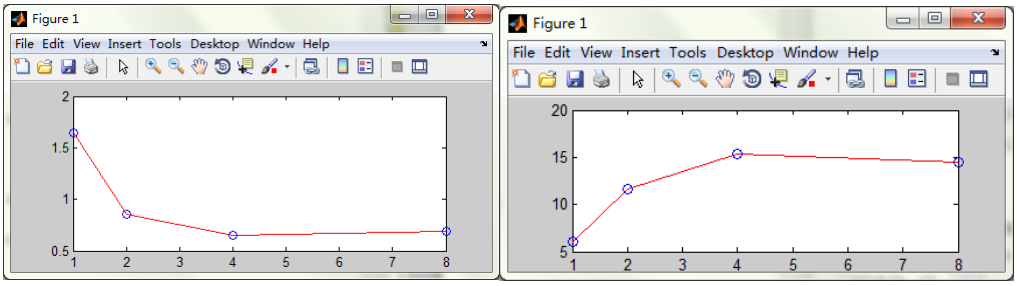
Explanation:

As the number of clients increases, it takes more time per request That is because the method of indexing server is synchronized, more time will be taken on getting the lock resource.

- **obtain**

The test download file size is 10KB

clients	response time per client	throughput(MB)
1	1.64	6.1
2	0.86	11.6
4	0.65	15.4
8	0.69	14.5



Explanation:

When the clients number is less than 8, the response time decreases when the clients number increases. But when the clients number goes up to 8, the response time increases. The throughput is just opposite.

I think that is because when clients number isn't that much, the multithread improves the transport efficiency, but when the number of threads gets too many, it takes more time in thread switch, that increases the response time and so is the throughput.

2, Centralized VS decentralized indexing server

version	response time
Centralized	1.15
Decentralized	0.74

Explanation:

Decentralized indexing server gets less response time over the centralized version. Because it has more than one peers, the registry and search operation can be dispatched to different peer, thus decreasing the response time.