Abstract

In our daily life, it is very frequent for us to make use of file sharing system, like Google Drive. After studying the CAN201 'Introduction to Networking', the coursework is released in week 5. In this CW1, we are required to design a Large Efficient Flexible and Trusty (LEFT) Files sharing program which must meet four basic requirements. In Introduction and Methodology section, I will show how I search the source to identify how to do it. In Implementation section, I will use flow chart to demonstrate the steps of my program. In results and conclusion section, I will show the result screenshot of my successful program and make a conclusion of this coursework, and give further plan of it.

Catalogue

1. Introduction

1.1 Project requirement

This project requires to develop a file sharing program which must meet four standards. Initially, large means the system could transport both any different files and folders that less than 500MB, excepting hidden files and folders. Additionally, the files/folders should be synchronized automatically while IP addresses are set as an argument. In the meantime, the speed for transmitting should be fast as soon as possible. Importantly, this system has to achieve resume function when interrupted. Finally, the program must have no error for any files. Remember that Application layer protocol need to be designed personally while the Transport layer protocol using TCP, and we could just use the python modules give in project PDF.

1.2 Background and literature review

File sharing is the practice of distributing or providing access to digital media, like computer programs and multimedia. Take Google Drive as an example, it is a cloud-based file syncing and sharing services that allow users to create particular folders and synchronize them [1]. As for layer protocol, for one part (Application layer protocol), we could construct a socket structure to transmit data and call function of the socket. Data transmission would be finished when the request node and response node setting up [2]. For another part (TCP), it is more reliable, byte-stream. Using TCP for socket programming, Client must contact server by creating TCP socket, specifying IP address and port number of server process. Furthermore, TCP provides reliable, in-order byte-stream transfer ("pipe") between client and server. After have seen several related topics IEEE papers, I begin to design my own LEFT file sharing program.

1.3 What I did

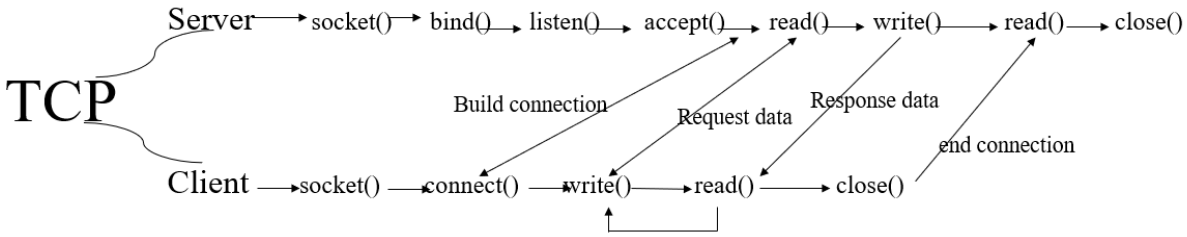
I search on Internet about related topic passages and papers and labs' Powerpoint of CAN201, including IEEE, GitHub etc. to identify how to develop my own file sharing program. Then I design application layer protocol, make use of TCP and socket program to establish my system. Finally, I wrote this development report. The

more details about what I did would be demonstrated in the Implementations section.

2 Methodology

2.1 Proposed protocols

As required that we should make use of existing application layers like HTTP, I designed the application layer protocol and use TCP for transport layer protocol. As for package, I also design the header body which you can find in my code. The first four bytes is the length of header and the last four is the length of body. I use the header length to detect the files.



2.2 Proposed functions and ideas

There are a lot of functions I used. Here, I would like to introduce several functions that I used in my code. For os module, I use it judge whether the files are existing or not and the size of the files. For argparse, I use it to add arguments for my program. For time and math, it is easy to know they are used for time control and math calculation. In addition, I use hashlib module to transform input of arbitrary length by some hash algorithm into the output of fixed length (hash value), including MD5 algorithm. Lastly, I use tqdm module to create a progress bar.

