15-744 Computer Networks — Spring 2015
Homework 3
Due by 4/8/2015

(to be submitted in the beginning of class in hard copy)

Name:

A DHTs

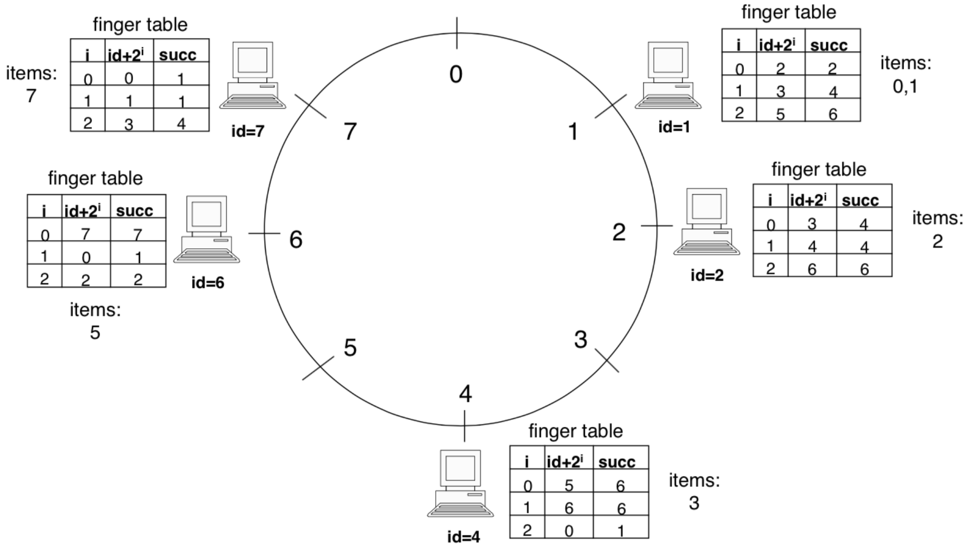
1. BitTorrent is one of the most popular peer-to-peer file sharing system today. In the original design of BitTorrent system, there is centralized component called "tracker", which informs each peer the presence of other peers in the same torrent. More specifically, to join a torrent a peer needs to first connect to the tracker listed in the .torrent file. Then the tracker replies this new peer with a list of known peers currently downloading the same torrent so that this peer can connect to them. Each tracker is usually serving a large number of torrents.

   (a) Briefly state the advantages and disadvantages of using tracker to manage peers.

   (b) Briefly, how could BitTorrent system use a DHT such as Chord to replace the tracker and maintain a directory of many torrents?

   (c) Your DHT-based tracker is more and more popular and growing much bigger. Meanwhile you found that it takes a long time to perform lookups. Explain why are lookups taking longer.

2. Brian, in fear that the RIAA will shut down his centralized P2P server (like Napster), sets up a Chord DHT for lookups and routing in his peer to peer network. Unfortunately (or fortunately, for you), Brian's P2P network is not very popular and only consists of five peers at the moment with the tables and items illustrated below. For example, node 4 has item 3.



finger table (id=7)

i	id+2^i	succ
0	0	1
1	1	1
2	3	4

items:
7

finger table (id=1)

i	id+2^i	succ
0	2	2
1	3	4
2	5	6

items:
0,1

finger table (id=6)

i	id+2^i	succ
0	7	7
1	0	1
2	2	2

items:
5

finger table (id=2)

i	id+2^i	succ
0	3	4
1	4	4
2	6	6

items:
2

finger table (id=4)

i	id+2^i	succ
0	5	6
1	6	6
2	0	1

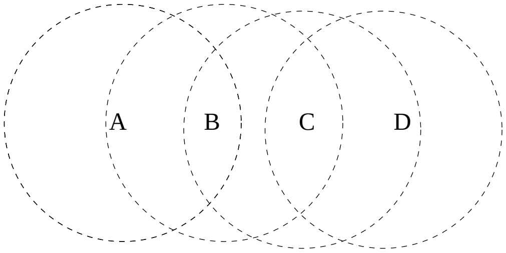
items:
3

(a) List the nodes that will receive a query from node 2 for item 0.

(b) Suppose node 4 crashes. node 7 queries for item 5. List the nodes that will receive this query, assuming the the tables have had time to converge after noticing that node 4 has left.

B Wireless

3. Brian, excited to setup a new adhoc wireless network on campus, uses the following topology with nodes *A*, *B*, *C*, and *D*. The dotted lines indicate the range of the wireless transmission from each node. For example, *C* is within transmission range of nodes *B* and *D*. In the following question, you will determine the outcome of transmissions that Brian has scheduled to take place in order to test his new network. Also assume that packets are never rescheduled for transmission, i.e., if a packet did not succeed the first time the node does not bother to retransmit it.



(a) In the first set of scheduled transmissions, carrier sense is enabled, but Brian has disabled RTS/CTS. Each transmission has a source, destination, associated relative start time, and total transmission time. For each transmission, briefly state the result and relate it to other transmission numbers if needed to justify your answer. **In this section, transmissions are never rescheduled.** A result would be "success" or "prevented by ...something..."

#	Src	Dst	Start Time	Duration	Type	Result
1	A	B	0	15	DATA	
2	C	B	5	10	DATA	
3	C	D	20	20	DATA	
4	B	C	22	8	DATA	
5	B	A	30	10	DATA	

(b) After the first series of transmissions, Brian enables RTS/CTS on each node and reschedules the same transmissions from *part a* with the same start and transmission times. For each scheduled transmission give the outcome **and** any additional transmissions that may result from them. Make sure to specify the type of packet using DATA, RTS, or CTS. Assign new transmission numbers, but consider the times of RTS/CTS packets to be negligible (do not assign times). You may not need all of the blank lines.

(Fill in the table on the next page)

#	Src	Dst	Time	Length	Type	Result
1	A	B			RTS	Success
2	B	A			CTS	Success
3	A	B	0	15	DATA	Success
4	C	B	5	10	DATA	Prevented: heard CTS#2

C Tools

4. In this question you will use the unix utility `dig` to explore the contents of DNS messages.

   The format of a `dig` request is simple. Just type: *dig www.mit.edu* to perform a look-up for that DNS name. As you now know, DNS requests can do more than just ask for the IP address corresponding to a single DNS name. Type *dig mit.edu ANY* to see DNS records of all types that are associated with the domain 'mit.edu'.

   (a) What IP address did the computer you are logged into contact to make the DNS request? Where do you think this server is located?

   (b) List all of the different types of records received as a result of your query. For each record, explain its purpose, using one of the entries provided in the reply as a concrete example.

   (c) Note that some of the names in the reply are not in the domain 'mit.edu'. Use the DNS names to find the general location of one of these servers. Where is it? Given the type of record, why would mit do this?

   (d) Use `dig` to find the names of two non-local servers you *could* contact in the process of identifying the nameserver for the domain 'cnn.com' (assume no DNS information is cached anywhere).

   (e) Use `dig` to find the TTL for the DNS mappings of 'www.cnn.com' and 'www.cs.stanford.edu'. What are they? If your boss asks you to provide two positive and two negative effects of having a short DNS TTL for the company's e-commerce site, what would you say?