Problem Set 4 Jack Sangdahl

Problem 1 Hashing

The hash table has 13 slots, and integer keys are hashed into the table with the following hash function H:

```
int H(int key) {
  int x = (key + 5) * (key - 3);
  x = (int)(x / 7) + key;
  x %= 13;
  return x;
}
```

1. Fill in the final hash table with the following keys: 17, 22, 73, 56,

310, 100, 230, 12, 42, 18, 19, 24, 49.

Slot	0	1	2	3	4	5	6	7	8	9	10	11	12
Contents	310, 49		18		22, 230, 42			100, 12, 24	73, 19	17	56		

1. List one or two methods that can handle collision hashing.

- **Separate chaining**: Each slot in the hash table points to a linked

list (or similar data structure). All keys that hash to the same slot are stored in the list, so collisions are handled by adding the new key to the list.

- **Open addressing**: Resolves collisions by finding another open slot

in the table using strategies like simply linearly searching for an open slot, quadratic probing, or double hashing to calculate the step size for probing when a collision occurs.

Problem 2 (Distributed Hash Tables)

There is a chord DHT in Figure 1 with 5 nodes. The finger tables are listed beside the nodes. Each node may be storing some items according to the Chord rules (Chord assigns keys to nodes in the same way as consistent hashing).

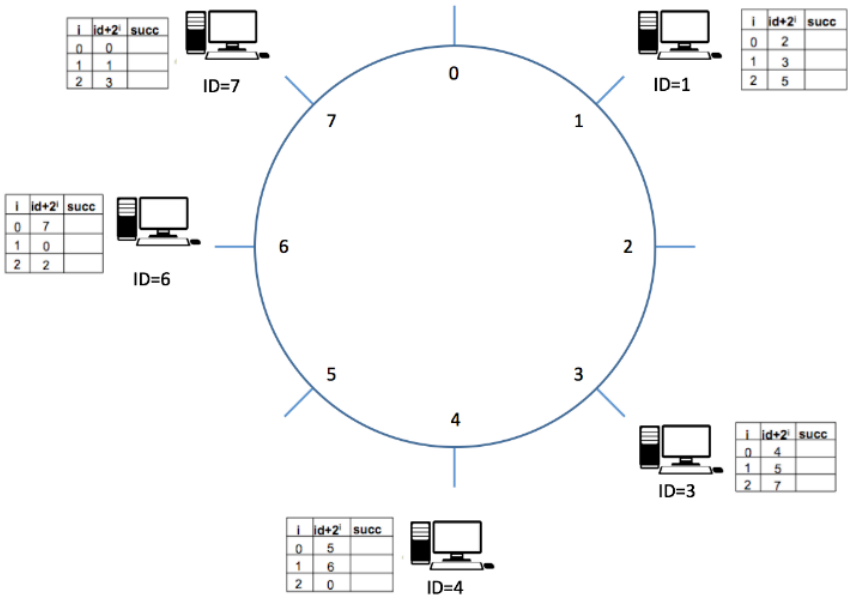

Figure 1

1. Fill in the table for node ID's 1 and 7.

ID 1	ID + 2′	Successor
0	2	2
1	3	3
2	5	5

ID 2	ID + 2′	Successor
0	0	0
1	1	1
2	3	3

1. List the node(s) that will receive a query from node 1 for item 5

(item named by key 5).

If node ID 1 is querying for item 5, Node 1 will check its finger table and find the closest entry that does not exceed 5. This entry is 5, meaning the success of 5 is Node 5 itself. Therefore, Node 1 will directly query Node 5 for item 5.

Problem 3 Bloom Filters

Derive the probability of false positive rate after 10 keys (or elements) are inserted into a table of size 100. Assume that 5 hash functions are used to setup bit positions in the table for the keys. 5 hashing functions mean for each key there will be 5 bits to set to 1 in the table (it can also be less than 5 if the hash functions generate the same output).

Assume m is the number of bits, n is the number of elements, and k is the number of hash functions used. After inserting one key, the probability of a particular bit being 0 is $\left(1 - \frac{1}{m}\right)^k$. This is because k hash functions are independent, and each hash function will have $\left(1 - \frac{1}{m}\right)$ probability for a particular bit remaining 0. Then after inserting n keys, the probability for a particular bit remaining 0 is $\left(1 - \frac{1}{m}\right)^{kn}$. Now apply this this to case of false positive.

