Overview

We have implemented a peer-to-peer(P2P) concurrent system with a centralized index(CI), which contains both servers and clients, using socket programming. And we also defined an application protocol and guaranteed the peers and server follow precisely the specifications for their side of the protocol. This system can be used to share RFCs and some other files such as PDF during peer servers and peer clients.

Implementation

RS server

The RS server waits for connections from the peers on the port 65423 for this project. When a peer client tries to connect to the RS server, four types of request are accepted by the server :

Method
1. Register
2. Leave
3. PQuery
4. KeepAlive

And all requests must follow us **message format** which will be provided in the following instructions.

In the RS server thread, we maintains a linked list to store the Peers that have been registered and not logged out.
And each record contains some information are as follows:

Attribute	Type	description
hostName	String	IP address of the peer
cookie	Integer	Cookie assigned to peer by RS
isActive	Boolean	Currently active or not
TTL	Integer	live period
port	Integer	Port of peer
registered_count	Integer	# of times that the peer has been active
last_registered	String	most recent time the peer registered

MESSAGE FORMAT

request format

method	document	version
Host		
OS		
Data		

response format

Status Code	Phrase	Version
Host		
OS		
Data		

status code

status code	description
100	FILE_NOT_FOUND
200	OK
300	BAD_REQUEST
403	FORBIDDEN

Examples :

Request	Response
Register	
Register NULL P2P_DI_1.0 Host 10.153.44.84 OS Mac OS X cookie: port:4394	200 OK P2P-CI/1.0 Host: 10.153.15.5 OS: Mac OS X cookie:4

Request	Response
Leave	
Leave NULL P2P_DI_1.0 Host 10.153.44.84 OS Mac OS X cookie:4 port:4394	200 OK P2P-CI/1.0 Host: 10.153.15.5 OS: Mac OS X you have been unregistered from the RS server!

Request	Response
KeepAlive	
KeepAlive NULL P2P_DI_1.0 Host 10.153.44.84 OS Mac OS X cookie:4 port:4394	200 OK P2P-CI/1.0 Host: 10.153.15.5 OS: Mac OS X you have been renewed! TTL is 7200 seconds

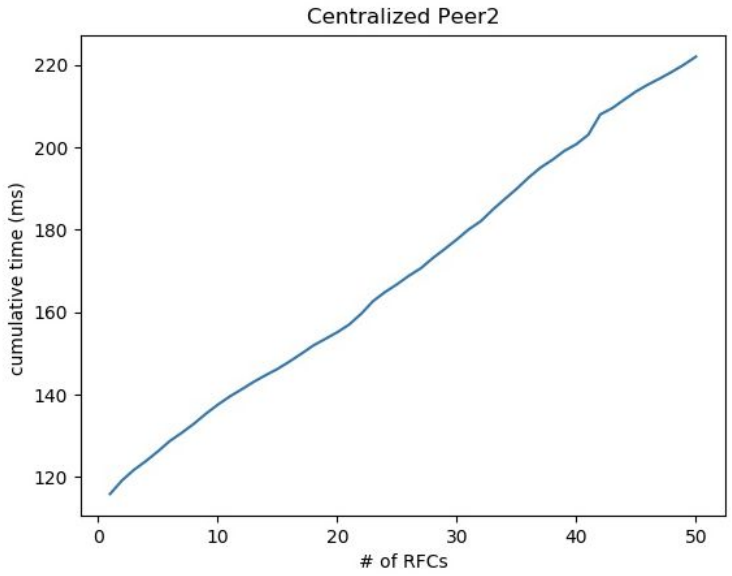
Request	Response
PQuery	
PQuery NULL P2P_DI_1.0 Host 10.153.44.84 OS Mac OS X cookie:6 cookie:4396	200 OK P2P-CI/1.0 Host: 10.153.15.5 OS: Mac OS X 10.153.44.84 4391 10.153.44.84 4392 10.153.44.84 4393 10.153.44.84 4394 10.153.44.84 4395 10.153.44.84 4396

Request	Response
RFCQuery	
GET INDEX P2P_DI_1.0 Host 10.153.44.84 OS Mac OS X	200 OK P2P-CI/1.0 Host: 10.153.15.5 OS: Mac OS X

