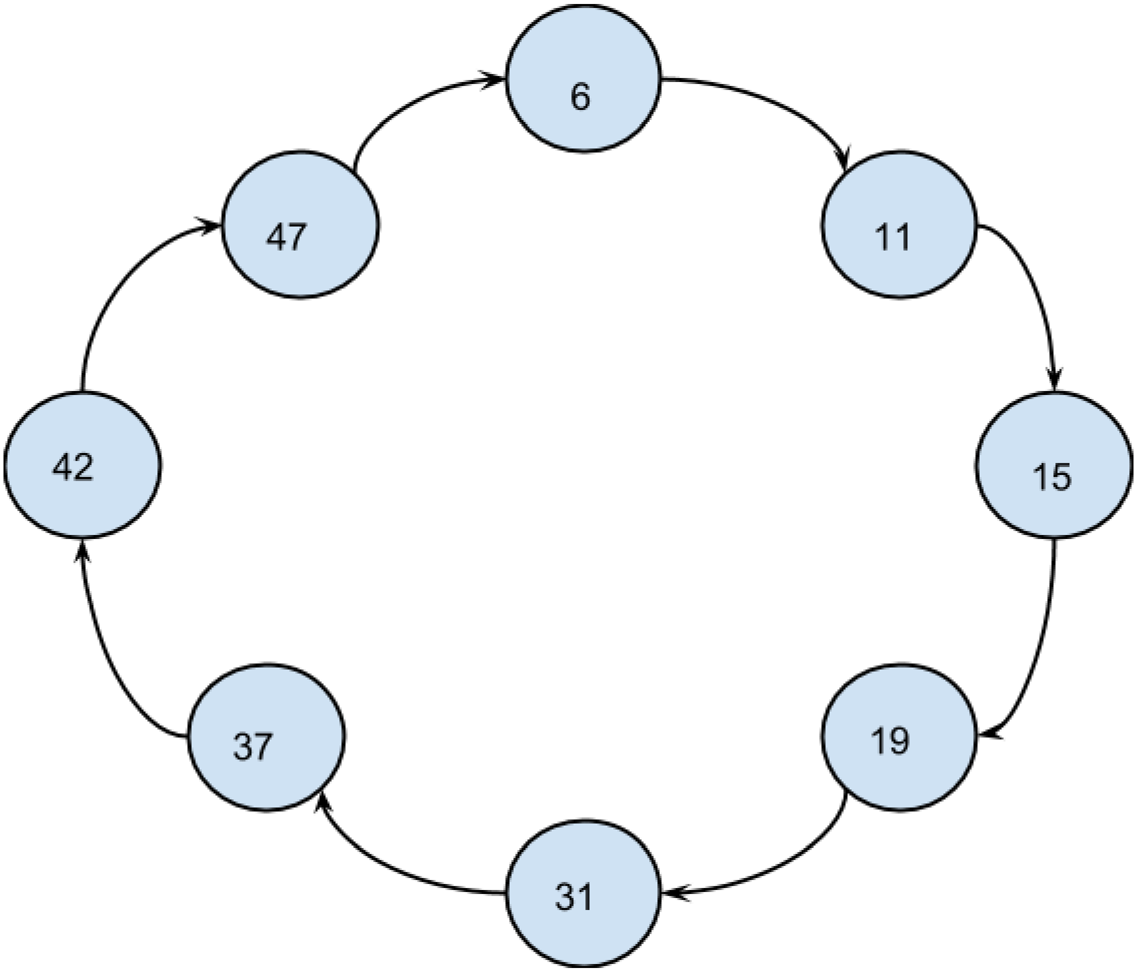
Computer Networks Lab

Assignment 6

Problem Statement

The purpose of this assignment is to implement Chord peer-to-peer protocol. Let the number of nodes (machines) be ‘n’. Each node is connected to every other node. However, we assume an overlying ring topology amongst the nodes.

Each node is assigned an identifier (ID) using a hash function (consistent hashing). The nodes are arranged in a circle in increasing order of their IDs as shown in the following diagram for n = 8:



The hash function ‘h’ is defined as follows:

‘h’: *std::string* ---> *unsigned long long integer*

For generating IDs for nodes we formulate the input string for the hash function by concatenating the IP address of the machine followed by the port number with a ‘:’ in between.

Each node ‘*i*’ is capable of answering two kinds of queries:

1) *i.successor()*
   # For eg. *Node(37).successor()* would result in *Node(42)*
2) *i.predecessor()*
   # For eg. *Node(19).predecessor()* would result in *Node(15)*

Now lets shift our focus to the distribution and retrieval of files from these nodes. As with each node, each file is also assigned a unique identifier (*file_id*) using the same hash function ‘h’. For generating identifiers for files we formulate the input string for the hash function by specifying the filename of the corresponding file. For making this assignment simple, you may assume that each file has a different name. However, in a practical situation if that cannot be assumed, one can easily take the MD5 hash of each file and then use that hash output as an input for ‘h’.

Each file identified by *‘file_id_x’* would be stored on the node with *ID_m* defined by the following function:

ID_m = map_file_to_node (file_id_x) = min{ID: ID >= file_id_x }
where ID is a machine’s valid identifier.

Similarly, when a search query comes for a file (with *file_id_a*) at node (with *ID_b*), we compute *ID_a* = *map_file_to_node* (*file_id_a*). Then we use a combination of *successor()* and / or *predecessor()* functions until we get a pointer to *Node(ID_a)* starting from *Node(ID_b)*. Then we can setup a TCP connection to download the required file.

For this assignment, you are given a set of nodes (machines) and a set of files. Your task is to achieve the following:
1) Arrange the nodes in the ring topology as described above.

2) Implement a Node class which has the following functions in addition to others which are described below:
Node successor();*
Node predecessor();*

3) Distribute the files amongst these nodes using the mechanism described above.

4) Implement a search function which returns a pointer to the node which contains the file described by the *file_id* provided as input.

5) Implement a download function which is given a filename as well a pointer to the node which ‘should’ contain that file.

Each node runs a single process (defined by the Node class). The client process interacts with the user as well as with processes running on different nodes.

All communication between nodes, except for actual file transfer, is done using User Datagram Protocol. However, use of Transmission Control Protocol is recommended for all file transfers.

You may use a combination of *std::hash* function and other elementary operations such as modulo (%) to define ‘h’.

Clean and modular design will earn extra credit.

Reference: http://en.wikipedia.org/wiki/Chord_%28peer-to-peer%29