

EE565 – Project 4 – Design Document

Team members: Peillen Li, Lee Li, Pinyi Wang


Peillen mainly worked on gossip protocol for /peer/search, test and the integration to /peer/view, Pinyi helped us for debugging, Lee mainly worked on writing the build.xml to make the project more professional.

A. What are the drawback/limitations of the given protocol?

The gossip protocol can become less effective about scalability (as the number of peers in the network grows). This is because each peer needs to communicate with a randomly selected peer, which can become a bottleneck as the number of peers increases. Moreover, there is no guarantee that a message will reach all peers in the network. Some peers may be missed, which can result in incomplete search results or missed content. In addition, the gossip protocol can generate a large amount of network traffic as peers constantly exchange messages. This can result in high bandwidth usage, which may not be feasible for some networks or devices.

B. Libraries used (optional; name, version, homepage-URL; if available)

External libraries: in /lib

 json-simple-1.1.1 version:1.1.1

```
import org.json.simple.*;
import org.json.simple.parser.*;
```

Java standard libraries:

```
import java.net.*;
import java.io.*;
import java.lang.Thread;
import java.util.*;
import java.text.SimpleDateFormat;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.UUID;
import java.util.Properties;
import java.nio.file.Files;
import java.nio.file.Paths;
```

C. Extra instructions on how to execute the code.

Our build.xml will create /META-INF and /org for the external library – org.json.simple.

4 steps to execute our code:

1. After turning on the peer nodes, it will take them about 10s to do the LSA routing and get the global map in each node, after the frontend port will be turned on, and then you can:

2. use /peer/search/filename to search content, and there are hashmap to record peer lists for each content in each node, so after a gossip search, each node will have the content peer list. And after a round of gossip search, frontend will show

```
{“content”:“video.ogg”, “peers”:[“f94fc272-5611-4a61-8b27-de7fe233797f”, “24f22a83-16f4-4bd5-af63-9b5c6e979dbb”, “e94fc272-5611-4a61-8b27-de7fe233797f”]},
```

3. use /peer/rank to do the Dijkstra and get the shortest metric to each peer node.

4. use /peer/view/filename to view the content (we do project3 and project 4 on top of project2 checkpoint, so we use .txt file to test it), it can get content from the nearest peer who has the content file.