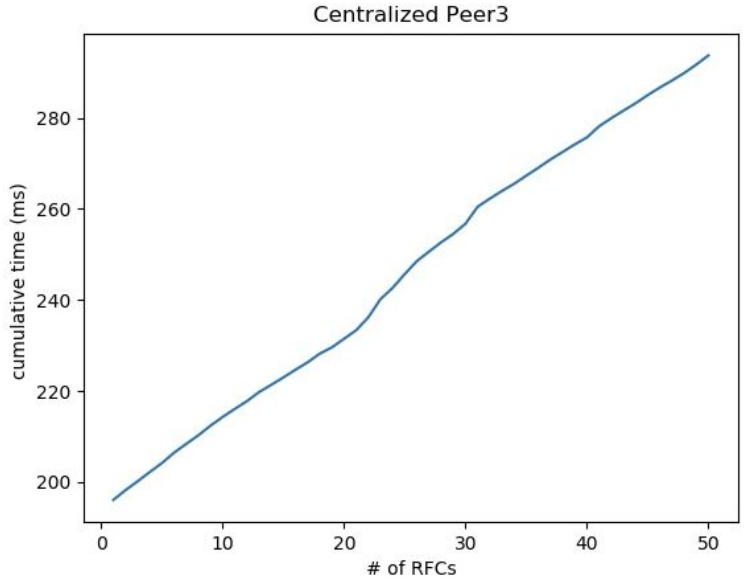
Request	Response
GetRFC	
GET FILE P2P_DI_1.0 Host 10.153.44.84 OS Mac OS X filename filetype	200 OK P2P-CI/1.0 Host: 10.153.15.5 OS: Mac OS X

Task 1 Centralized File Distribution

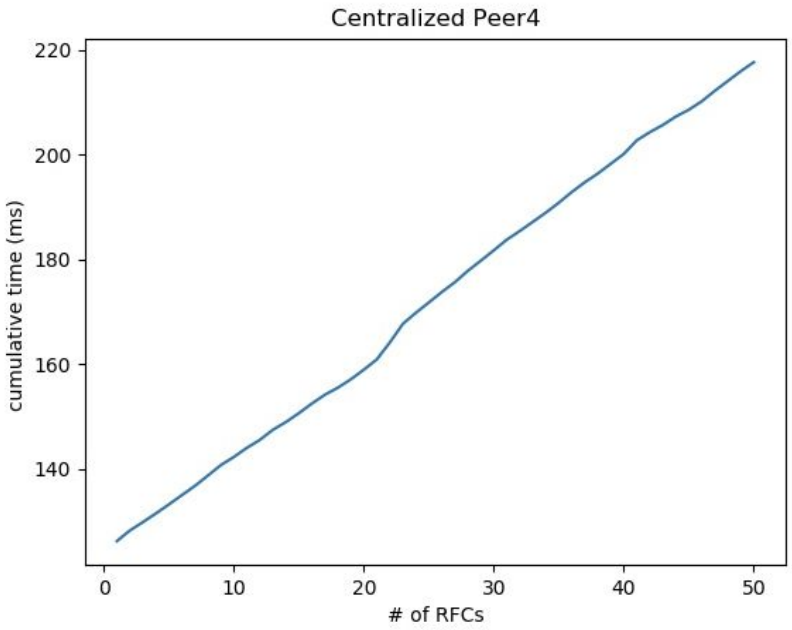
Peer2:



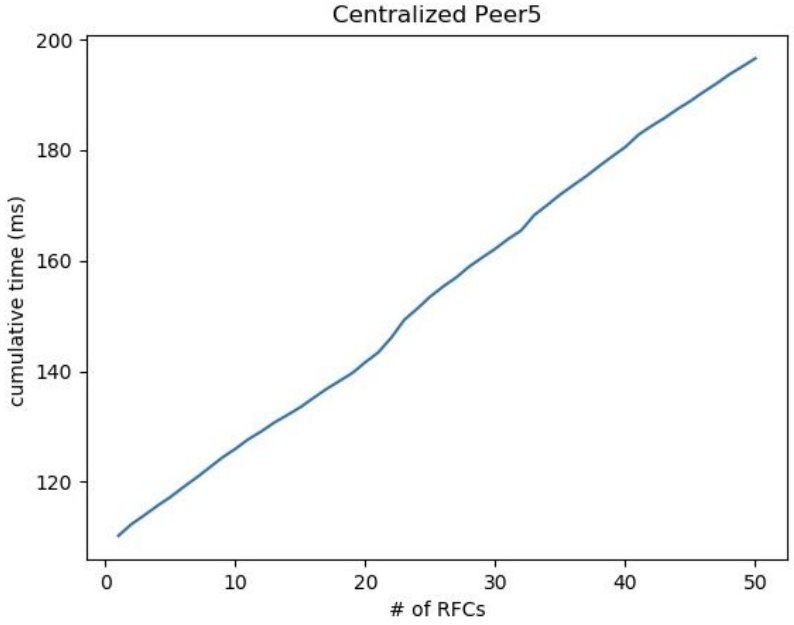
Peer3



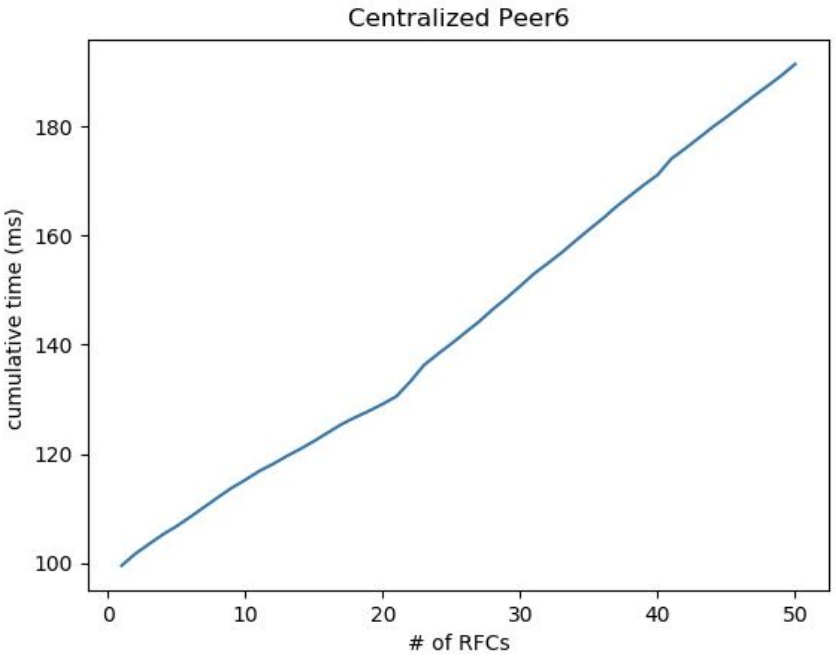
Peer4:


Centralized Peer4
cumulative time (ms)
220
200
180
160
140
0
10
20
30
40
50
# of RFCs

Peer5:


Centralized Peer5
cumulative time (ms)
200
180
160
140
120
0
10
20
30
40
50
# of RFCs

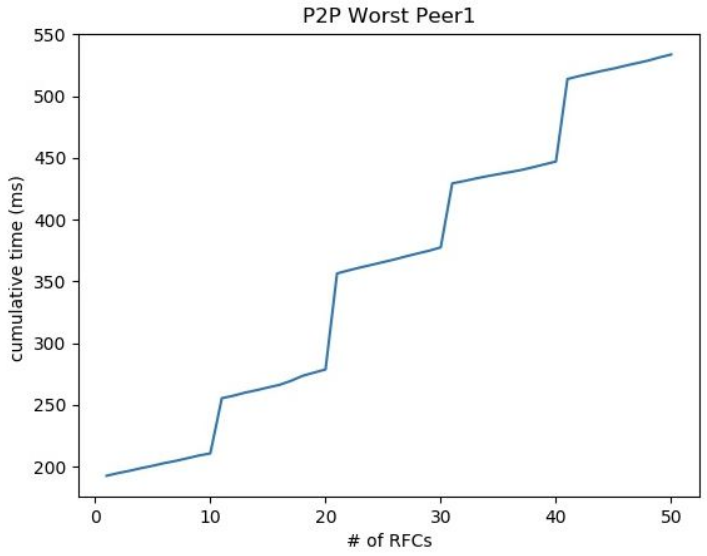
Peer6:


Centralized Peer6
cumulative time (ms)
100
120
140
160
180
0
10
20
30
40
50
# of RFCs

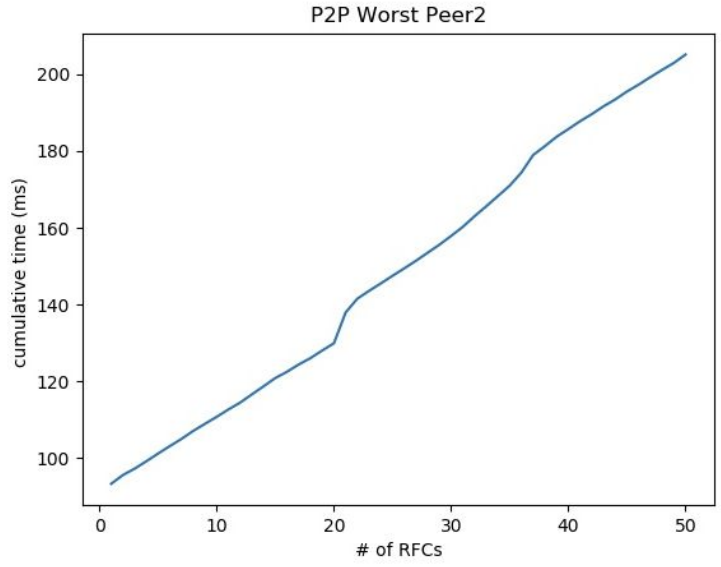
Task 2 P2P File Distribution

Worst case: peer 1 get each 10 files from peer 2 to peer 6

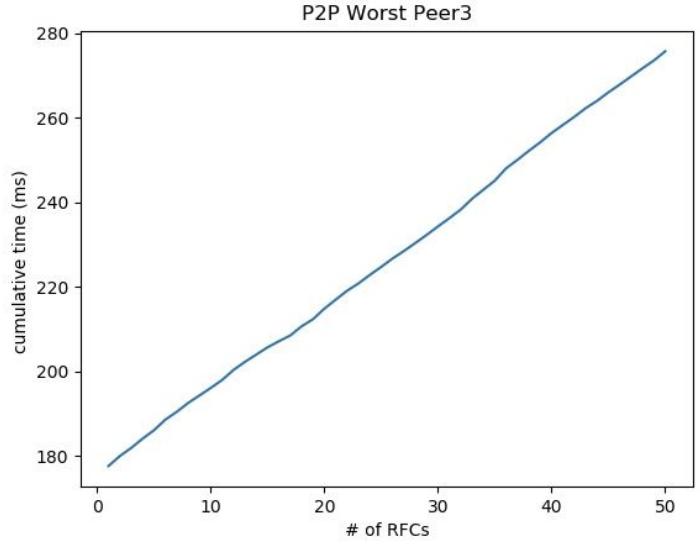
Peer1:



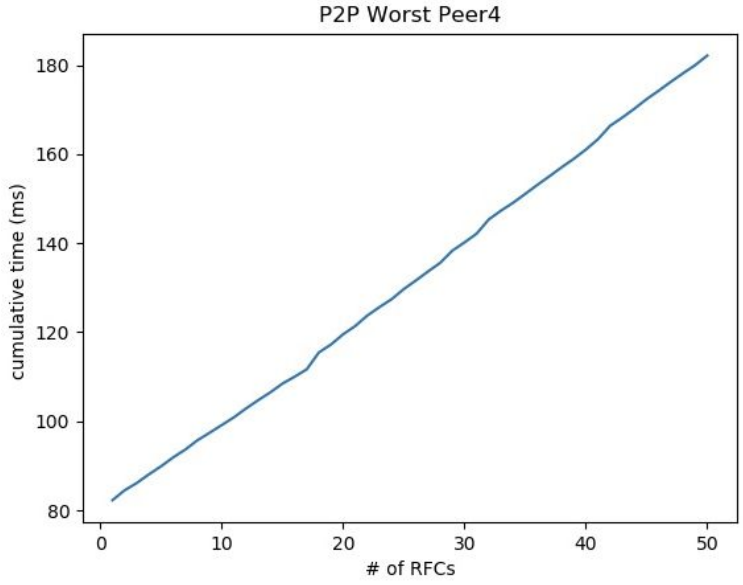
Peer2:



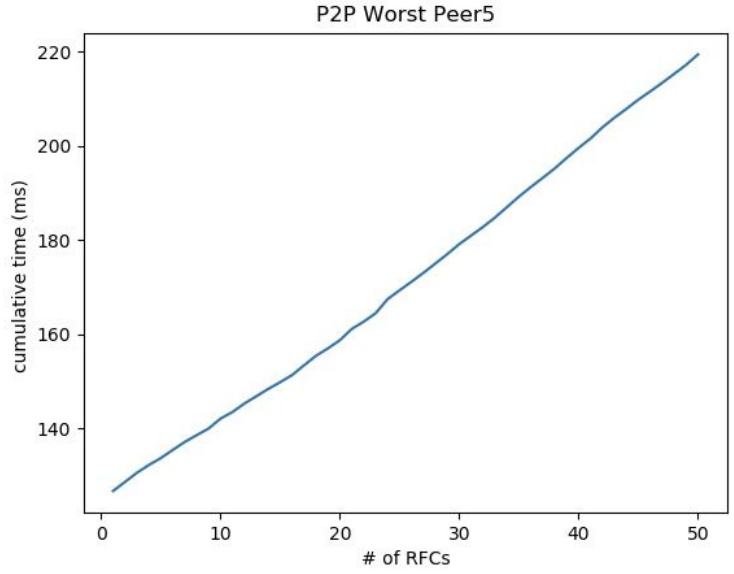
Peer3:


P2P Worst Peer3
cumulative time (ms)
# of RFCs
280
260
240
220
200
180
0
10
20
30
40
50

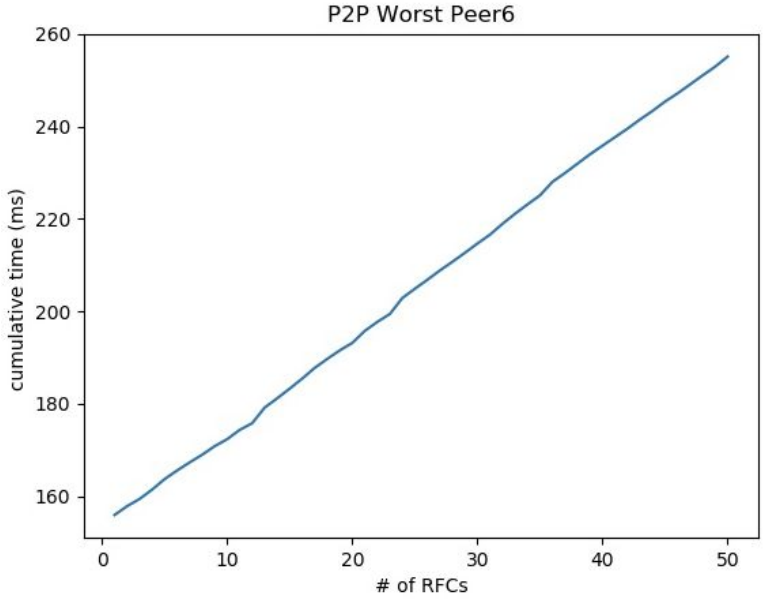
Peer4:


P2P Worst Peer4
cumulative time (ms)
# of RFCs
180
160
140
120
100
80
0
10
20
30
40
50

Peer5:



Peer 6

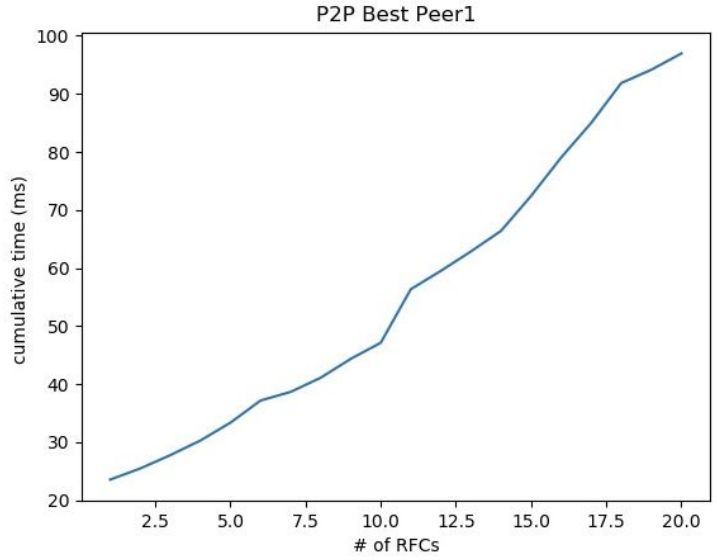


For the worst case, Peer 1 first download all 10 files from each peer 2 - 6, then peer 2-6 download all files they need from peer1, which will even slower than the centralized system.

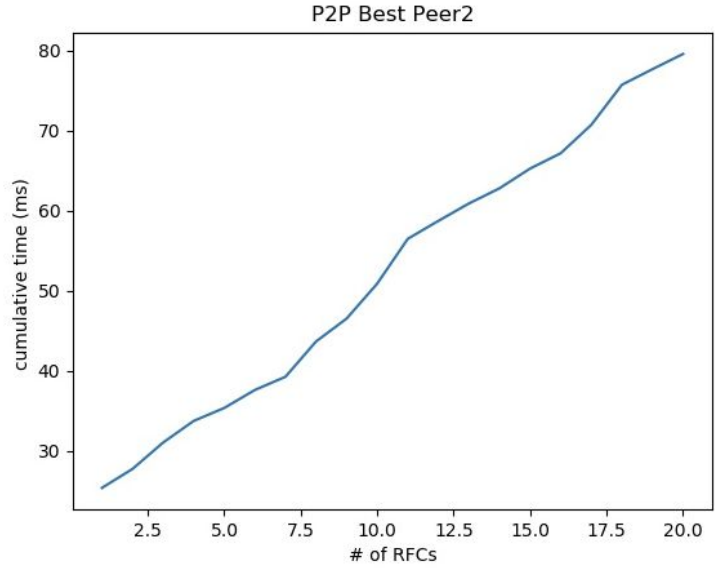
Task 2: best

We only use 3 peers to test the best case, where each peer starts to download the RFCs at exactly the same time. In this situation, peers can concurrently download and upload files and make the best use of P2P system. In order to make peers start at the same time, we need to manually press the start button. So we only test this situation with 3 peers. Even 3 peers are used, we still can see the time for each file is smaller than the centralized situation.

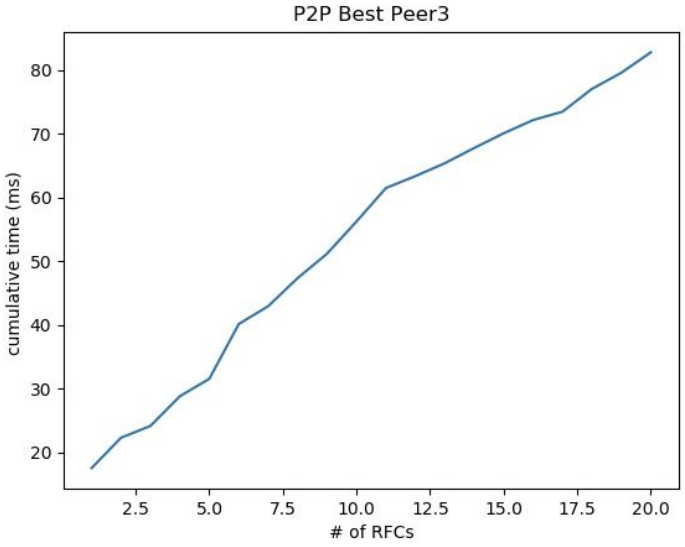
peer1



peer2



peer3



you can see from the figure that the P2P distributed system is much better than the centralized system.