15-440: Distributed Systems

Problem Solving Assignment 3

School of Computer Science

Carnegie Mellon University, Qatar

Fall 2016

Assigned Date: October 3rd, 2016

Due Date: October 17th, 2016

Part 1: Networking & Scalability (20 Points)

The Internet is far too large for any router to hold routing information for all destinations. How does the Internet routing schemes deal with this issue? Specifically, your answer should describe how Internet routing handles the following cases:

- Information storage at routers: **(7 Points)**
  - How do the routing protocols store the information to route packets to all the computers connected to the Internet?
  - What information is stored?
  - Is this information sufficient to route any packet to any computer on the Internet?
- Packet forwarding: **(8 Points)**
  - How does the router use the stored routing information for forwarding each packet?
  - What are the challenges while forwarding?
  - What is the fault-tolerance mechanism employed by routing?
- Scalability: Explain the scalability challenge for routing over the Internet. **(5 Points)**

Part 2: Chord (30 Points)

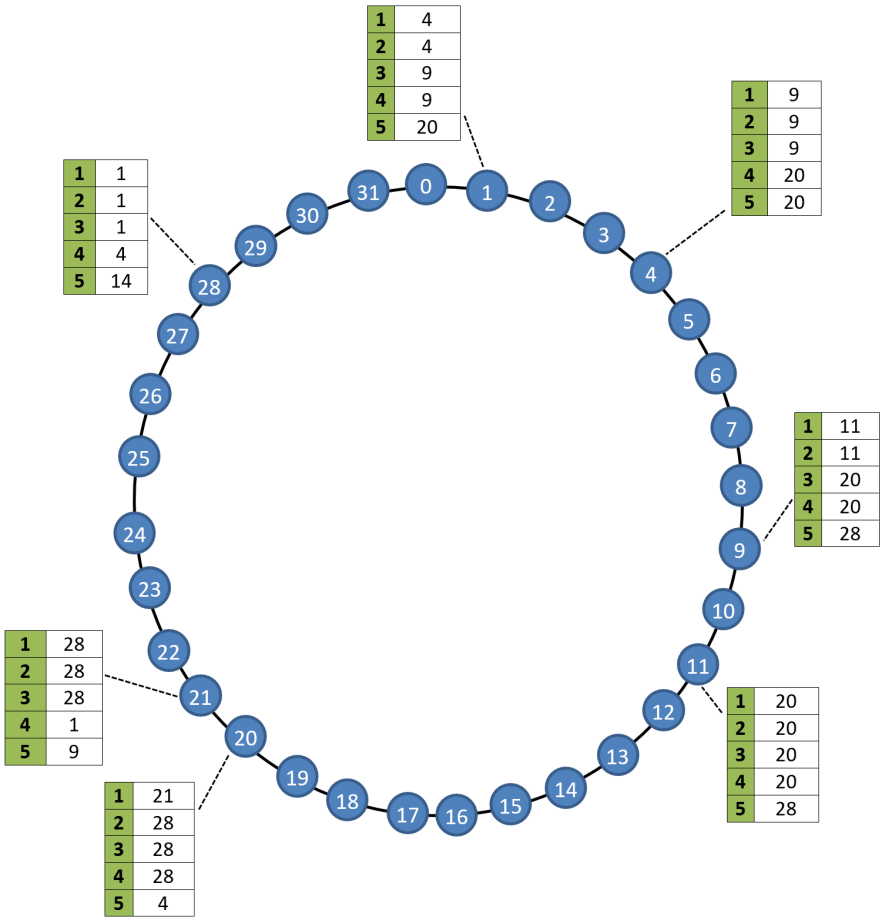

Figure 2.1

Q2.1. Consider the Chord system shown in *Figure 2.1*. Suppose that **Node 20** leaves the system voluntarily. Describe all the steps that need to be taken to bring the Chord system up-to-date. **(15 Points)**

Q2.2. Suppose that, instead of **Node 20** *leaving* the Chord system voluntarily, it *fails* abruptly. This means that, with the current finger tables, the closest successor for **Node 11** would be **Node 28** (since **20** is down). Therefore, **Node 20** disregards all the nodes between **Nodes 21** & **28, excluding Node 28** (regardless if these nodes are active or not). How can Chord resolve this problem? *(Hint: you can change or add additional data at each node).* **(10 Points)**

Q2.3. Assume in our Chord system that k bits of an m-bit identifier space are reserved for assigning to superpeers. If identifiers are randomly assigned, how many superpeers can one expect to have in an N-node system? **(5 Points)**

Part 3: Group Communication & Synchronization (50 Points)

Suppose that we can classify the various kinds of multicasts possible into four classes of source–destination relationships, as illustrated in *Figure 3.1*:

- SSSG: Single source and single destination group.
- MSSG: Multiple sources and single destination group.
- SSMG: Single source and multiple, possibly overlapping, groups.
- MSMG: Multiple sources and multiple, possibly overlapping, groups.