Given the following:

- $m = 100$: number of bits in the table
- $n = 10$: number of keys/elements inserted
- $k = 5$: number of hash functions

Probability of bit being after inserting 1 key:

$$P(x) = \left(1 - \frac{1}{m}\right)^k$$

Probability of bit being 0 after inserting n keys:

$$P(x) = \left(1 - \frac{1}{m}\right)^{kn}$$

Probability of a false positive:

$$P(x) = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k$$

Substitute values:

$$P(x) = \left(1 - \frac{1}{100}\right)^{5\times10} = \left(\frac{99}{100}\right)^{50}$$

$$P(x) = \left(1 - \left(\frac{99}{100}\right)^{50}\right)^{5}$$

$$\left(\frac{99}{100}\right)^{50} \approx 0.605$$

$$P(x) = 1 - 0.605 = 0.395$$

$$P(x) = 0.395^5 \approx 0.0096$$

The probability of a false positive after inserting 10 keys into a table of size 100 with 5 hash functions is approximately **0.96%**.

Problem 4 P2P System

Three nodes A, B, and C are connected with each other via 5 MBps duplex link as shown in Figure 2. Node A wants to share a 2500 MB file to Node B and C. During the actual transmission, a 2.5 MB chunk of the file can be sent on the links each time. Ignore RTT in this problem.

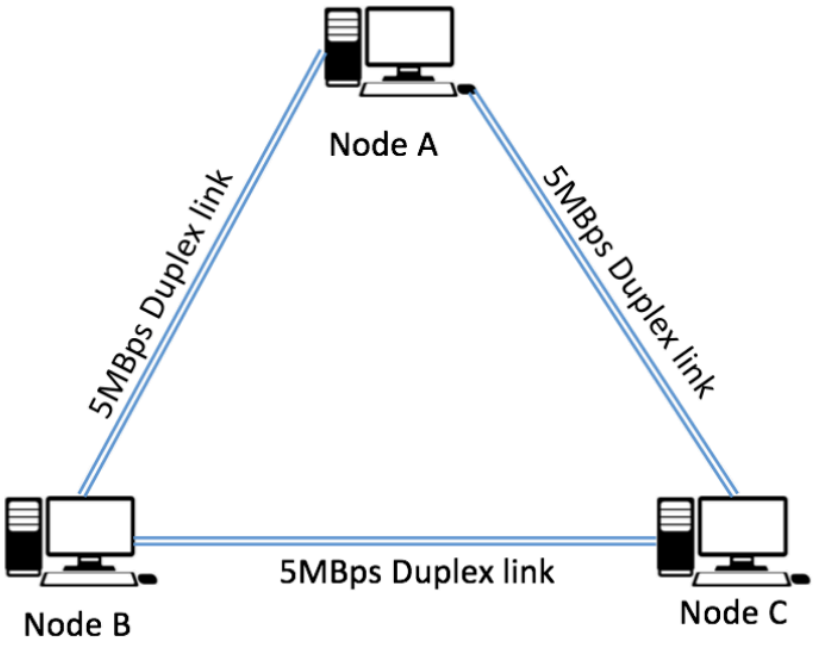

Figure 2

1. What is the time of the sharing process using a centralised approach?

(The process ends when B and C all received the file, and the centralised approach means B and C are communicating with A independently and there is no communication between B and C.)

Given the following:

- File size: 2500 MB
- Chunk size: 2.5 MB per transmission
- Link capacity: 5 MBps between each pair of nodes
- Ignoring RTT

With a duplex link, Node A can send data simultaneously to Node B and Node C. The time to transfer one chunk of 2.5 MB over a 5 MBps link is given by:

$$T_{\text{chunk}} = \frac{2.5}{5} = 0.5 \text{ seconds}$$

However, Node A can send a chunk to both B and C at the same time because the links are independent. So the time to send the entire 2500 MB file, split into 2.5 MB chunks is:

$$T_{\text{cent}} = \frac{2500}{5} = 500 \text{ seconds}$$

This lengthy time is because Node A can transmit to both nodes at the same time, and the bottleneck is the time required to send the full file to one node.

1. What is the ideal minimum time of sharing process P2P approach?

(P2P approach means after B and C received a chunk from A, they immediately share their piece to the other node.)