As for ideas, the requirements are in the project PDF. I want to achieve some functions then I go to the Internet and courseware Powerpoint to find what I need. We are not allowed to use any mature application layer protocol, so I search how to design application layer protocol in the Internet. Comparing with the last year project (using UDP), we use TCP for Transport layer protocol. For reliable perspective, it has its advantage. About detecting files, I separate four statuses. Different number means different operations that the PC_A need to do. When comparing the files,

another PC just need to traverse the header file instead of the whole files, which could decrease the time for running.

3 Implementation

3.1 Steps of implementations

1) Identify the project requirement of the coursework.

2) Find related source to use

3) Divide the whole coursework to several blocks and achieve every blocks functions, respectively. Draw a chart of what we need to do

4) Search in the Internet to deal with the blocks, learn how to use these functions in Python.

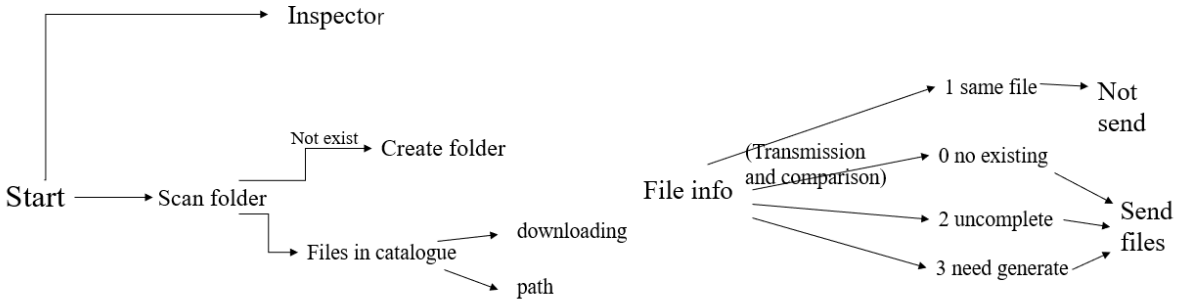
5) Attempt to write the codes to achieve functions respectively and test them if I succeed, RUN_A, RUN_B, MD5_1B. (More details would be in next section)

6) Attempt to improve the algorithm to decrease the running time.

7) Optimize the code format to make it clear and easy to read.

8) Record the result and write the report.

3.2 Program flow charts



3.3 Programming skills

OOP(Object-Oriented Programming): Object Package for packaging, and add header and body of the files.

Multithreading: In this CW, the program starts, the thread will divide into multiple to execute functions.

Module/Block ideas: Although I do not separate several modules in my codes, I just write in main.py. However, I use modular ideas when I initially design the program

(different blocks achieve different functions).

3.4 Difficulties and Solutions

To tell you the truth, I began to do this project during the week 7 because ICS has no Midterm. Initially, I could just finish the first easy task which has been taught in lab 3 and lab 4. I didn't know how to finish the rest of the task. In that case, I searched useful materials and read it, including from Internet and books. After that, I discussed with my schoolmates. Then I knew how to do the second task.

As for the resume from interruption, means all the files should also be synchronized when shared during its downtime. I still confused about it. When the PC_A killed, the connection is refused, the program need to detect again. After that, I watched the related videos in Bilibili and Youtube to solve it.

4 Testing and results

4.1 Testing environment

My computer is : Dell G3 3590 ,win 10， py3.8

The virtual box is 6.1.30 (latest version) tinycore Linux

4.2 Testing plan

Firstly, I wanted to confirm whether I complete all the tasks successfully. I ran my program to ensure it had no error. I would show you one of the successful running screenshots below.

After that, I wanted to test my code in different computers to know whether the type of the computer would affect the result of running time. I asked Feimax about this question, he said he would test on his computer. Hence, the type of the computer is not influence factor. It is useless to do this.

Following that, I wanted to find whether the different block sizes would affect the result or not. Initially, I set the original block size as 1024*1024*5 (5mb), then I clicked the run for the program. To reduce the error, I ran 10 times to calculate the average data for the program.

Next, I changed the block size to do the test (control variate method). I set 2mb,

10mb and 20mb to compare with 5 mb for their running time. I also ran 10 times for each size.

