Problem 1 Hashing (20 pts)

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

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

(a) 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	**49**		**18**		**22,42 (Collision Here)**			**310,12,24 (Collision Here)**	**73,100,19 (Collision Here)**	**17**	**56**	**230**	

(b) List one or two methods that can handle collision in hashing.

Two most common ways in handling collision in hashing are:

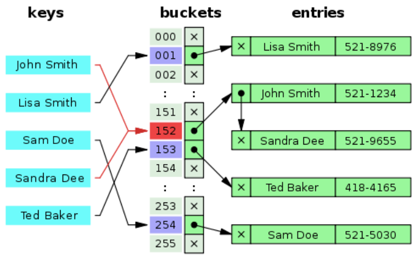
Linear probing - When a collision occurs using this method, we search for the next space that can hold the key value. This method operates linearly, so if a key at position 1 collides with another key at position 2, we check to see if position 2 is free; if it is, we move the key there; if it is not, we move on to the next position.

Quadratic Probing - If a key has experienced a collision in this method, we try quadratic probing to determine the next position using quadratic formulas. For instance, using the formula x2, we look for the next empty space. where x represents the number of probe times. We therefore look 12 (one element away) for the first probe, 22 (four elements away) for the next, and so on. After identifying a vacant space, we insert the key.

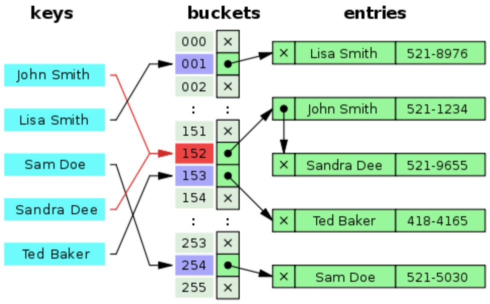
There are also other few different methods available to handle collision in hashing. The most popular are “Separate Chaining”, “Open Chaining/addressing”, “Coalesced hashing”,” Cuckoo hashing”, etc.

The first type of large method calls for the keys (or pointers to them) to be stored in the table along with the values they are associated with, which also includes:

Separate Chaining:



Open Addressing:



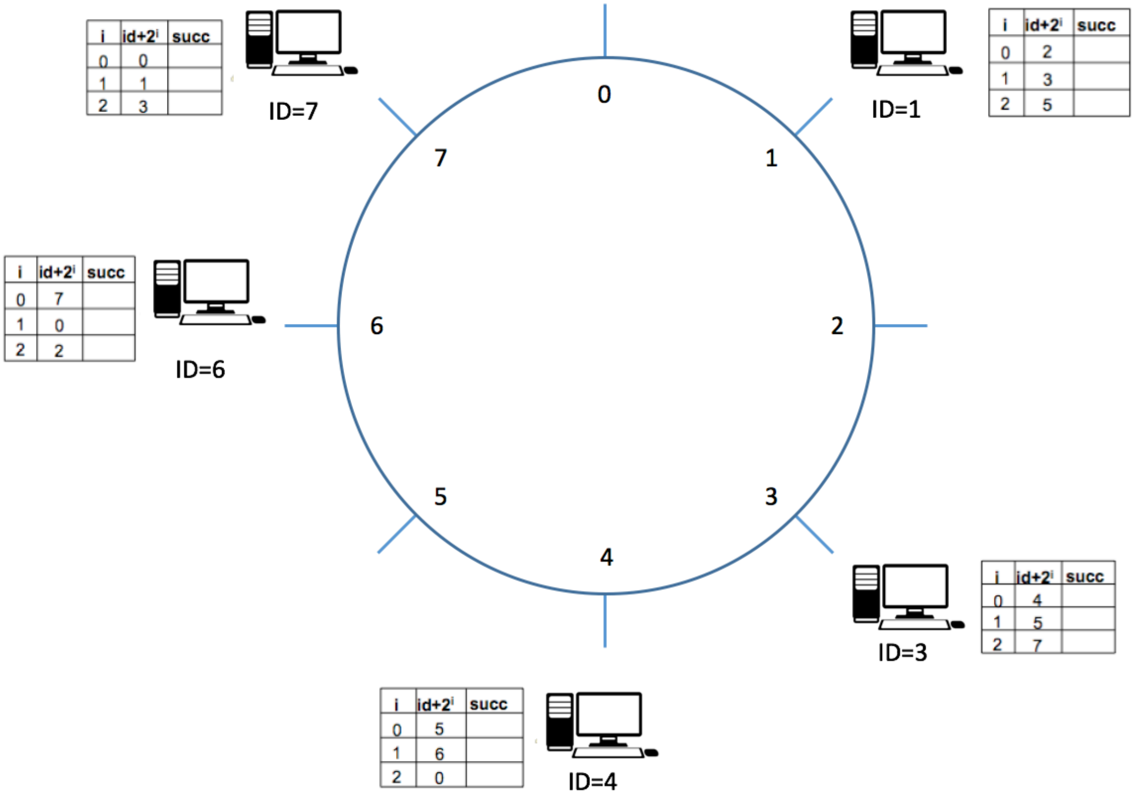
Another crucial technique for handling collision is dynamic resizing, which can be done in several different ways:

- Resizing by copying all entries
- Incremental resizing
- Monotonic keys

Problem 2 Distributed Hash Tables (20 pts)

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. Chord DHT for Problem 2



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

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

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

i	id+2^i	succ
0	4	
1	5	
2	7	

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

(a) Fill in the table for node id=1 and 7

	ID＋2^i	successor
0	2	3
1	3	3
2	5	6

Table for ID=1

	ID＋2^i	successor
0	0	1
1	1	1
2	3	3

Table for ID=7

(b) List the node(s) that will receive a query from node 1 for item 5 (item named by key 5)

Flow:

Node1 ⇨ Node6

Before performing any routing, we must first check the local storage. Since item 5 already exists locally, there is no need to forward the query to the largest node in the table that does not exceed the id.

The table entries show that node 5 maps to its successor 6. So, node 6 will house item 5.

Therefore, the Query would be: Node1 -> Node 6

Problem 3 Bloom Filters (10 pts)

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 (elements).

5 Hashing functions means: For each key (element) there will be 5 bits to be set to 1 in the table (it can also be less than 5 if the hash functions generate same outputs).

Assume “m” is the number of bits in the filter, “n” is the number of elements and “k” is the number of hash functions used then

For given condition in problem we have,
$m = 100 \quad n = 10 \; and \; k = 5$

After one key is inserted, probability of any bit being ‘0’ must be $(1-\frac{1}{m})^k$. Since we have around 5 hash functions to setup bit positions in table, each function is responsible for $\left(1-\frac{1}{m}\right)$ times probability for a particular bit to go 0.

So, after inserting 10 keys (n keys) probability will be $(1-\frac{1}{m})^{kn}$.

The definition of false positivity is that bloom filter thinks it has inserted an element but, it is not.

So, for the elements that’s not been inserted, probability of ‘k’ hashing functions pointing to all the bits that’s been already set to 1 has to be probability of false position.

$$P = (1 - \left(1-\frac{1}{m}\right)^{kn})^k \quad \text{............ eq(1)}$$

Substituting the values of m, n and k into equation 1 gives as follows:

$$P = (1 - \left(1-\frac{1}{100}\right)^{10*5})^5$$

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

$$P = 9.615 * 10^{-3} = 0.009615$$

Problem 4 P2P system (10 pts)

