Distributed Peer-to-Peer System

Title: Distributed Peer-to-Peer System File Searching and Downloading.

A peer-to-peer (P2P) system, for the purpose of the project, consists of a set of autonomous computers that act as a distributed repository of files, and allow participating computers to search for and download files from each other.

The model to consider is the following:

Each computer, Ci, maintains a file, Fi that contains a list of files (and associated keywords) that are being made visible to the outside world. Only the files in Fi can be downloaded from Ci by other computers.

When a computer, Ci, wishes to join the P2P system it is assumed that it knows of at least one other computer, Cj , that is already part of the P2P system. If Ci is the node that initiates the P2P system then Ci knows that, too.

In order to join the P2P system, computer Ci sends a join request to Cj . Both Ci and Cj add each other to their list of neighbors and establish a TCP connection between themselves.

File Sharing Procedure:

Simple Search Approach: If computer Ck is looking for a file, it can issue the search request using either a file name or a keyword, and a hop-count which is initially set to 1.

1. The search request is flooded into the overlay network (along the links represented by the list of neighbors) to other computers that are no more than hop-count away from the computer issuing the search request. Having initiated the search request, Ck starts a timer which is set to expire after thop count seconds.

2. You must ensure that a computer forwards a search request issued by another computer at most once, i.e., your implementation should be able to detect duplicate requests.

3. Let there be a computer Cx that has a file, listed in the corresponding Fx, whose name matches the specified file name, or whose keyword matches the specified keyword. Then, Cx sends a response to the neighbor from which it received the first copy of the search request. The response contains Cx, and the keyword and file name of the matched entry. If a computer that initiated the search request (Ck) receives a reply, it consumes the reply.

4. When a computer that did not initiate the search request (Cl) receives a reply, it forwards the reply to the neighbor from which it received the first copy of the corresponding search request.

5. As part of consuming a reply, the search initiator, Ck, collects all the replies received until the expiry of the timer, and then displays them as a list of tuples of the form (kwd, file name, comp name). Replies received after the expiry of the timer are ignored.

6. The user, on Ck, that made the request can specify which response tuple he/she is interested in.

7. If the selected tuple corresponds to a file located at computer Cm then Ck establishes a TCP session with Cm and copies the file from Cm to its own directory, and updates Fk accordingly. Once the file has been copied from Cm, the TCP session established with Cm for the purpose of obtaining the file is terminated.

8. If the search terminated without success (no reply received before the timer expired), then Cm should double the hop-count and re-initiate the process. This should be repeated until there is success, or the hop-count exceeds sixteen (whichever happens earlier). Note that the timer value is a monotonically increasing function of hop-count.

Termination:

Having grown to fifteen nodes, you should initiate departure of nodes, one at a time. If a departing node has only one neighbor then all that node has to do before departing is to terminate its TCP session with its neighbor and all ongoing TCP file transfer sessions. If a departing node has two or more neighbors then before departing it should arbitrarily select one of its neighbors, n1, and make it a neighbor of all other neighbors (provided the two nodes are not already neighbors of each other). Then, it should terminate its TCP sessions with all its neighbors as well the ongoing TCP file transfer sessions.

SUMMARY:

>> Setting Up the Peer-to-Peer Network

Each Peer's Role:

- Every computer (peer) has a list of files it shares, including file names and keywords.

Joining the Network:

- A new computer (Ci) must know at least one existing computer (Cj) in the network.
- Ci sends a **join request** to Cj.
- Both Ci and Cj add each other as neighbors and establish a **TCP connection**.

Starting the System:

- If a peer is the first node in the system, it starts without needing another peer.

>> How File Searching Works

Initiating a Search:

- A peer (Ck) looking for a file sends a **search request** with a file name or keyword.
- The request starts with a **hop count of 1** (limits how far the request travels).

Flooding the Request:

- The request is forwarded to all **neighboring peers** within the hop count limit.
- Each peer **checks its file list** for a match.

Avoiding Duplicate Requests:

- Each peer processes and forwards a **search request only once** to avoid duplicates.

Replying to a Search Request:

- If a peer (Cx) finds a matching file, it **replies along the same path** the request came from.
- The reply includes: **file name, keyword, and Cx's identity**.

Collecting Responses:

- The **initiating peer (Ck) starts a timer** after sending the request.
- It collects **all replies** before the timer expires.
- Replies arriving **after the timer expires are ignored**.

>> Downloading the File

User Selection:

- Ck displays a **list of matching files** (keyword, file name, computer name).
- The user **chooses** which file to download.

Establishing a Download Connection:

- Ck creates a **new TCP session** with the selected peer.
- The file is **copied to Ck's directory**.
- After downloading, the **TCP session is closed**.

Retry Mechanism:

- If **no replies** are received before the timer expires:
    - The **hop count doubles** and the request is resent.
    - This repeats **until success or hop count exceeds 16**.

>> Handling Node Departures (Leaving the Network)

If the Departing Peer Has One Neighbor:

- It closes its TCP connection and any ongoing file transfers.

If the Departing Peer Has Multiple Neighbors:

- It selects one neighbor (n1).
- It ensures n1 connects with all its other neighbors (if not already).
- It then closes all TCP sessions and exits the network.