CSC/ECE 573 – Internet Protocols
Project Report

Peer-to-Peer with Distributed Index (P2P-DI) System for Downloading RFCs

Problem Statement

To implement a simple peer-to-peer (P2P) system with a distributed index (DI) in which peers who wish to download an RFC that they do not have in their hard drive, download it from another active peer who does. All communication among peers or between a peer and the registration server takes place over TCP.
Programming Language Used – C

Objectives Achieved

- Created server processes that wait for connections,
- Created client processes that contact a well-known server and exchange data over the Internet,
- Defined a simple application protocol and made sure that peers and server follow precisely the specifications for their side of the protocol in order to accomplish particular tasks,
- Created and managed a distributed index among multiple peers, and
- Implemented a concurrent server that is capable of carrying out communication with multiple clients simultaneously.

Application Message Format

1. The general format of messages exchanged between RS and peers, and peer to peer is

Request Messages

Method<sp> Message <sp> Version <cr> <lf>
Header Field Name: <tab> Value <cr> <lf>
Header Field Name: <tab> Value <cr> <lf>
Header Field Name: <tab> Value <cr> <lf> <cr> <lf>

Response Messages

Status<sp> Phrase <sp> Version <cr> <lf>
Header Field Name: <tab> Value <cr> <lf>
Header Field Name: <tab> Value <cr> <lf>
Header Field Name: <tab> Value <cr> <lf> <cr> <lf>

- The specific format for request and response messages exchanged between RS and Peers-

a. Register Request Message

Registration <sp> Register <sp> P2P – DI/1.0 <cr> <lf>
RFC Server Listen Port: <tab> <RFC Server Listen Port Number Between65400-65500)<cr><lf>

Cookie: <tab> <Cookie Number> <cr> <lf>
Date&Time: <tab> <Register Date and Timestamp> <cr> <lf> <cr><lf>

b. Register Response Message

OK <sp> Register <sp> P2P – DI/2.0 <cr> <lf>
Hostname: <tab> <Hostname Value> <cr> <lf>
Cookie: <tab> <Cookie Number> <cr> <lf> <cr> <lf>

c. Peer Query Request Message

Query <sp> PQuery <sp> P2P – DI/1.0 <cr> <lf> <cr> <lf>

d. Peer Query Response Message

OK <sp> PQuery <sp> P2P – DI/2.0 <cr> <lf> <cr> <lf>

e. Keep Alive Request Message

Poll <sp> KeepAlive<sp> P2P – DI/1.0 <cr> <lf>
Cookie: <tab> <Cookie Number> <cr> <lf> <cr> <lf>

f. Keep Alive Response Message

OK <sp> KeepAlive<sp> P2P – DI/2.0 <cr> <lf><cr> <lf>

g. Leave Request Message

Exit <sp> Leave <sp> P2P – DI/1.0 <cr> <lf>
Cookie:<tab> <Cookie Number> <cr> <lf> <cr> <lf>

h. Leave Response Message

OK <sp> Leave <sp> P2P – DI/2.0 <cr> <lf><cr> <lf>

- The format of messages exchanged between peers
  - **RFC Query Request Message**