Keep in mind the following assumptions:

- A source that is sending messages to any group(s), must also be part of that group(s).
- The source that is sending a message to a group does not receive the message itself.
- There can be some sort of information sharing amongst group members.

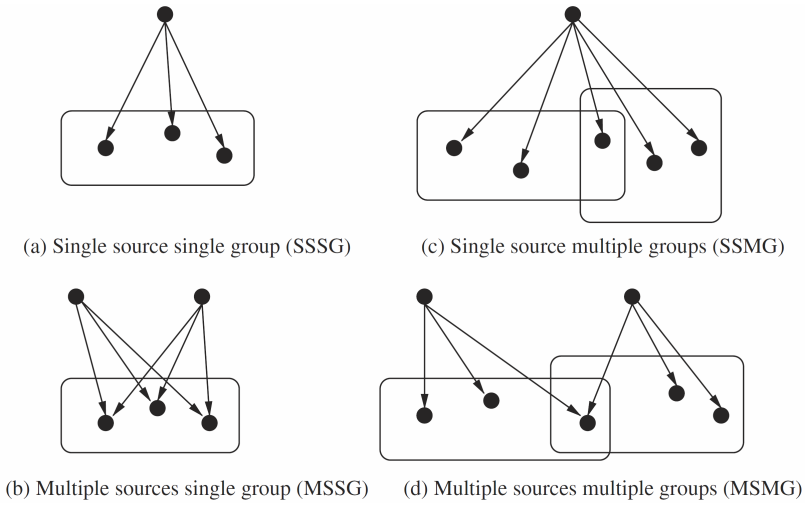

Figure 3.1

Q3.1. Consider two source nodes ***N1*** and ***N2*** in MSSG (see part (b) in *Figure 3.1*). Assume that these source nodes do not have synchronized clocks, and that the underlying network is a LAN. If these sources were to deliver multiple messages to the designated group, how can we ensure that the messages are received *in order*? That is, if ***N1*** sends message *M1* before ***N2*** sends message *M2*, the group members on the receiving end must receive *M1 before M2. (Hint: try to logically convert MSSG to SSSG.)* **(10 Points)**

Q3.2. Consider the following scenario: source nodes ***N1*** and ***N2*** send messages *M1* and *M2* respectively, to Group ***G1***, where ***G1*** has members {N1, N2, N3, N4}. How many messages are sent in total? Is this the optimal number of messages that can be sent? If not, then what would be the optimal number and how can that be achieved? **(7 Points)**

Q3.3. Suppose that we would also like to have messages received in order in MSMG (see part (d) in *Figure 3.1*). Without creating a completely new algorithm/solution, can you rely partially or fully on the one you devised for MSSG in **Q3.1.1**? If not, suggest a completely new solution. Explain your approach in detail. **(7 Points)**

Q3.4. Consider the following scenario: source node ***N1*** sends messages *M1* and *M2* to group ***G1***, which has members {N1, N2, N4, N5, N6, N8}, and source node ***N2*** sends messages *M3* and *M4* to group ***G2***, which has members {N1, N2, N3, N4, N6, N7, N8, N10}. How many messages are sent in total? Is this the optimal number of messages that can be sent? If not, then what would be the optimal number and how can that be achieved? **(7 Points)**

Q3.5. Consider the behavior of two nodes in any of the classes depicted in *Figure 3.1*. Both nodes have clocks that are supposed to tick 1000 times per millisecond. One of them actually does, but the other ticks only 990 times per millisecond. If UTC updates come in once a minute, what would be the maximum clock skew? **(7 Points)**

[Part 3 continued on the next page]

Q3.6. Assume a node in any of the classes shown in *Figure 3.1* is attempting to synchronize with a time server. As a result, it records the round-trip times and timestamps returned by the server in a table as shown below (i.e., *Table 3.1*). **(12 Points)**

Round-Trip (ms)	**Time** (hr:min:sec)
22	10:54:23.674
25	10:54:25.450
20	10:54:28.342

Table 3.1

Q3.6.1. Which of the times in *Table 3.1* should the node use to adjust its clock? To what time should the node set its clock? Estimate the accuracy of such an adjustment with respect to the server's clock.

Q3.6.2. If it is known that the time between sending and receiving a message in any of the classes portrayed in *Figure 3.1* is at least 8 milliseconds, do your answers (i.e., the time setting and the accuracy) change? Explain.