COMP3331/9331 Computer Networks and Applications

Assignment Circular DHT

Z5147046 Meiyan Pan

1. Language

Python 3.6 (also works well on python 2.7)

2. Data Structure

1.2.1 class Peer

element	description	element	description
self.peerid	Integer in [0,255] name of peer	self.is_alive	1 peer is alive, otherwise 0
self.suc1	Integer in [0,255] first successor	self.input	The input captured in the xterm
self.suc2	Integer in [0,255] second successor	self.file	File number in the request
self.psc	A list contains the predecessors	get_psc()	Returns the predecessors
self.psc1	First predecessor	get_suc1()	Returns the value of suc1
self.psc2	Second predecessor	get_suc2()	Returns the value of suc2
get_is_alive()	Returns the value of self.is_alive	hush()	The hush function determines where the file is stored
get_psc1()	Returns the value of psc1	get_psc2()	Returns the value of psc2
get_file()	Determine whether a peer has the file: 1 has the file ,otherwise 0		

1.2.2 messages

messages	structure	example
Ping request	str(50000+peerid)+'request'+str(sequence_no)	50001request1
Ping response	str(50000+peerid)+'respons'+str(sequence)	50002respons1
File request	str(1000+peerid)+input	1008request 2012
File response	'has'+str(peerid+1000)+filename	has10012012

Quit request	1+'quit'+str(peerid+1000)+str(suc1+1000)+str(suc2+1000) 2+……	1quit101010121015
Kill response	str(suc1+1000)	1012
Ask for successor	'ask'+str(1000+suc1)	ask1008

1.2.3 important parameters

Set the timeout for UDP messages to 2 seconds.
Set the the frequency of the ping messages to be 2 seconds.
Set the number of consecutive ping request messages that a peer fails to respond to before that node is declared to be no longer alive to be 4.

3. Program Structure

In the main fuction,4 threads are always alive when the program runs:

Pingclient_thread	UDP client
Pingserver_thread	UDP server
Input_thread	TCP client
TCP_thread	TCP server

3.1. UDP
The ping client thread includes two subthreads to send ping request to the two successors while the peer is alive. The ping server thread is always on as long as it is alive.

1.3.2 TCP Client

Three kinds of client fuction:
1 Client of the peer who got the input:
While the peer is alive,when the Input_thread catches a file request input, open a TCP client thread for this peer. Build the TCP connection to its first successor and send the request message.
2. Client of the peer who received request message sent from other peer but does not store the file:
Build the TCP connection to its first successor and pass the request message.
3. Client of the peer who stores the file:
Build the TCP connection to the peer who request for the file originally and send the response message to it.
4. Quit client:
While the peer is alive,when the Input_thread catches a quit input, open a TCP client thread for this peer. Build the TCP connections to its two successors and send the quit messages. Then, close the socket.
5. Kill client:
In the ping client function, if 4 consecutive responses are not received, make its suc2 be the new suc1. Then open a TCP client thread for this peer. Connects to its successor and send Ask for successor message to get its new second successor.Wait for the response and after receiving it , cut the connection.

1.3.3 TCP server

TCP server has only one always on thread and five kinds of server function:
1 server who request for the file: recieve File response message from the peer who has the file and then cut the connection.
2 server who received the file request message and does not store the file:
Cut the current connection and open a thread for TCP client of the same peer.
3 server who received the file request message and store the file:
Cut the current connection and open a thread for TCP client of the same peer.
4 quit server:
After receiving the message,simply print the output and cut the connection.
5 kill server:

Send the response contains the new suc2.

1.4 Demo Link

https://youtu.be/PE2QMiRM4bs