Assignment 3 COL334

Lokesh Acharya 2018CS10352

25 November 2020

Part 1

Download data in one go form **norvig.com**
Normal TCP connection on port 80

```
   part1()
1   - create socket
2   - connect to the host via socket
3   - send GET request
4   - while true
5        - receive, decode data and append to global string variable
6   - remove HTTP header
7   - write data into a text file
8   - calculate MD5 of downloaded data
9   - compare calculated MD5 to original MD5
```

Received file byte by byte
Time taken $\approx 20sec$

Part 2

Download data in one go form **norvig.com**
Normal TCP connection on port 80
Connection : keep-alive

```
   part2()
1   - create socket
2   - connect to the host via socket
3   - send GET request
4   - while true
5        - receive, decode data and append to global string variable
6   - remove HTTP header
7   - write data into a text file
```

We have downloaded the data in Range 0-99

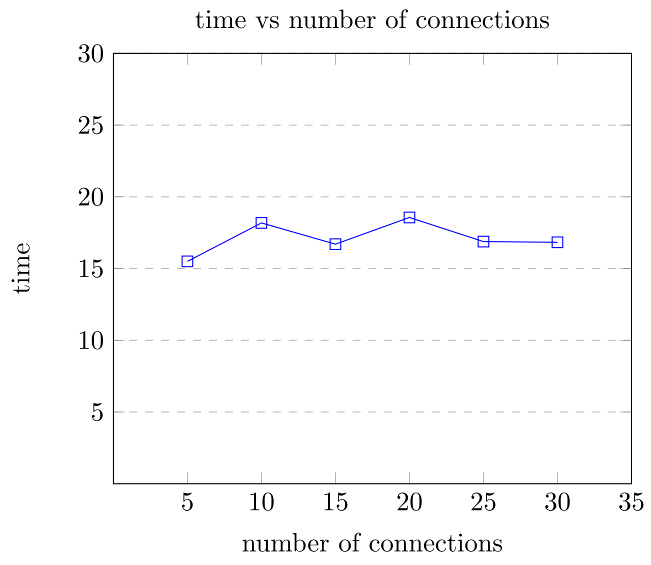
Part 3

```
part3()
1   - create class tcp_thread
2      initialization:- takes in start byte and creates a tcp socket
3      run:- connects to socket and send sequential GET requests
            and collects data
4   - initialize an empty list L to store threads
5   - for i in number of connections
```

```
6       initialize a thread and append to the L
7    - for t in L
8       start thread t
9    - join threads
10   - remove HTTP headers from stored data
11   - calculate and print MD5 sum of downloaded data
```

input = number of threads to run
BLOCK_SIZE = size of chunk to be downloaded
THREAD_LIFE = number of send requests a thread send
SIZE = size of the file to download

As the number of parallel connections is increased the download time do not vary much, possibly because of the fact the we have to wait for all threads to join before reassembling them.



The download time should further decrease with different sources to download, because we can then reduces the load on a single bottleneck link to multiple different bottleneck links to increase our download speed.
One server maybe be faster because the it may have less traffic then the other, we can modify our program to download more data to download from faster server

Part 4

```
part4()
1    - create class tcp_thread
2        initialization:- takes in start byte and creates a tcp socket
3        run:- connects to socket and send sequential GET requests
                 and collects data
4    - initialize an empty list L to store threads
5    - for i in number of connections
6        initialize a thread and append to the L
7    - for t in L
8        start thread t
9    while data not downloaded
10       check if some connection is lost
11           restart that connection form the point it is lost
12   - join threads
13   - remove HTTP headers from stored data
14   - calculate and print MD5 sum of downloaded data
```