Distributed Proxy Server

FAIZAN SAFDAR ALI (17100152)

1 INTRODUCTION AND IMPLEMENTATION

A proxy server is an intermediary between a web server and a client, that is, the proxy server receives requests for pages from different clients that would normally be served by different web servers. Note that the server can be another proxy server. We needed to support only GET. A GET request that is addressed to a proxy server has to have an absolute URI (instead of a relative URI). e.g. http://serverName/fileName. When web server runs on the same machine as proxy server, it is inefficient to transmit files using sockets. Instead, two processes can use shared memory to pass file efficiently. We implemented shared memory optimization in addition. My program is capable of enabling/disabling this feature.

For this assignment, I used the pthread library and server/client libraries for making server and server-proxy, I used python for making the client code for testing. I tried to answer the following questions.

- Is using shared memory faster or slower?
- fi Is it a bandwidth issue or a latency issue? (Think about the kinds of experiments you need to perform to separate the two.)
- Compare the performance of the web server and proxy server pair to just the web server. How much slower does the proxy server make the system?

2 TESTING

For testing and getting the data for the above mentioned questions, I used the python code to make the client (named client.py). I sent the files of different sizes using different parameters and found the following results.

3 EXPERIMENTATION AND REPORTING RESULTS

Many experiments were run and following are the results I obtained after the experiments:

3.1 Is using shared memory faster or slower?

I sent the files of different sizes from the server to the client with and without the shared memory. Following are the results I got.

File size(bytes)	Time(without SM)	Time(with SM)
10000	0.010674	0.00040
20000	0.01944	0.0005619
100000	0.02328	0.0006549
200000	0.01830	0.000771
500000	0.38650	0.81180
1000000	0.41453	1.0004
2000000	0.75831	1.12532
5000000	3.98825	1.35487
10000000	17.11707	13.8958

and the graph I get is:

Fig. 1. With vs Without SM. Green line shows without SM and red shows with SM

We can see that for smaller files, SM takes more time than socket. This maybe because of the synchronization over head and the shared memory handling by operating system. For larger files, shared memory has significantly high efficiency than the socket one.

3.2 Is it a bandwidth issue or a latency issue?

I think it's a bit of the both. The socket connection efficiency is both effected by the limited bandwidth and high latency. We can alot as many shared memory as we want and the latency of accessing the shared memory is very low as compared to the socket connection.

3.3 Performance of the web server and proxy server

I downloaded the files of various sizes from the server through the proxy server and through the direct connection. Following are the results I got.

File size(bytes)	Time(proxy without SM)	Time(proxy with SM)	Time(Direct Connection)
10000000	11.683	0.3068	0.06544
20000000	26.117	0.34167	0.1566
30000000	27.438	0.379592	0.1843
40000000	50.185	0.45176	0.2434
50000000	53.769	0.40076	0.2973
60000000	56.4537	0.5224	0.3023
70000000	114.897	0.7725	0.479580

and the graph I got is:

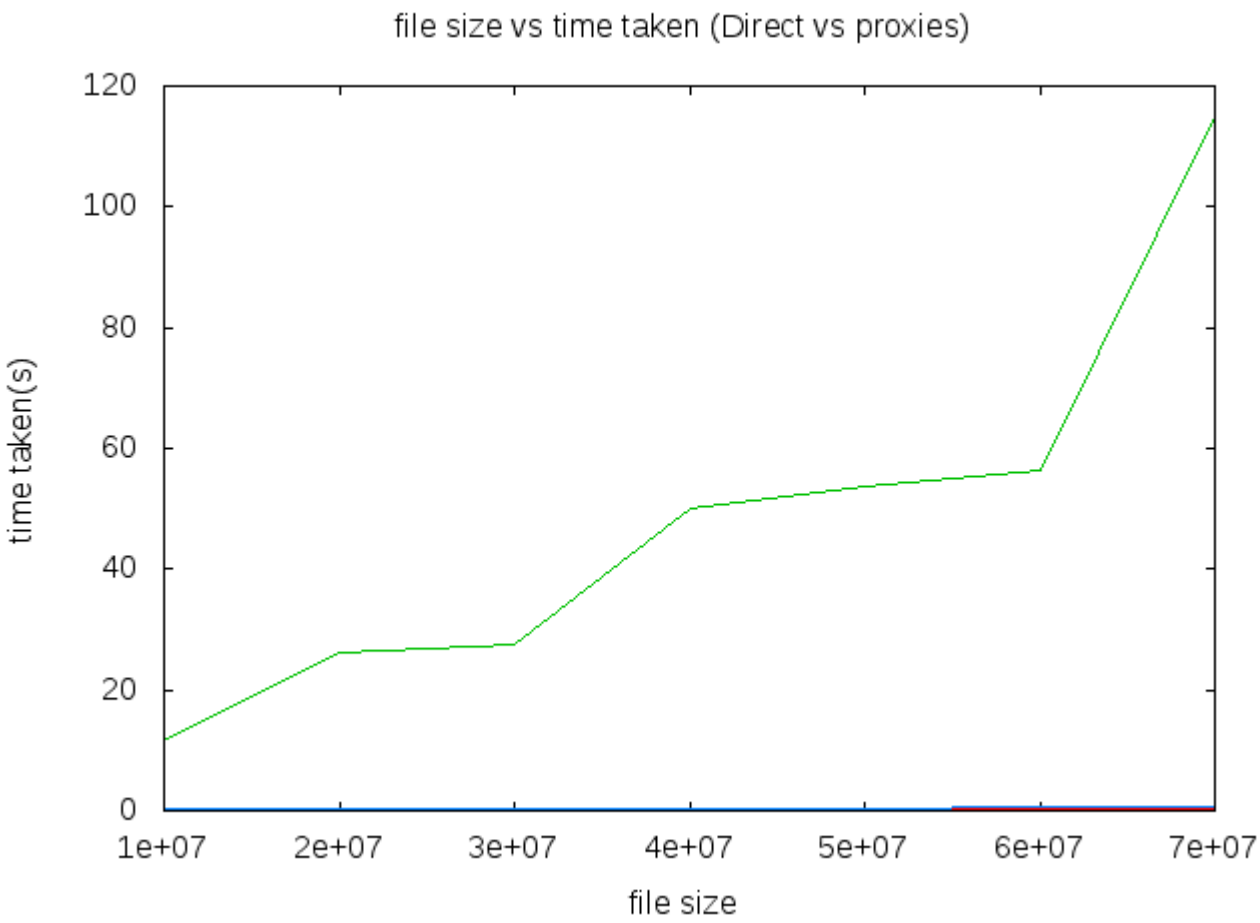

Fig. 2. Green line shows proxy without SM, blue is proxy with SM and red is the direct connection

Clearly, the proxy reduces the efficiency specially if there is no shared memory.

4 CONCLUSION

So from above we can conclude that:

- It is better to have a shared memory for the large files sharing. Also the shared memory should be large enough so that the overhead time of shared memory is balanced with increase in efficiency.
- The socket is slower because of limited bandwidth and latency. Also, there are other issues of buffer size available.
- Proxy server can increase the latency by the large amount and can be ineffective.
- If we implement the cache at the proxy side, it can be very helpful. (I didn't implemented it in this assignment but I think the reult will be great.)