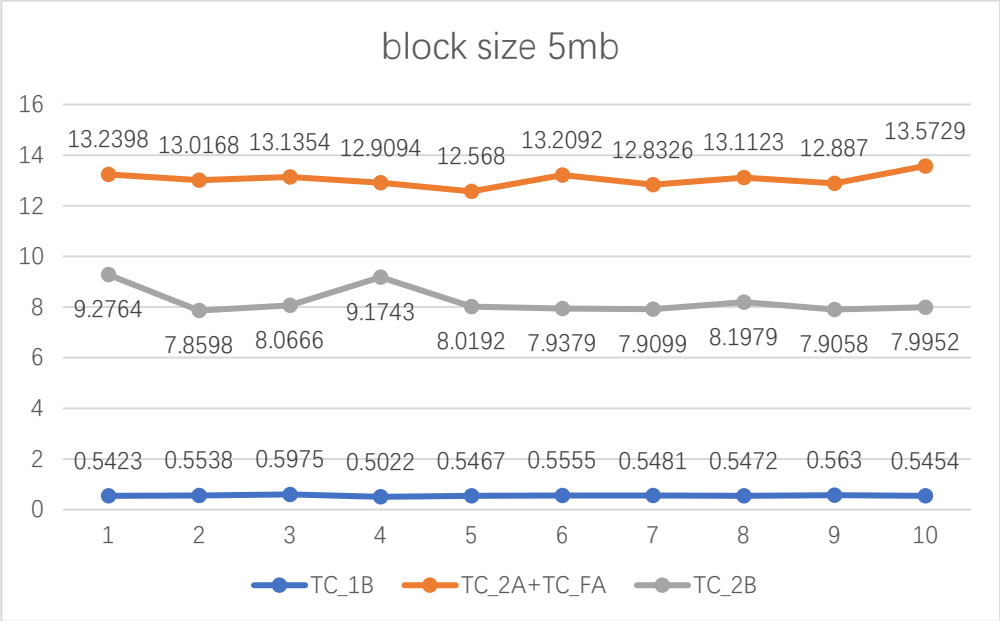
Finally, I found the better block size during 2mb, 5mb, 10mb and 20mb. Then I compare it with near numbers, 4mb and 6mb. Each size I ran 5 times, then to conclude that the best block size for running fastest is approximately 5.

4.3 Testing results

One of the successful running screenshots is below that you can see it clearly.

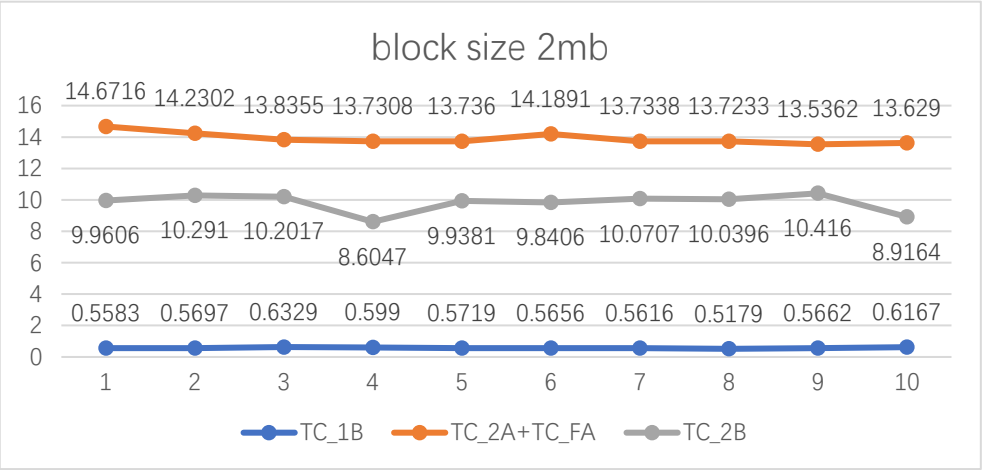
```
MD5_2B: PASS
Result: {'RUN_A': True, 'RUN_B': True, 'MD5_1B': True, 'TC_1B': 0.5538408756256104, 'MD5_2A': True,

Process finished with exit code 0

'MD5_2A': True, 'TC_2A+TC_FA': 13.016807794570923, 'MD5_FA': True, 'MD5_2B': 1, 'TC_2B': 7.859887361526489}
```

Then I will show you the average time of block size 5mb.