In the ideal P2P case, Node B and C work together perfectly to distribute the file. After the first chunk is transmitted, the additional upload bandwidth between Node B and C effectively halves the remaining time to distribute the remaining chunks. Meaning the minimum time for the P2P approach will be roughly half the centralised time because B and C are working together after transmitting the first chunk:

$$T_{\text{P2P}} = \frac{T_{\text{cent}}}{2} = \frac{500}{2} = 250 \text{ seconds}$$

Problem 5 File Distribution

Considering distributing a file of $F = 20$ gigabits to N peers. The server has an upload rate of $u_s = 30$ Mbps, and each peer has a download rate of $d_i = 2$ Mbps and an upload rate of u. For $N = 10, 100$ and $u = 300$ and $u = 2$ Mbps, prepare a chart giving the minimum distribution time for each of the combination of N and u for both client-server distribution and P2P distribution.

Client-Server

In this distribution, the server must upload the entire file to each peer. The time to distribute depends solely on the server's upload rate and the number of peers N. The distribution time for client-server is given by:

$T_{\text{cs}} = \frac{F}{u_s}$, where $F = 20$

Given the following:

- $F = 20$: the file size in gigabits
- $u_s = 30$: the server's upload rate in Mbps

This time is the same for any number of peers because the server needs to upload the entire file, regardless of how many peers download it.

$$T_{\text{cs}} = \frac{20}{30} = \frac{20 \times 10^3}{30} = 666.67 \text{ seconds}$$

U	$\boldsymbol{N = 10}$	$\boldsymbol{N = 100}$
300 Kbps	666.67 sec	666.67 sec
2 Mbps	666.67 sec	666.67 sec

Peer to Peer

In this distribution, peers both download and upload parts of the file. The time to distribute depends on both the server and peers' rates. The minimum distribution time in a P2P network is determined by two factors:

1. The server constraint, meaning the server must upload one copy of

the file, which we know from the above work is 666.67 seconds.

1. The peer constraint, meaning all peers must download the file, and

some must upload parts of it. This depends on the peer upload rate u, the peer download rate d_i, and the total number of peers N. For a P2P network, the total amount of data that needs to be transferred is $F \times N$. The total capacity in the network is the sum of the upload rates of the server and the peers. Therefore, the distribution is given by:

$$T_{\text{P2P}} = \max\left(\frac{F}{u_s}, \frac{F}{u_s + \sum_{i=1}^{N} u_i}\right)$$

Given the following:
- u_i is the upload rate of each peer
- u_s is the server's upload rate

For $u = 300$ Kbps:
- Each peer uploads at 0.3 Mbps
- For $N = 10$, the total upload rate is $0.3 \times 10 + 30 = 33$ Mbps
- For $N = 100$, the total upload rate is $0.3 \times 100 + 30 = 60$

Mbps

For $u = 2$ Mbps:
- Each peer uploads at 2 Mbps
- For $N = 10$, the total upload rate is $2 \times 10 + 30 = 50$ Mbps
- For $N = 100$, the total upload rate is $2 \times 100 + 30 = 230$

Mbps

Calculate the distribution times for each combination:

For $u = 300$ Kbps:
- $N = 10$:

$$T_{\text{P2P}} = \max\left(666.67, \frac{20 \times 10^3}{33}\right) = 666.67$$

- $N = 100$:

$$T_{\text{P2P}} = \max\left(666.67, \frac{20 \times 10^3}{60}\right) = 666.67$$

For $u = 2$ Mbps:
- $N = 10$:

$$T_{\text{P2P}} = \max\left(666.67, \frac{20 \times 10^3}{50}\right) = 666.67$$

- $N = 100$:

$$T_{\text{P2P}} = \max\left(666.67, \frac{20 \times 10^3}{230}\right) = 666.67$$

U	$N = 10$	$N = 100$
300 Kbps	666.67 sec	666.67 sec
2 Mbps	666.67 sec	666.67 sec

Problem 6 BGP

1. Give the types of business relationships in BGP peering and mention

who pays whom. What conditions make the internet stable?

- **Customer-provider**: where a customer pays a provider for transit

services to reach the rest of the internet. For example, in Figure 3, Node 1 is a customer of Node 3.