    Query <sp> RFCQuery <sp> P2P – DI/1.0 <cr> <lf> <cr> <lf>

  - **RFC Query Response Message**

    OK<sp> RFCQuery <sp> P2P – DI/1.0 <cr> <lf> <cr> <lf>

  - **Get RFC Request Message**

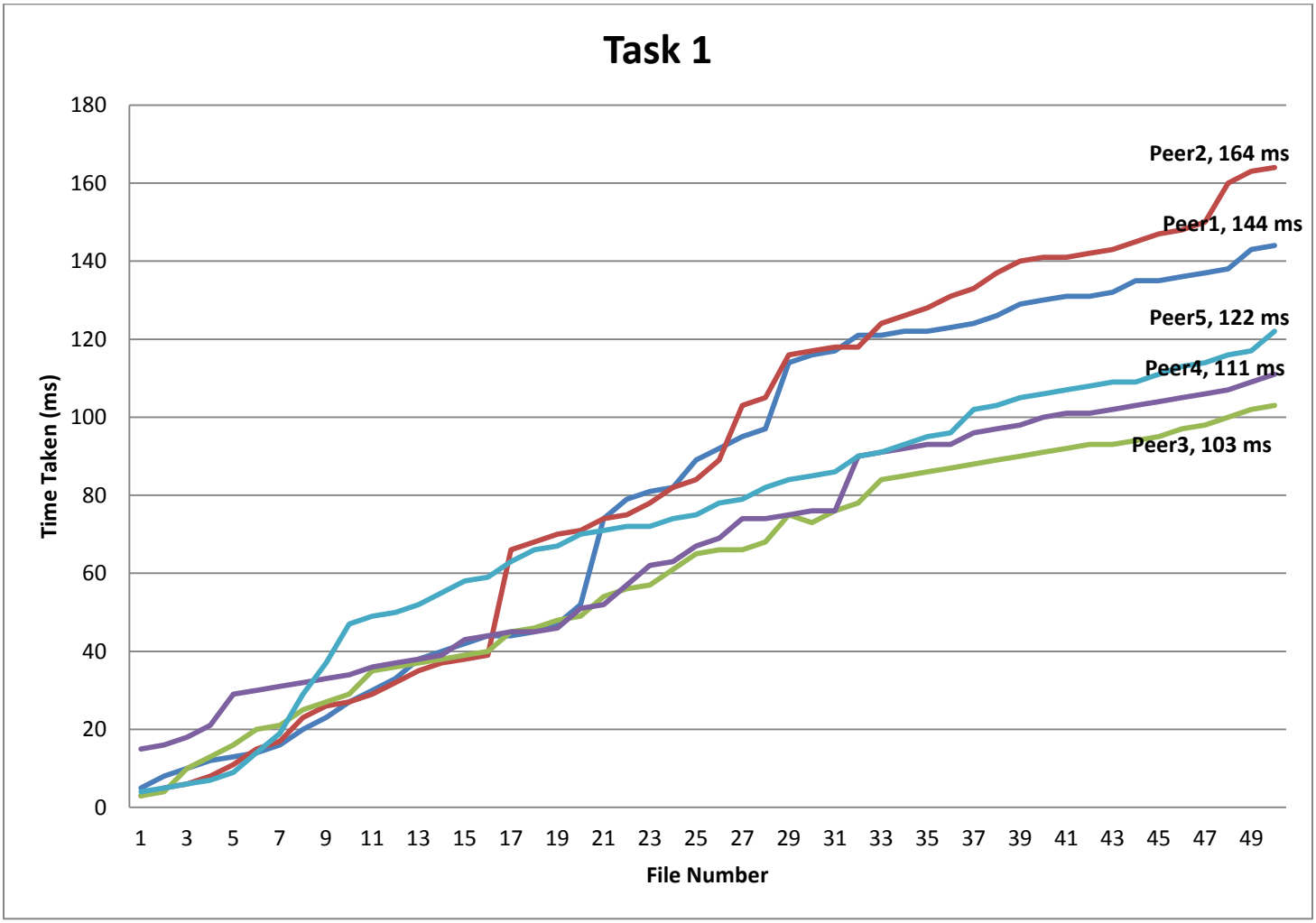
    Get <sp> GetRFC <sp> P2P – DI/1.0 <cr> <lf>
    RFC Number: <tab> <RFC Number Value> <cr> <lf> <cr> <lf>