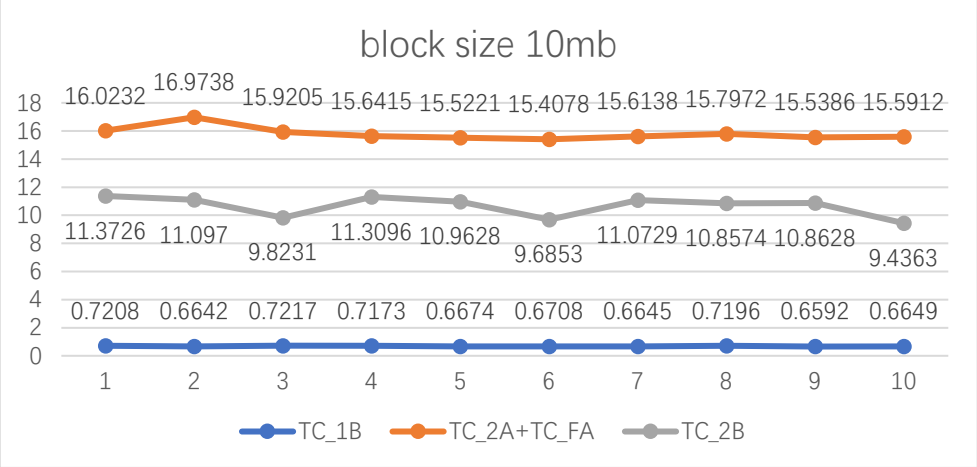


The diagram is above. The data is almost a straight line. It fluctuates, but it's basically stable that you can see.

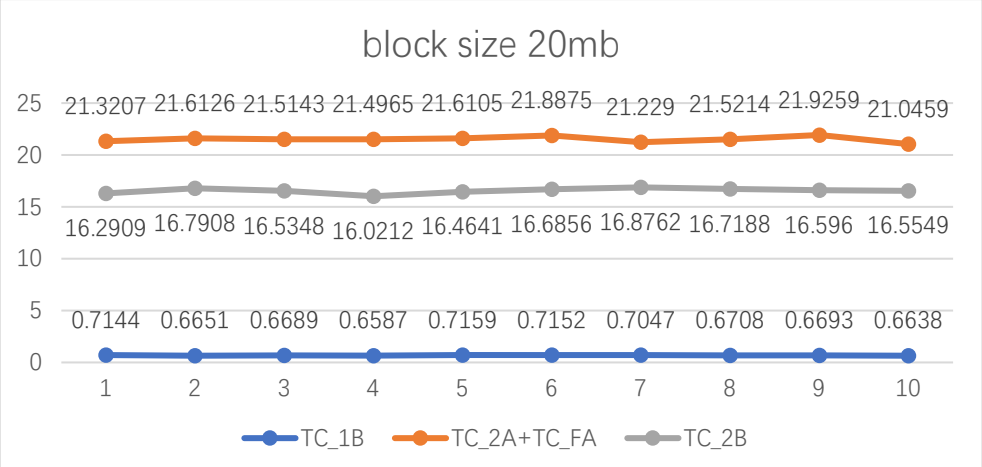
As for the running time, the average time of TC_1B is 0.55017s, TC_2A+TC_FA is 13.04834s, TC_2B is 8.2343s.



For block size 2mb: the average time of TC_1B is 0.57598s, TC_2A+TC_FA is 13.90155s, TC_2B is 9.82794s.



For block size 10mb: the average time of TC_1B is 0.68704s, TC_2A+TC_FA is 15.80297s, TC_2B is 10.64798s.