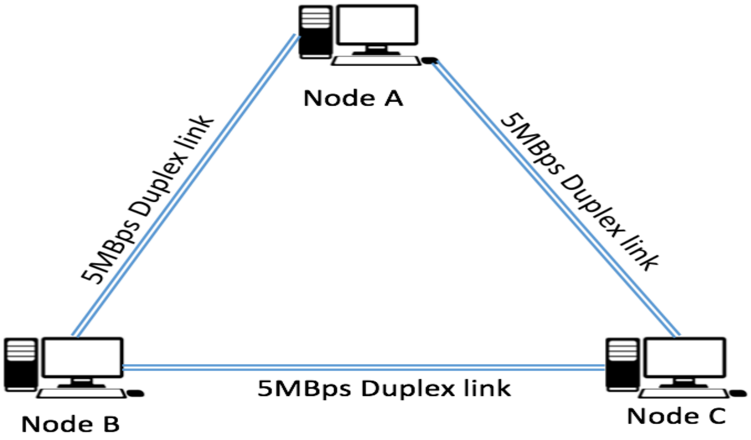


Figure 2. Network topology for Problem 4

3 nodes A, B and C relate to each other via 5MBps duplex link as is now in Figure 2. Node A want to share a 2500MB file to Node B and C. During the actual transmission 2.5MB piece of the file can be send on the links each time. In the problem we ignore the RTT delay.

(a) What is the time of the sharing process using centralized approach? (The process ends when B and C all received the file, and the centralized approach means B and C are communicating with A independently and there is no communication between B and C)

Using centralized approach time for sharing process to be completed can be calculated as follows:

$$Time\ Taken\ = maximum\ \{\frac{Data\ Size}{Bandwidth\ Between\ AB}\ ,\ \frac{Data\ Size}{Bandwidth\ between\ AC}\}$$

$$= \frac{2500\ MB}{5\ MBPS} = 500\ secs$$

Time taken using centralized method were communication between A and B & A and C takes independently is around ***500 seconds.***

(b) What is the ideal minimum time of the sharing process using P2P approach? (P2P approach means after B and C received a piece from A, they immediately share their piece to other party)

Using P2P approach each file different halves of a 2500 MB file and parallelly thy also constantly share what they have received between them. That means B and C can receive their part of file from A and in parallel B and C are also sharing data among themselves.

Exactly at time 250 Sec (Half of 1000 Sec), B has 1250 pieces of data from A and C has 1249 from A. C have 1250th part sent to hm and yet to send it to B.

To exchange the 1250th piece, time taken would be calculated as 2.5 MBPS/5 MBPS i.e 0.5 sec. Hence using P2P technique, it would need total of 50 Sec and 0.5 sec in to total i.e., ‘***250.5 Sec’*** for completing the sharing process.

Or to be more precise even ‘***250 sec’*** is a correct solution based on above requirement.

Problem 5 File Distribution (10 pts)

Consider distributing a file of F = 20 Gbits 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 and 100 and u = 300 Kbps and 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.

$Given,$
$F = 20\ Gbits = 20 * 1024\ Bps$
$u_s = 30\ Mbps$
$d_i = 2\ mbps$

Client Server

One can compute the minimum distribution for client-server distribution as,

$$\boldsymbol{Dcs = \max\{\frac{NF}{u_s}, F/d_i\}}$$

U	N= 10	N= 100
300 Kbps	10240	68266.67
2 Mbps	10240	68266.67

Peer to Peer

We use the following formula to determine the minimum distribution time for P2P distribution.

$$\boldsymbol{Dcs = \max\{\frac{NF}{u_s}, \frac{F}{d^i}, \frac{NF}{u_s + \sum_{i=1}^{N} ui}\}}$$

U	N= 10	N= 100
300 Kbps	10240	34538.07
2 Mbps	10240	10240

Problem 6 BGP (20 pts)

1. Give the types of business relationships in BGP peering and mention who pays whom. What conditions make the Internet stable? Explain each condition in a line or two.

BGP or Border Gateway Protocol, links autonomous systems, or AS, with one another. An outskirt switch in one AS (Autonomous System) makes friends with an outskirt switch in another AS using the Border Gateway Protocol, and the two routers then exchange routes or distances that are familiar to them.

Two common business relationships in BGP peering are: -

(1.) Provider-Customer
It's frequently used in this relationship as a way for companies and service providers to exchange information. The sum that the Client system pays the Provider system is entirely dependent on the volume of information that is produced and exchanged to accomplish their objective (and back). When a relationship of this nature exists, the provider system provides access to every route to its client.