  - **Get RFC Response Message**

    OK<sp> GetRFC <sp> P2P – DI/1.0 <cr> <lf> <cr> <lf>

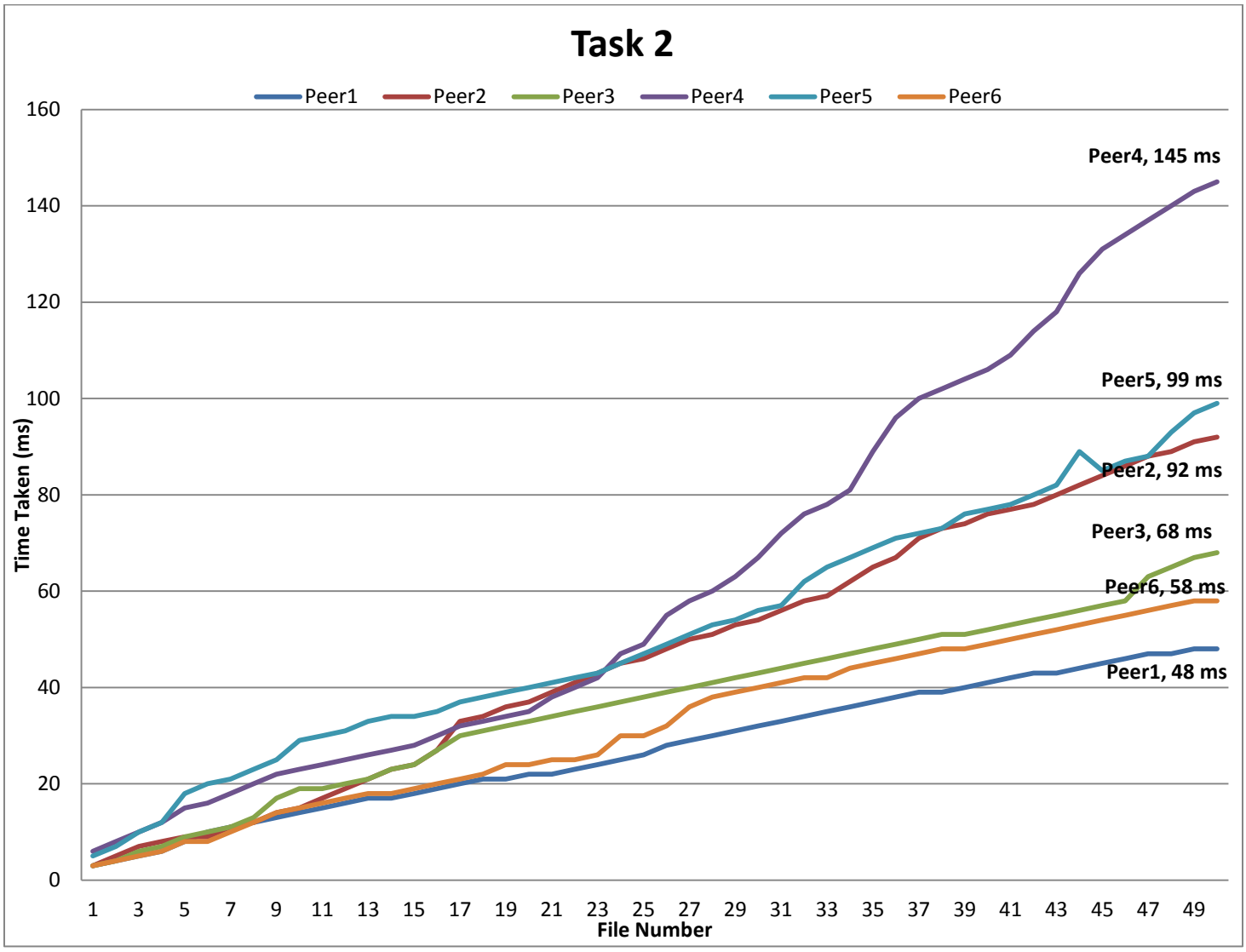
2. Task 1: Centralized File Distribution

- **The Download Time Curve for task 1**
  Here Peer 0 emulates the centralized server and maintains all the information and handles simultaneous requests from clients Peer1, Peer2, Peer3, Peer4, Peer5.



3. Task 2: P2P File Distribution

- **The Download Time Curve for task 2**
  Here initially, all the Peers have 10 RFCs each. Then each Peer after obtaining a Peer Index from the RS, queries each other for RFC Index. On receiving the complete RFC Index for all RFCs, the peers then download each RFCs from each other.



Differences observed in the above download time curves

- The download time for Task 1 varies between 103 to 164 milliseconds for all the peers. However a worst case and best case scenario is observed for Task 2. This is due to **decentralized coordination** in peer to peer communication; unlike centralized file distribution system which maintains coordination centrally.
- The minimum download time for a peer is 48 ms in P2P system is considerably less than the minimum download time for Client Server Distribution which is 103 ms .
- Though the **number of peers in Task 2 are more** than the clients in Task 1, still **the average download time** and both the **minimum and maximum download times** for Task 2 (P2P) are **much lesser** than the corresponding values in Task 1.

Conclusions regarding scalability of P2P versus Centralized File Distribution

- P2P architecture for file distribution is **self -scalable**, in our code we can add as many as 100 peers, and there will be not be significant difference in time for downloading RFCs. This is

because as we add a new peer into a system, though it generates its own load, it also adds service capacity to the system by distributing files to other peers and there minimum (or no) reliance on dedicated servers.

- On the other hand for every client we add in a client server architecture, the load on server increases significantly and it might get overwhelmed with the number of download requests which will increase the download time significantly for the clients.
- Thus from these tasks we can conclude that Peer 2 Peer File Distribution System is inherently more scalable than Centralized File Distribution.