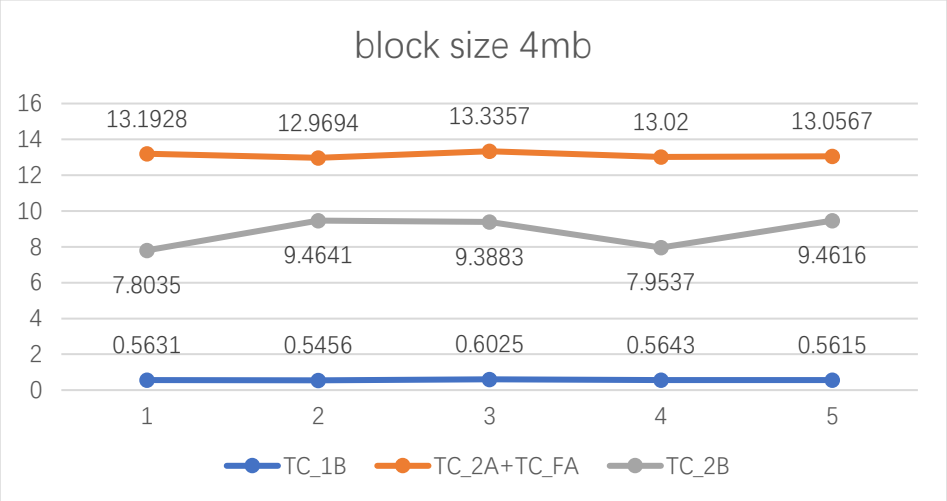


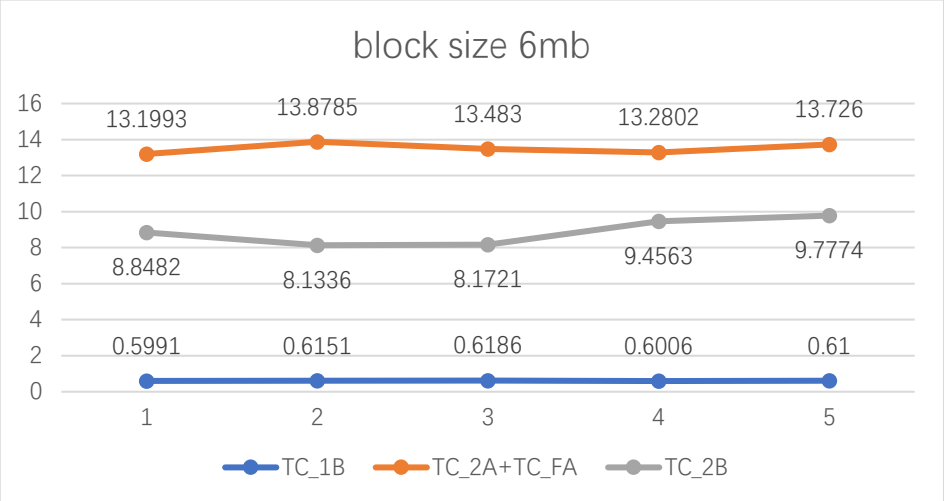
For block size 20mb: the average time of TC_1B is 0.68468s, TC_2A+TC_FA is

21.51643s, TC_2B is 16.55333s.

From the data, it is not difficult to find that when the block size is 5mb, it is better than 2mb, 10mb, and 20mb.

Then we test the nearly numbers of 5, 4 and 6.





The data is still like a straight line. It fluctuates, but it's basically stable.

From the charts above, the average time is almost the similar while 5mb is faster just a few milliseconds. Hence, the better block size for running faster is approximately block size 5mb.

5 Conclusion

5.1 Conclusion what I did

In conclusion, I designed my own application layer protocol for file synchronization,

designed the system for file synchronization, and implemented it in python. This program meets the requirement of the coursework. And I change the block size to find out the better block size for running faster of this program is approximately 5. It is highly possible that my program is not the best for this coursework 1. However, I spare no effort to do this CW and I succeed in completing it.

5.2 Future plan

For running time speed ranking, I might not rank top 10%. I'd like to change the algorithm to decrease the running time.

6 Reference

[1]: A. Sarkar and N. Prakash, "File sharing system encapsulated with customized social networking and learning management system," 2015 International Conference and Workshop on Computing and Communication (IEMCON), 2015, pp. 1-7, doi: 10.1109/IEMCON.2015.7344450.

[2]: Y. Ma, L. Wang, J. Cui and B. Zheng, "Design of P2P Application Layer Protocol," 2016 12th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), 2016, pp. 175-178, doi: 10.1109/MSN.2016.036.