(2.) Peer to Peer
Friendships between peers are not compensated, and each AS only reports its own paths., where the client's routes to other systems are among them. Providers with a global reach typically use these connections. Top-Tier Providers are another name for providers who make use of this connection.

There are a few requirements to make the internet stable: -

- The biggest factor is money.
- To provide access to the internet, customers must pay an internet service provider, and peers are based on the policy configuration based on the bilateral business relationship between ISPs.
- We adhere to the "Gao-Rexford" condition for global internet stability!
  - ○ Route Export: Don't share the routes you've learned with another peer or provider.
  - ○ Global topology: The customer-provider relationship graph is acyclic (A graph with no graph cycles)
  - ○ Route Selection: Routes through customers should be preferred over those through peers and providers.

Few other ways to make internet stable are :

A. Position the router in a high location.

Keep the router in a high location if the modem or router is installed at home so that it can receive a strong signal. It is not a good idea to keep it on the floor or under a coffee table because it reduces the router's connectivity.

B. Disconnect the router and plug it back in.
Remove the router and reattach it after 30 seconds if the speed is slow or the internet connection is unstable. This may help the data packets move quickly and improve the speed.

C. Use of the router's firmware
Some firmware, like Tomato or DDWRT, can aid in enhancing the routers' connectivity. Still, if it doesn't make the connection stable, consider buying a new router as an alternative.

2. Please identify which of the following paths are valid, which of them are invalid based on the network topology of Figure 3.

Path 1 3 d	Invalid Path	No link from 1-> 3 as it passes through customer
Path 1 4 d	Valid Path	Both are customer to provider path
Path 8 d	Valid Path	Given path is customer provider
Path 6 d	Valid Path	Given path is peer to peer
Path 4 d	Valid Path	Given path is customer to provider
Path 7 5 d	Valid Path	Given path is from customer to provider and finally peer
Path 7 5 3 d	Valid Path	All paths are customer to provider
Path 2 1 3 d	Invalid Path	No link as path passes through customer
Path 1 4 6 d	Valid Path	Given paths are customer to provider and provider to peer

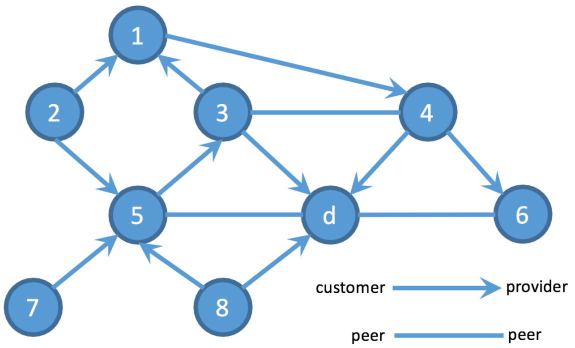


Figure 3. Network topology for Problem 6

Problem 7 Security (10 pts)

Diffie-Hellman Symmetric Key Exchange solves key the distribution issue of symmetric keys. Fill in the brackets below. 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**. Please 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, p = 23
g = 11
a =13
b = 8
Alice’s Key Generated = x = g^a mod p = 17
Bob’s Key Generated = y = g^b mod p = 8
Secret Key :
Alice(K_a) = y^a mod p =18
Bob(K_b) = x^b mod p = 18

Alice		**Bob**		**Eve**	
Known	Unknown	Known	Unknown	Known	Unknown
p = 23 ***g*** = 11 ***a*** = 13	***b***	***p*** = 23 ***g*** = 11 ***b*** = 8	***a***	***p*** = 23 ***g*** = 11	***a, b***
1. Alice chooses a private key ***a*** and sends its public key ***A*** to Bob					
A = [17]					
2. Bob chooses a private key ***b*** and sends its public key ***B*** to Alice					
		B = [8]			
3. Eve knows Alice’s public key ***A*** and Bob’s public key ***B***					
A, B		***A, B***		***A, B***	
4. Alice and Bob calculate its shared key ***s***					
***s* = [18]**		***s*** = [18]			***s***