- **P2P**: where two networks of roughly equal size exchange traffic

with each other without any payment. Peers do not provide full transit to each other; they exchange traffic only for their customers. For example, in Figure 3, Node 4 and Node D are peers.

Conditions for internet stability:

- **Route leak avoidance**: Peers should not send traffic that violates

the business relationship (eg. a customer announcing a provider's route to another provider).

- **Routing policies**: Relationships should enforce policies where

providers can announce all destinations known to them, and peers should only announce customer routes to other peers

- **Global reachability**: Providers should provide access to all

reachable destinations on the internet, while peers should limit route announcements.

1. Identify which of the following paths are valid and which of them

are invalid based on the network topology of Figure 3.

Paths are valid or invalid based on the business relationships (customer to provider and peer to peer) and the typical routing behaviour where customers can send traffic to providers, providers cannot use customer links to transit to other providers, and P2P paths only allow traffic between their customers.

Path	Validity
1 3 d	Valid
1 4 d	Valid
8 d	Invalid
6 d	Invalid
4 d	Valid
7 5 d	Valid
7 5 3 d	Valid
2 1 3 d	Valid

Path	Validity
1 4 6	Invalid

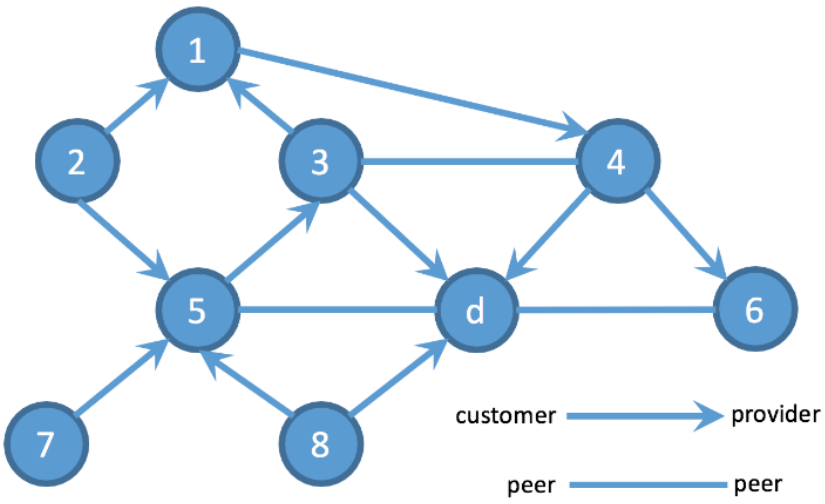

Figure 3

Problem 7 Security

Diffie-Hellman Symmetric key exchange solves the distribution issue of symmetric keys. Assume that Alice and Bob know p-ordered group G and a generator g, and Alice's and Bob's random seeds are a and b and their public keys are A and B respectively, and computes a shared key s. Use $p = 23$, $g = 11$, $a = 13$, and $b = 8$. Note that Eve is an eavesdropper and can see any communication between Alice and Bob.

Given the following:

- $p = 23$: the group order's prime number (modulo 23)
- $g = 11$: the value that generates the group
- $a = 13$: Alice's private key
- $b = 8$: Bob's private key

Begin by calculating public keys for Alice and Bob, respectively.

$$A = g^a \bmod p = 11^{13} \bmod 23$$

$$11^{13} \bmod 23 = 17$$

$$B = g^b \bmod p = 11^8 \bmod 23$$

$$11^8 \bmod 23 = 8$$

So Alice's public key is $A = 17$, and Bob's public key is $B = 8$.

Next calculate the shared key s for Alice and Bob, respectively.

$$s = B^a \bmod p = 3^{13} \bmod 23$$

$$3^{13} \bmod 23 = 9$$

$$s = A^b \bmod p = 15^8 \bmod 23$$

$$15^8 \bmod 23 = 4$$

So Alice's shared key is $s = 9$, and Bob's shared key is $s = 4$.

Eve can see their public key values $A = 17$ and $B = 8$, the generator value $g = 11$, and the prime number $p = 23$. However, without knowing the private values a and b, she cannot compute the shared key s without brute-forcing it (discrete logarithm problem, which is computationally infeasible for large enough numbers). Thus, Alice and Bob securely share the key $s = 21$, even though Eve can observe the public communication.